PENERAPAN *WEBSOCKET* UNTUK TRANSMISI DATA PADA *IOT (INTERNET OF THINGS)* GUNA MENDUKUNG ERA INDUSTRI 4.0

Skripsi

Diajukan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Pendidikan Program Studi Pendidikan Teknik Informatika dan Komputer

oleh

Zulham Azwar Achmad

NIM.5302414040

PENDIDIKAN TEKNIK INFORMATIKA DAN KOMPUTER

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS NEGERI SEMARANG

2019

PERSETUJUAN PEMBIMBING

Nama	: Zulham Azwar Achmad
NIM	: 5302414040
Program Studi	: Pendidikan Teknik Informatika dan Komputer
Judul	: Penerapan Websocket Untuk Transmisi Data Pada IoT (Internet Of Things) Guna Mendukung Era Industri 4.0

Skripsi ini telah disetujui oleh pembimbing untuk diajukan sidang panitia ujian skripsi Program Studi Pendidikan Teknik Informatika dan Komputer, Fakultas Teknik Universitas Negeri Semarang.

Semarang, 22 Mei 2019

Pembimbing

Dr.-Ing. Dhidik Prastiyanto, S.T.,
M.T.,
NIP. 197805312005011002

ii

HALAMAN PENGESAHAN

Skripsi dengan judul "Penerapan Websocket Untuk Transmisi Data Pada IoT (Internet Of Things) Guna Mendukung Era Industri 4.0" telah dipertahankan di depan sidang Panitia Ujian Skripsi Fakultas Teknik UNNES pada tanggal 22 bulan mei tahun 2019

Oleh

Nama : Zulham Azwar Achmad

NIM : 5302414040

Program Studi : Pendidikan Teknik Informatika dan Komputer

Panitia :

Ketua	Sekretaris
Dr.-Ing. Dhidik Prastiyanto, S.T., M.T.,	Ir. Ulfah Mediaty Arief, M.T.
NIP. 197805312005011002	NIP. 196605051998022001

Penguji 1	Penguji 2/Pembimbing 1	Penguji 3/Pembimbing 2
Dr. Hari Wibawanto, M.T.	Anggraini Mulwinda, S.T., M.Eng.	Dr.-Ing. Dhidik Prastiyanto, S.T., M.T.,
NIP. 196501071991021001	NIP. 197812262005012002	NIP. 197805312005011002

Mengetahui:

Dekan Fakultas Teknik

Dr. Nur Qudus, M.T

NIP. 196911301994031001

iii

PERNYATAAN KEASLIAN

1. Skripsi ini, adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik (sarjana, magister, dan doktor) baik di Universitas Negeri Semarang (UNNES) maupun perguruan tinggi lain.
2. Karya tulis ini adalah murni gagasan, rumusan, dan penelitian saya sendiri, tanpa bantuan pihak lain, kecuali arahan Pembimbing dan masukan Tim Penguji.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.
4. Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari ditemukan terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya ini, serta sanksi lainnya sesuai dengan norma yang berlaku di perguruan tinggi ini.

Semarang, 22 mei 2019

Yang membuat pernyataan,

Zulham Azwar Achmad

NIM. 5302414040

iv

MOTTO DAN PERSEMBAHAN

“I think young generation is always better than last generation. No matter you like it or don't like it. My father said, 'Jack, I'm so good, you'll never be' - but I'm better than him. My father is better than my grandfather. My children will be better than us.” - Jack Ma (Founder and CEO Alibaba Group).

“Sukses itu bukan tentang materi dan popularitas, Sukses itu ketika sudah bermanfaat bagi diri sendiri dan orang lain.”

Skripsi ini saya persembahkan untuk :

1. Tuhan Yang Maha Esa atas berkat dan rahmatNya yang selalu memberikan kelancaran dan kemudahan hingga skripsi ini dapat terselesaikan.
2. Kedua Orangtua tercinta yang senantiasa dengan tulus, sabar mendoakan saya, memberikan semangat untuk kebaikan dan keberhasilan saya.
3. Teman-teman seperjuangan teman-teman PTIK angkatan 2014.

Abstrak

Zulham Azwar Achmad, 2019, PENERAPAN WEBSOCKET UNTUK TRANSMISI DATA PADA IOT (INTERNET OF THINGS) GUNA MENDUKUNG ERA INDUSTRI 4.0, Dr.-Ing. Dhidik Prastiyanto, S.T., M.T., Pendidikan Teknik Informatika dan Komputer

Di era digital saat ini internet menjadi suatu kebutuhan pokok manusia. berbagi informasi yang sering digunakan di mana-mana, salah satunya adalah media sosial, rumah pintar, kecerdasan buatan, dan industri 4.0 itu sendiri. dimana semua data disimpan dan dianalisis untuk diproses. Karena bagian dari *Internet of Things* adalah *cloud computing* dimana membutuhkan *server* untuk mengatasi lalu lintas data, untuk itu trafik data sangat padat sehingga dari metode pengiriman data diperlukan agar trafik data menjadi lebih efisien dan lebih cepat.

Lalu lintas yang padat yang akhirnya menyebabkan banyak masalah, salah satunya adalah penggunaan *bandwidth* yang berlebihan, dan transmisi data yang tidak stabil, transmisi data yang biasa digunakan adalah HTTP (*HyperText Transfer Protocol*) seperti HTTP *Long Polling* berarti pengiriman data menggunakan interval waktu dan membutuhkan dua jalur komunikasi yaitu komunikasi ketika meminta data dan mengirim data. untuk mengatasi masalah ini diperlukan transmisi data satu arah sehingga tidak banyak lalu lintas yang padat dan pengiriman yang lebih cepat. Pengiriman satu arah biasanya disebut websocket, di mana data dikirim dengan koneksi TCP (*Transmission Control Protocol*) tunggal dengan komunikasi dupleks penuh. Pengujian pada metode HTTP Long Polling dan dilakukan menggunakan mikrokontroler. Hasil pengujian menggunakan metode dalam hal kecepatan menunjukkan respon data lebih efektif daripada Long Polls.

Hasil pengujian konsumsi RAM (*Random Access Memory*) menggunakan metode Polling Panjang lebih kecil dari penggunaan memori dalam metode websocket. Hasil pengujian bandwidth menunjukkan bahwa metode Long Polling mengkonsumsi bandwidth yang besar dibandingkan dengan Websocket. CPU (*Central Processing Unit*) Hasil Pengujian Penggunaan menunjukkan tingkat beban CPU yang stabil dibandingkan dengan Long Polling. Dari beberapa bagian pengujian yang dilakukan berdasarkan jajak pendapat panjang, metode ini lebih baik memenuhi kebutuhan berbasis IoT (*Internet of Things*) waktu nyata.

Kata Kunci — Websocket, Polling panjang, Internet of things, Era Industri 4.0, Transmisi Data, IoT

KATA PENGANTAR

Atas Berkat Rohmat Allah Yang Maka Kuasa yang telah melimpahan rahmat dan hidayahnya sehingga dapat terselesaikan skripsi yang berjudul "Penerapan Websocket Untuk Transmisi Data Pada IoT (Internet Of Things) Guna Mendukung Era Industri 4.0".

Proses penyusunan skripsi ini tidak terlepas dari dukungan berbagai pihak, sehingga pada kesempatan ini penulis ingin memberikan rasa hormat dan mengucapkan terima kasih kepada:

1. Dr.-Ing Dhidik Prastiyanto,S.T.,M.T., Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Negeri Semarang dan dosen pembimbing yang telah membimbing dan memberikan arahan sehingga skripsi ini terselesaikan.
2. Ir. Ulfah Mediaty Arief, M.T., Ketua Prodi Pendidikan Teknik Informatika dan Komputer.
3. Bapak,Ibu dosen dan staf di jurusan Teknik Elektro Universitas Negeri Semarang yang telah memberikan ilmu yang berguna bagi penulis.
4. Teman-teman PTIK yang banyak membantu penulis dan selalu memberikan semangat dalam pengerjaan skripsi.
5. Pasukan Wawa Apartemen yang selalu support dalam penulisan skripsi.

Segala budi yang telah diberikan semuanya diserahkan kepada Tuhan Yang Maha Kuasa. Semoga hasil penelitian ini bermanfaat bagi pembaca khususnya dan perkembangan pendidikan pada umumnya.

DAFTAR ISI

Halaman

BAB I

PENDAHULUAN

1.1. Latar Belakang

Era generasi teknologi perindustrian telah mengalami perkembangan-perkembangan. Perkembangan di era revolusi industri yang pertama dimulai dari mekanisme tenaga uap dan air dari generasi tahun 1800s, hal tersebut membawa permulaan transisi dari pekerjaan secara manual ke proses manufaktur yang pertama, revolusi industri yang kedua telah mengalami perubahan yang dipicu oleh elektrifikasi yang membuat produksi industri meningkat, di era revolusi industri yang ketiga telah timbul karakteristik digital yang mengenalkan *microelectronics* dan *automation*.

Sedangkan pada masa era revolusi industri yang ke-empat telah terjadi perubahan oleh developer-developer *ICT* (*Information and Communication Technologies),* dimana teknologi berbasis sistem kecerdasan komputer yang distabilkan oleh pengontrol yang saling terhubung satu sama lain (*IoT Functionalities),* hasil dari teknologi model baru ini untuk sistem produksi industri yaitu terjadinya peningkatan produksi industri dan hasil produksi yang stabil (Rojko, 2017).

Salah satu contoh penerapan IoT dibidang industry yaitu tentang pengolahan data. Pengolahan data merupakan hal dasar yang biasa dilakukan manusia, dan banyak dari data tersebut dimanfaatkan oleh manusia seperti halnya mengolah data angka, huruf, gambar dan lain-lain. Kaitannya dengan teknologi IoT

banyak sekali data-data penting yang tidak bisa di olah secara langsung oleh manusia, karena dalam hal ketelitian, ketepatan data, pengiriman data dan keterbatasan sumber daya manusia itu sendiri. Oleh sebab itu diciptakannya konsep teknologi *Internet Of Thing* (IoT).

IoT adalah paradigma komunikasi baru-baru ini yang membayangkan masa depan yang dekat, dimana objek dalam kehidupan sehari-hari akan dilengkapi dengan mikrokontroler, trans-komunikasi untuk komunikasi digital, dan susunan *protocol* yang sesuai yang akan membuat mereka dapat berkomunikasi satu sama lain dengan pengguna dan menjadi bagian integral dari internet (Zanella, Bui, Castellani, Vangelista, & Zorzi, 2014).

Teknologi IoT masih dalam tahap awal untuk mengimplementasikan ke industri, lingkungan masyarakat, dan industri keilmuan (Roblek, Meško, & Krapež, 2016). Industri 4.0 ialah spesialisasi dari penerapan IoT di peralatan industri dan manufaktur-manufaktur lainnya. Itu berarti dibutuhkan transmisi dan pengolahan data secara *real-time* untuk menanggulangi masalah pengolahan data, analisis *Big Data* dan *Cybersecurity* (Rojko, 2017).

Metode pengiriman data yang saat ini masih banyak digunakan adalah metode pengiriman data *Hyper Text Transfer Protocol* (HTTP) *Long Polling*, selain itu juga terdapat metode pengiriman yang lain seperti yang pernah diteliti oleh *Oliver Ornmyr* yang berjudul "*Performance comparison of XHR polling, Long Polling, Server Sent Event and Websocket*" dimana ia menyebutkan *Websocket* dan *Server sent event* memiliki performa lebih unggul dibandingkan eksperimen penelitian yang lain, lalu dibuktikan bahwa XML HTTP Request (XHR) *Polling*

memiliki *network traffic* lebih besar dari *Long Polling*. Performa *Long Polling* terbukti empat kali lebih baik dibandingkan XHR *Polling* (Örnmyr, n.d.). Namun pada kasus penelitian yang lain kelemahan dari metode HTTP *Long Polling* ialah sering terjadi *request* data ke server walaupun tidak ada pesan baru terhadap *client* (Rakhunde, 2014). Dan dari pengalaman yang pernah dilakukan oleh penulis, penulis melakukan uji coba terhadap penerapan konsep IoT untuk mengontrol robot dari jarak jauh, dan pemicu yang digunakan ialah "waktu" dimana *client (Robot Microcontroller)* akan melakukan *request* setiap 200ms agar mendapat data secara *continue* dan *up-to-date*.

Namun kelemahan dari metode *HTTP Long Polling* ialah penggunaan *bandwidth* internet yang tidak efisien karena akan terus melakukan *request* data ke *server* agar mendapat data terbaru dan banyak *request* yang tidak diperlukan. Hal ini juga membuat kinerja mesin akan terus diforsir selama masih berjalan. Lalu dalam penelitian yang lain menyebutkan bahwa untuk berkomunikasi antar klien dan server metode pengiriman yang biasa digunakan ialah *Polling, Long Polling* dan *Comet.* Dan dalam penelitian tersebut *Websocke*t adalah metode pengiriman data yang baru (Comert, 2016) dan menemukan bahwa banyak kerugian yang didapatkan ketika menerapkan metode pengiriman *Polling, Long Polling dan Comet*, banyak *request* yang tidak dibutuhkan ketika memperbaharui data yang berkelanjutan.

Alasan utama menggunakan W*ebsocket* ialah untuk menggantikan metode konvensional *Comet* dan HTTP *Polling/ Long Polling*, untuk meningkatkan pengiriman data yang akurat pada aplikasi web, perbedaan terbesar *Websocket*

dengan metode yang lain ialah websocket tidak mengikuti metode *request/response* yang sama dengan yang lain.

Penelitian tersebut hanya membandingkan metode pengiriman data di satu server dan *platform* yang sama, tidak ada penyeberangan lalu lintas *platform* diantaranya itu berarti tidak ada penerapan *Internet of Things* didalamnya. Dan sebagai penguatan selanjutnya peneliti mengajukan judul ini seperti yang disebutkan oleh *Rojko* yang berjudul "*Industry 4.0 Concept: Background and overview*" bahwa industry 4.0 ialah spesialisasi dari *internet of Things* dimana performa pengiriman data menjadi isu yang saat ini sedang dirasakan untuk mengatasi masalah *huge data* dan *cybersecurity*. Dan dalam penelitiannya ia juga mengatakan jika industry 4.0 dapat mengatasi solusi-solusi diantaranya dapat mengurangi biaya produksi 10% - 30%, biaya *logistic* berkurang 10% - 30% dan *quality management cost* sebesar 10% - 20% (Rojko, 2017).

Berdasarkan permasalahan diatas, maka judul penelitian ini yaitu "Penerapan *Websocket* untuk Transmisi Data pada IoT(*Internet of Things*) guna Mendukung Era Industri 4.0". Menerapkan metode transmisi yang benar-benar *real-time* seperti *Websocket* dan menerapkannya pada mobile platform, web *platform* dan *microcontroller*, karena pengiriman data pasti akan selalu menyeberangi *platform* yang berbeda agar selanjutnya dapat di olah dengan cara yang berbeda-beda. *WebSocket* adalah protokol yang memungkinkan komunikasi *dupleks* penuh antara klien dan web server. Protokol ini bekerja melalui *Transmission Control Protocol* (TCP) tunggal, seperti HTTP dan dibuat sebagai alternatif untuk polling berulang (Fette & Melnikov, 2011). Berbeda dengan

metode HTTP Long Polling yang akan bekerja ketika ada permintaan dari klien dan mengirimkan data terbaru ke klien yang artinya membutuhkan dua jalur transmisi antara klien dan server yaitu kondisi *request* dan kondisi *response*. Jika ada pesan baru yang datang ketika server telah menanggapi permintaan dan masih belum ada permintaan datansg, server harus menyimpan pesan dan menunggu permintaan berikutnya(Shuang & Feng, 2013).

Sedangkan Websocket akan merespon dan akan memberikan data terbaru secara langsung ketika server yang bekerja sebagai websocket menerima data dari dalam system maupun dari luar, dan bisa dari klien maupun jalur *Application Programming Interface* (API), maka dari itu *Websocket* dikatakan sebagai protocol TCP dengan koneksi tunggal yang artinya ia hanya membutuhkan satu jalur pengiriman data yang dibutuhkan oleh websocket untuk mengerjakan tugasnya ke klien-klien yang memiliki hubungan jalur terhadap *server* atau *port*. Penerapan *Websocket* untuk transmisi data ini diharapkan dapat digunakan untuk membuat pengiriman data digital menjadi lebih akurat, *real-time*, dan penggunaan memori lebih efisien guna membangun dunia industri yang lebih baik.

1.2. Identifikasi Masalah

Berdasarkan latar belakang yang telah dikemukakan diatas, maka masalah diidentifikasi sebagai berikut:

1) Perlunya pengiriman data yang cepat.
2) Perlunya pengiriman data yang hemat *bandwidth* dan *memory*.
3) Perlunya pengiriman data yang tidak membebani perangkat.

4) Perlunya perbandingan untuk membuktikan metode pengiriman mana yang lebih baik.

1.3. Pembatasan Masalah

Agar permasalahan diatas tidak meluas dan dibahas secara mendalam, maka penulis membatasi masalah yang dibahas pada aspek:

1) Penelitiam ini tidak berfokus pada uji praktik transmisi W*ebsocket* di bidang industry secara langsung.
2) Penelitian ini hanya memfokuskan pada uji kelayakan websocket dengan cara menganalisis kinerja dan performa serta membandingkan transmisi data *Websocket (Inovasi)* dengan metode konvensional (HTTP Long Polling) dari segi performa secara *real-time*, efisiensi data, *bandwidth*, dan beban kinerja mesin untuk mengetahui apakah *Websocket* layak digunakan pada IoT di era industry 4.0
3) Penerapan IoT (*Internet of Things*) menggunakan *microcontroller* dan *mobile application* sebagai *client*, dan *Dedicated Cloud server* sebagai *interface dan dedicated cloud server Websocket* sebagai monitoring pengiriman data.

1.4. Rumusan Masalah

Berdasarkan latar belakang masalah yang telah diuraikan sebelumnya, maka penulis merumuskan masalahnya yaitu untuk membuktikan metode pengiriman data websocket dapat menangani masalah masalah efisiensi data, akurasi data, kecepatan dan ketepatan data, bandwidth dan konsumsi memori, serta kinerja mesin yang diterapkan pada IoT di era industry 4.0 dengan cara

membandingkan websocket (Inovasi) dengan metode konvensional (HTTP *Long Polling*).

Untuk memperjelas perbandingan yang ingin dicapai dalam pembuatan aplikasi ini, maka penulis akan memaparkannya agar lebih detail yaitu:

1) Berapa lama waktu yang diperlukan untuk mengirimkan data menggunakan websocket dan HTTP Long Polling pada IoT?
2) Berapa banyak bandwidth yang dihabiskan untuk mengirimkan data menggunakan websocket dan HTTP Long Polling pada IoT?
3) Berapa beban CPU yang diperlukan untuk mengirimkan data menggunakan websocket dan HTTP Long Polling pada IoT?
4) Berapa banyak memori yang dihabisan untuk mengirimkan data menggunakan websocket dan HTTP Long Polling pada IoT?
5) Bagaimana perbandingan antara metode pengiriman data *websocket* dengan metode pengiriman data konvensional (HTTP *Long Polling*)?

1.5. Tujuan Penelitian

Maksud dari penelitian yang dilakukan adalah untuk menerapkan metode pengiriman data *Websocket* dengan cara menganalisis kinerja dan performa serta membandingkan dengan metode konvensional. Apakah dengan hasil tersebut *Websocket* layak digunakan dan dapat meningkatkan kinerja sistem *Internet of Things* guna mendukung era *industry* 4.0, adapun tujuan dari penelitian ini adalah sebagai berikut :

1) Membangun *prototype* transmisi data pada *Internet of Things* yang lebih efektif, efisien dan *real-time*.

2) Membandingkan metode pengiriman data *Websocket* (inovasi) dan metode pengiriman data konvensional (HTTP *Long Polling*) dari segi efisiensi data, akurasi data, kecepatan dan ketepatan data, bandwidth dan konsumsi memori, serta kinerja mesin pada *Websocket* dan HTTP *Long Polling*.
3) Membuktikan hasil dari metode transmisi data *Websocket* apakah dengan penelitian ini websocket layak digunakan sebagai metode pengiriman data pada IoT (*Internet of Things*) di era industry 4.0.

1.6. Manfaat Penelitian

Adanya penelitian ini diharapkan dapat memberikan manfaat baik secara teoritis maupun teknis:

1) Manfaat teoritis

   Hasil penelitian ini diharapkan dapat diterapkan diberbagai jenis industri dan dapat meningkatkan hasil produksi serta kualitas produk. Penerapan dengan menggunakan metode ini masihlah sangat jarang dan sumbernya juga masih sangat sedikit, jadi diharapkan konsep teknologi ini dapat di ajarkan diberbagai lembaga perguruan tinggi.

2) Manfaat Teknis

   Bagi peneliti dan Industri Dapat diterapkan usaha milik pemilik dimana usaha tersebut bergerak di dunia *digital marketing*, *software house* dan banyak dari pekerjaan tersebut bekerja sama di dunia perindustrian maka manfaat dari skripsi ini sangat berpengaruh dalam perkembangan teknologi di era saat ini.

BAB II

KAJIAN PUSTAKA DAN LANDASAN TEORI

2.1 Kajian Pustaka

Berdasarkan *Journal of New Results in Science* oleh O. Comert dengan judul "*The Comparison of New Web Communication Method Websocket with Traditional Method*" dimana tujuan penelitian tersebut untuk membandingan metode *Websocket* dengan metode tradisional dari segi kecepatan transaksi dan banyak data yang dikirimkan dari *server* terhadap *client* begitu pula sebaliknya yang menggunakan jenis *platform* yang sama yaitu *Web* degan spesifikasi sebagai berikut :

Sistem Operasi : Ubuntu 14.04..3

Processor : Intel® Xeon® CPU E7-2850 @2.00GHz

Memory : 8GB

Dari hasil penelitian data yang dikirim dari *client* ke *server*, didapatkan 18 eksperimen yang berbeda, yaitu:

Kondisi ke-1 : Tiap 1 detik, 1 byte data dikirim.

Kondisi ke-2 : Tiap 1 detik, 100 byte data dikirim.

Kondisi ke-3 : Tiap 1 detik, 1 kilobyte data dikirim.

Kondisi ke-4 : Tiap 25 detik, 1 byte data dikirim.

Kondisi ke-5 : Tiap 25 detik, 100 byte data dikirim.

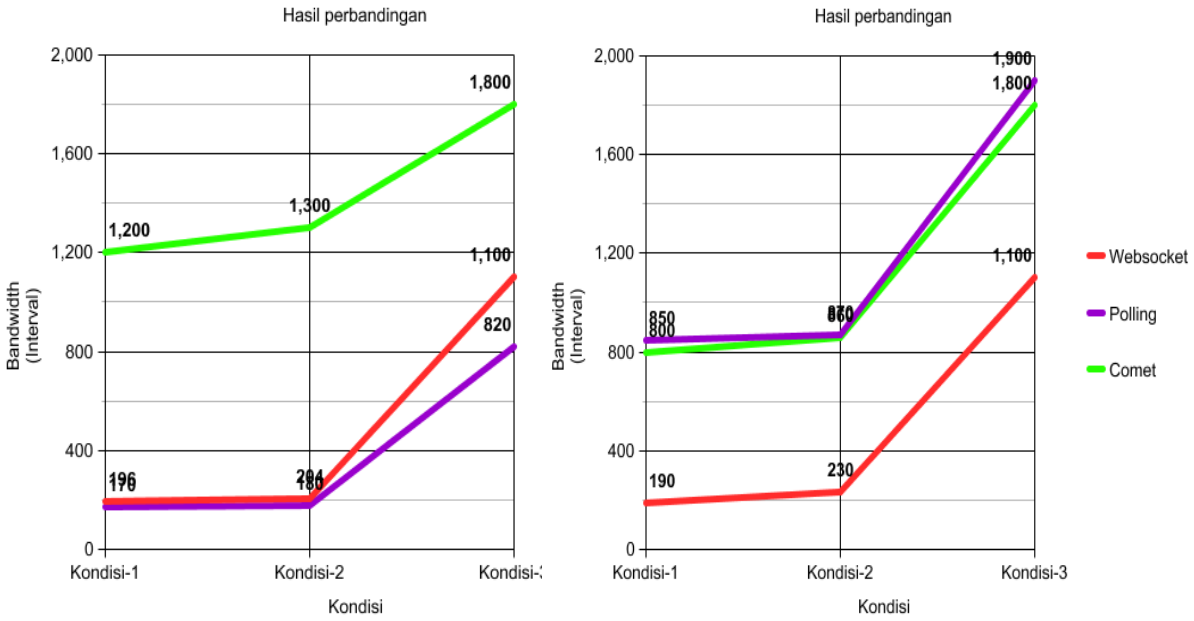
Kondisi ke-6 : Tiap 25 detik, 1 kilobyte data dikirim.

Kondisi ke-7 : Tiap 60 detik, 1 byte data dikirim.

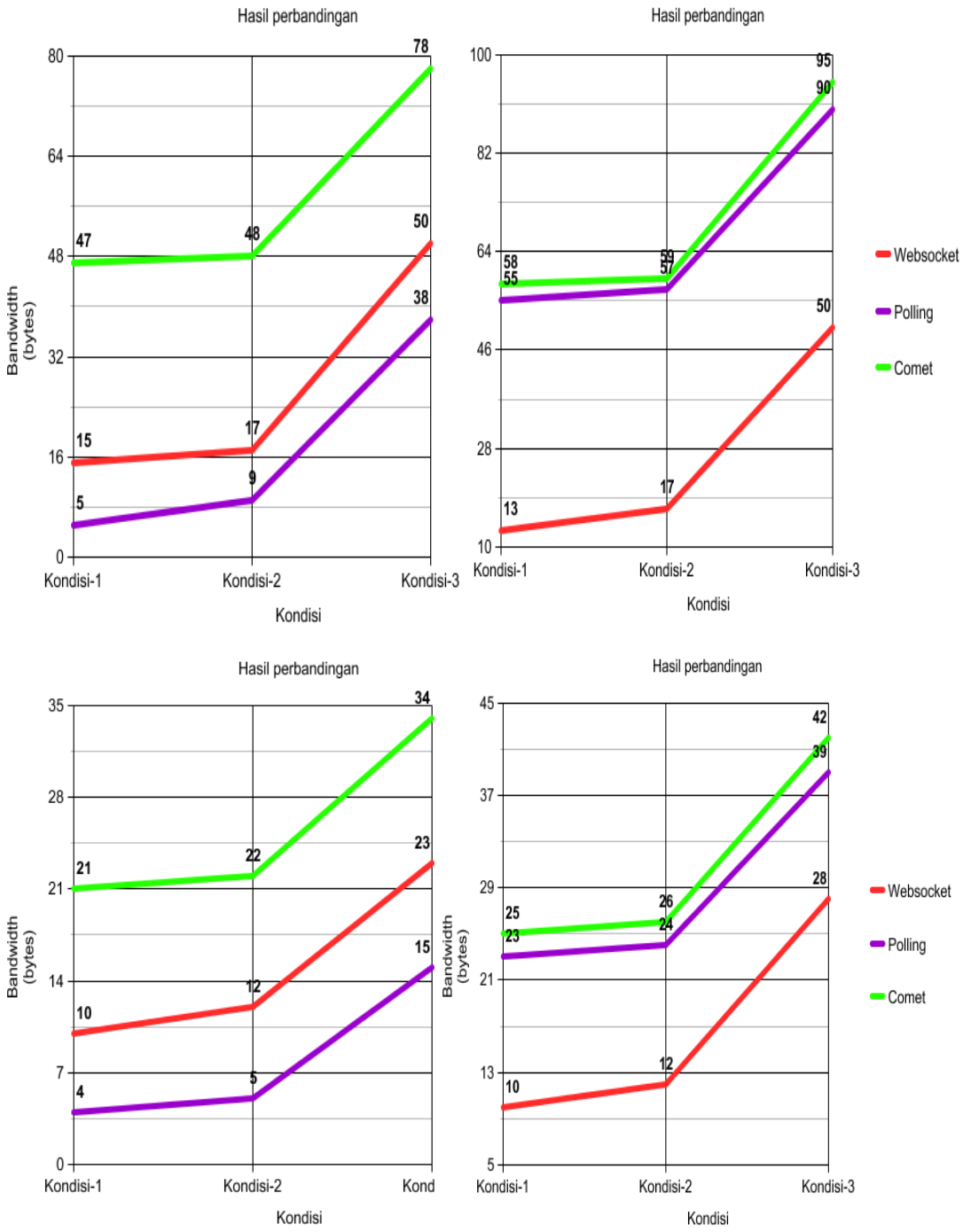
Kondisi ke-8 : Tiap 60 detik, 100 byte data dikirim.

Kondisi ke-9 : Tiap 60 detik, 1 kilobyte data dikirim.

Hasil dari eksperimen diperlihatkan dalam bentuk grafik dibawah



Gambar 2.1 Kondisi-1, Kondisi-2, Kondisi-3



Gambar 2.2 Kondisi-7, Kondisi-8, Kondisi-9

Dari hasil yang dihasilkan oleh O.Comert, penggunaan *Long Polling* lebih efisien untuk pengiriman data secara terus-menerus dari *server* ke *client*. Ini karena metode tersebut akan terus berjalan yang membuat performa nya lebih baik dari *Websocket* dan Comet.

Dan di sisi lain dimana data dikirim secara terus menerus, metode *Websocket* terbukti lebih efisien dari pada *Long Polling* dan Comet dari segi *bandwidth*. Ini karena *Long Polling* dan Comet membutuhkan *request* terhadap *server* secara terus menerus untuk agar data terkirim dengan baik, ini artinya lalu lintas data nya sangat tinggi.

Hasilnya metode *Websocket* diketahui mempunyai performa lebih tinggi dibandingkan metode *Long Polling* dan Comet. Indikasi dari *Websocket* dapat diterapkan di berbagai aplikasi di masa mendatang, terlepas akan metode baru kedepannya.

Dari penelitian yang dilakukan oleh O.Comert peneliti juga sependapat dari hasil tersebut, namun menurut peneliti penelitian milik O.Comert tidak memperhitungkan dari segi performa mesin dan *big data* dan kelemahan perbandingan milik O.Comert menurut peneliti hanya membandingkan dari satu jenis *platform* saja. Dimana di dunia perindustrian pengiriman data dapat dikirimkan dari berbagai jenis *platform* seperti android/ios dan mikrokontroller.

Dengan demikian adanya penelitian dengan tema serupa milik peneliti akan memperhitungkan dari segi performa mesin dan efisiensi big data dan penerapan *Internet of Things* dari jenis *platform* yang berbeda maka akan ditemukan hasil yang lebih kuat.(Comert, 2016)

Berdasarkan penelitian *Faculty of Computing at Blekinge Institute of Technology* milik O.Ornmyr yang mengatakan motivasinya melakukan penelitian ini karena banyaknya *website* di internet yang memperbarui isi halaman web secara dinamis tanpa membuka ulang halaman web tersebut. Ini menjadi sesuatu yang teknologi yang dapat dikembangkan. Sebagai contohnya dalam website dengan fitur *chat*, blog, web berbasis email untuk menyediakan halaman web. Ini sangat penting bagi perusahaan-perusahaan untuk mengurangi biaya ketika menambah kapasitas *server*. spesifikasi *hardware* yang kecil dengan metode efisiensi dari *software* dapat membuat potensi *hardware* meningkat walaupun memproses aplikasi yang besar sekalipun.

Penelitian ini dapat membuat pengaruh terhadap teknologi yang menggunakan Webpage yang dinamis dan masalah performa. Performa yang lebih baik dapat meningkatkan kemampuan pengguna menggunakan website.

Dan dari tujuan penelitian tersebut yaitu agar dapat menciptakan metode yang lebih eifisien walaupun menggunakan *hardware* dengan spesifikasi rendah akan tetap berjalan optimal. Serta dapat mengurangi biaya *hardware* untuk meningkatkan spesifikasi *hardware*. Dimana penelitian milik O. Ornmyr menggunakan *hardware* dengan spesifikasi berikut:

Sistem Operasi : Windows 10 64bit terinstall di SSD

Processor : Intel® i5 4670K CPU 3.40 GHz

SSD : 840 EVO 250 GB

HDD : TOSHIBA DT01ACA200 2TB

Memory : 16GB

Terdapat tiga jenis tes performa dalam penelitian ini untuk membandingkan performa masing-masing teknologi. Tiap tes dites dua kali, dengan atau tanpa mengambil resource. Tiap tes akan selesai antara 40 sampai 60 menit agar lebih akurat. 100 simulasi *client* digunakan di penelitian ini.

Deskripsi Kondisi perbandingan ke-1 :

100 simulasi pengguna

500ms interval pesan ke *server*

500ms interval XHR *Polling*

Tes ini selesai dengan membandingan teknologi di kondisi yang serupa. Pengambilan data XHR-*Polling* interval, rate sama dengan teknologi yang lain. Hasil yang sebanding antara ketiga teknologi mempunyai perintah yang sama yaitu mengirim permintaan dengan interval waktu sama-sama 500ms.

Deskripsi Kondisi perbandingan ke-2 :

100 simulasi pengguna

2000ms interval pesan ke *server*

500ms interval XHR *Polling*

Deskripsi Kondisi perbandingan ke-3 :

100 simulasi pengguna

2000ms interval pesan ke *server*

2000ms interval XHR *Polling*

Dan hasil dari perbandingan ini ditemukan bahwa *Websocket* dan *Server Sent Event* memiliki performa lebih baik dari pada jenis metode yang lain melalui uji coba yang sudah dilakukan oleh O.Ornmyr. Dan faktor yang mempengaruhi komunikasi antara *server* dan klien adalah :

1. Jumlah koneksi TCP yang terhubung ke *server*
2. Jumlah data yang diterima dan proses *request* terhadap *server*. Ini berlaku hanya untuk metode XHR *Polling* dan *Long Polling*
3. Menggenerasi *overhead* data dari *server* ke klien
4. *Overhead* data

Kesimpulan dari penelitian ini ialah pemilihan dari teknologi yang akan dipakai sangat berpengaruh ketika berhadapan performa web *server*. Bagaimana masalah yang akan timbul tergantung dari besarnya data yang akan diambil.

Websocket dan *Server* Sent Event dari hasil penelitian diketahui memiliki kesamaan performa dan menjadi metode yang paling efisien di penelitian ini. Saran dari penelitian ini akan meningkatkan performa dan mengurangi biaya *hardware* dengan hasil yang optimal.

Faktor yang mempengaruhi performa *server* ialah jumlah koneksi yang terhubung. Jika *request* harus di terima dan diproses di *server* dan berapa banyak data overhead yang dibutuhkan. *Websocket* dan *Server Sent Event* diketahui memiliki performa lebih baik dari pada penelitian dengan metode yang lain dengan kondisi eksperimen. Perbedaan performa antara *Websocket* dan *Server Sent Event* ini menjelajahi bagian kecil dari kondisi nyata. Ini kenapa hasil dari penelitian ini akan

menentukan teknologi mana yang akan di jadikan standar teknologi saat ini.(Örnmyr, n.d.).

Berdasarkan penelitian Int. J. Communication, Network, and System Sciences oleh Q.Liu dan X. Sun. Dari penelitiannya menjelaskan bagaimana internet menjadi salah satu kebutuhan pokok manusia di era digital saat ini dari era Web 1.0 tentang informasi ke era Web 2.0 dimana terjadi banyak interaksi informasi didalamnya, dan di era sekarang informasi dengan banyaknya interaksi dan terkandung nilai jual, *e-commerce*, dan berita *online*.

Sekarang, komunikasi antara *browser client* dan *server* berbasis Hypertext Transfer Protocol (HTTP), Aplikasi dengan lapisan protocol bekerja dengan cara membuat kondisi *request* dan *response*. *Client* HTTP memberikan *request* terhadap *server* yang di trasmisikan melalui koneksi Transmission Control Protocol (TCP). Setelah menerima pesan *request* dari *client*, *server* meresponnya dengan cara memberikan pesan balik dan memutuskan hubungan. Dengan model ini, *server* tidak bisa memberikan data secara *real-time* kepada *client*. Lalu, teknologi seperti Flash, Comet, dan AJAX *Long Polling* telah menerapkan sistem *real-time* antara *server* dengan *client*. Namun teknologi itu tidak berhasil menerapkan komunikasi *real-time*, karena beberapa dari teknologi itu harus menginstall *plugin-plugin* tambahan di *browser* karena banyaknya beban terhadap *server*.

Akhirnya Q.Liu dan X. Sun mencari solusi terbaik untuk masalah ini. *Polling*, *Long Polling* dan *Streaming* HTTP adalah solusi utama yang digunakan oleh pengembang Web untuk mencapai komunikasi *real-time* antara *browser* dan *server* di

masa lalu. *Polling* sebuah pendekatan halaman penyegaran secara manual digantikan oleh program *auto running* adalah solusi paling awal untuk komunikasi *real-time* yang diterapkan pada *browser*. Implementasi yang mudah dan tidak ada kebutuhan tambahan untuk *client* dan *server* adalah keuntungan besar dari solusi ini. Namun, ada beberapa kekurangan yang jelas dalam solusi ini, sangat sulit untuk mengetahui frekuensi pemutakhiran data, sehingga *browser* tidak dapat memperoleh data terakhir pada waktunya. Selain itu, jika tidak ada pembaruan data yang terjadi selama periode waktu tertentu, permintaan sering *browser* akan menghasilkan lalu lintas jaringan yang tidak perlu dan menyebabkan beban *server* yang tidak perlu.

Untuk membuat *server* berkomunikasi dengan *browser* apapun Waktu, pengembang Web merancang mekanisme kunjungan baru yang disebut *Polling* atau komet panjang, yang memungkinkan *server* untuk menyimpan permintaan baru dari peramban selama beberapa periode alih-alih mengirimkan tanggapan dengan segera. Jika *update* data terjadi pada periode ini, *server* akan merespons *browser* dengan data kedatangan baru, dan *browser* akan melakukan permintaan lain saat menerima respon, Dengan mekanisme ini, *browser* bisa mendapatkan data terbaru dari sisi *server* pada waktunya. Namun, jika sejumlah besar *concurrency* terjadi, memori *server* dan kapasitas komputasi akan sangat banyak dilakukan dengan mempertahankan koneksi HTTP yang hidup.

Pengembang juga mencoba "HTTP Streaming" kunjungi mekanisme. Perbedaan utamanya adalah *server* tidak akan pernah menutup koneksi yang disponsori oleh *browser*, memutuskan akan menggunakan koneksi ini untuk

mengirim pesan kapan saja Dalam kasus ini, karena *server* tidak akan menandakan selesainya koneksi, respon dari *server* mungkin akan disangga oleh *firewall* dan *server* proxy di jaringan, menyebabkan beberapa kesalahan terjadi saat *browser* menerima data.

Komunikasi antara *client* dan *server* biasanya didasarkan pada koneksi HTTP yang membutuhkan *header* yang sesuai dengan permintaan klien dan respon *server*, sesuai definisi protokol HTTP, header ini berisi beberapa informasi pengendalian transmisi seperti tipe *pro-tocol*, versi protokol, jenis *browser*, bahasa transmisi, tipe pengkodean, dari waktu ke waktu, Cookie dan Sesi. Di bawah bantuan perangkat lunak seperti Firebug dan Turning on Live HTTP *Header, header* permintaan dan tanggapan dapat diamati dengan jelas. Contoh dari satu *header* permintaan dan tanggapan didefinisikan sebagai berikut:

From client (browser) to server:

GET /*Long-Polling* HTTP/1.1

Host: www.kaazing.com

User-Agent: Mozilla/5.0 (X11; U; Linux x86_64;

en-US; rv:1.9) Gecko/2008061017 Firefox/3.0

Accept:

text/html,application/xhtml+xml,application/xml;q = 0.9, */*; q = 0.8 Accept-

Language: en-us,en;q = 0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q = 0.7,*;q = 0.7

Keep-Alive: 300

Connection: keep-alive

Cache-Control: max-age = 0 Referer: http://www.example.com/

***From server to client* (*browser*):**

Date: Tue, 16 Aug 2008 00:00:00 GMT

Server: Apache/2.2.9 (Unix) Content-Type: text/plain

Content-Length: 12

Hello world

Ditunjukkan oleh dua *header* di atas, selain dari data "*Hello World*", sebagian besar data di header ini tidak berguna bagi pengguna akhir selama interaksi antara klien dan *server* ini, membiarkan *Cookie* and *Session* (informasi yang terdapat dalam dua item ini biasanya lebih dari sekedar informasi kontrol di header di sebagian besar situs web), Selanjutnya, jenis header ini akan disertakan dalam setiap interaksi. Jadi, itu harus menghabiskan banyak *bandwidth*, menghasilkan sejumlah besar lalu lintas jaringan jika menggunakan solusi pemungutan suara dan Komet. Selain itu, membangun dan menganalisa header akan memakan waktu yang digunakan untuk memproses permintaan dan tanggapan, dan menyebabkan tingkat latensi tertentu. Kekurangan *Polling* dan komet ini mengindikasikan bahwa kedua teknik ini harus digantikan oleh teknologi komunikasi *real-time* lainnya di masa depan. Mari kita beralih ke koneksi Socket Web.

Websocket menggunakan mekanisme HTTP *Upgrade* kelas ke protokol *Websocket.* Mekanisme goyangan tangan Web Socket kompatibel dengan HTTP. Oleh karena itu, *server* HTTP dapat berbagi *port* HTTP dan HTTPS *default* (80 dan 443) dengan *server Websocket*. Untuk membuat koneksi *Websocket* yang baru, protokol HTTP akan diupgrade ke protokol *Websocket* selama guncangan awal antara *client* dan *server*. Setelah koneksi dibuat, *Websocket* akan ditransmisikan bolak-balik antara *client* dan *server* berdasarkan model *full-duplex*. *Header handshake* awal diberikan seperti di bawah ini:

From *client* (*browser*) to *server*:

GET /text HTTP/1.1

Upgrade: *Websocket*

Connection: Upgrade

Host: www.*Websocket*.org

From *server* to *client* (*browser*):

HTTP/1.1 101 *Websocket* Protocol Handshake

Upgrade: *Websocket*

Connection: Upgrade

"Hello world" Ini jelas menunjukkan bahwa informasi kontrol termasuk-ed di header koneksi *Websocket* jauh lebih sedikit dari pada header koneksi HTTP. Di sisi lain, *Cookie* and *Session* tidak diperbolehkan dalam header koneksi *Websocket* sesuai dengan spesifikasi protokol *Websocket* dan yang pertama dan terutama, begitu koneksi terjalin dengan sukses, klien dapat berkomunikasi dengan *server* secara gratis,

dan hanya dua bit informasi kontrol dilampirkan pada data pengguna akhir yang dibutuhkan yang dikodekan oleh UTF-8, satu bit adalah "\ x00" pada awalnya, bit lainnya adalah "\ xFF" lokasi di akhir. Ini definisi *Websocket* membuat penurunan besar pada *bandwidth* dan waktu yang dikonsumsi oleh *header* pengolahan dalam koneksi Socket Web, kemudian menyebabkan lalu lintas jaringan kurang dan latensi yang lebih rendah. Inilah alasan pasti mengapa *Websocket* lebih cocok daripada *Polling* dan Comet untuk komunikasi *real-time* berbasis web.

Dari sisi keamanan, baik *Websocket* protocol dan protokol HTTP dapat mewujudkan transmisi yang aman. Wss dan https adalah transmisi transmisi aman terpisah mereka. Jadi, *Websocket* dianggap sebagai teknologi ideal untuk komunikasi *real-time* mengenai aspek lalu lintas, latency dan keamanan jaringan.

Hasil dari penelitian ini membuat Q.Liu dan X. Sun menyadari bahwa Efisiensi adalah isu utama dari *real-time* data trans- mission; Ini juga merupakan standar penting untuk mengevaluasi apakah sebuah protokol cocok untuk transmisi data *real-time*. Sebuah tes telah dilakukan untuk memantau kinerja *Websocket* dalam transmisi asinkron. Pengujian dibagi menjadi dua bagian, bagian pertama adalah memilah tabel yang berisi lima kolom dan tiga baris data di halaman phpMyAdmin berdasarkan permintaan HTTP, dan bagian lainnya adalah memilah tabel ukuran yang sama di halaman terpisah berdasarkan *Websocket*. komunikasi.

Selama proses pengujian keseluruhan, gunakan Google Chrome 5 sebagai *browser* klien, dan perangkat lunak Wireshark Network Protocol Analyzer sebagai alat monitor untuk melihat perubahan paket data dan *bit stream*. Akhirnya, dapatkan hasil

berikut (Gambar 2.3) koneksi Socket sepuluh kali lebih efisien daripada koneksi HTTP. Di sisi lain, lihat uji Peter Lubbers dan Frank Greco tentang perbandingan efisiensi antara *Polling* Ajax dan *Websocket*, dapat disimpulkan bahwa kinerja *Websocket* jauh lebih baik daripada HTTP dalam hal lalu lintas dan penundaan jaringan, terutama dalam jumlah besar kasus konkurensi. Angka.

	Number of packets		Number of bits		Time (second)	
	HTTP	Web Socket	HTTP	Web Socket	HTTP	Web Socket
Client to server	83	5	33,662	372		
Server to client	77	8	45,600	7456		
Total	160	13	79,262	7828	~2.5	~0.25

Gambar 2.3 perbandingan penelitian serupa

Kesimpulan dari penelitian ini ialah Transmisi data *real-time* akan menjadi tren yang tak terelakkan untuk sistem informasi berbasis web. *Websocket* dianggap sebagai generasi berikutnya dari Ajax yang akan banyak digunakan di Internet. Saat ini, *browser* IE8 yang paling populer dan versi yang lebih rendah masih belum mendukung Web Socket. Namun, Kaazing Company telah mengembangkan sebuah *gateway* cerdas yang dapat mengkonversi *Polling* Ajax dan Comet yang digunakan pada *browser* versi yang lebih rendah ke koneksi instan *Websocket*. Protokol *Websocket* dan *Websocket* API masih diperbarui. Mungkin, *Websocket* akan menjadi solusi sempurna untuk masalah "C10K" di masa depan (Liu & Sun, 2012).

2.2 Landasan Teori

2.1.1 Definisi Industri 4.0

Produksi industri saat ini didorong oleh persaingan global dan kebutuhan akan adaptasi produksi yang cepat dengan permintaan pasar yang senantiasa berubah. Persyaratan ini hanya dapat dipenuhi oleh kemajuan radikal dalam teknologi manufaktur saat ini. Industri 4.0 adalah pendekatan yang menjanjikan berdasarkan integrasi proses bisnis dan manufakturing, serta integrasi semua pelaku dalam rantai nilai (pemasok dan pelanggan) perusahaan. Aspek teknis dari persyaratan ini ditangani oleh penerapan konsep umum Sistem Cahaya Cyber (CPS) dan *Internet of Things* (IoT) ke sistem produksi industri. Sistem eksekusi Industri 4.0 oleh karena itu didasarkan pada koneksi blok bangunan CPS.

Blok-blok ini merupakan sistem tertanam dengan kontrol terdesentralisasi dan hubungan tingkat lanjut yang mengumpulkan dan menukar informasi *real-time* dengan tujuan untuk mengidentifikasi, menemukan, melacak, memantau dan mengoptimalkan proses produksi. Lebih jauh lagi, dukungan perangkat lunak yang ekstensif berdasarkan pada versi terdesentralisasi dan adaptasi dari Sistem Pelaksanaan Manufaktur (MES) dan *Enterprise Resource Planning* (ERP) diperlukan untuk integrasi proses manufaktur dan bisnis. Aspek penting ketiga adalah menangani sejumlah besar data yang dikumpulkan dari proses, mesin dan produk. Biasanya data disimpan dalam penyimpanan *cloud*.

Data ini memerlukan analisis ekstensif yang mengarah pada data 'mentah' ke informasi yang berguna dan, akhirnya sampai pada tindakan nyata yang mendukung proses produksi industri yang adaptif dan terus menerus. Karena pentingnya transisi ini untuk posisi suatu negara secara global pasar.

Beberapa inisiatif yang dipimpin oleh pemerintah diperkenalkan di seluruh dunia untuk mendukung transisi tersebut. Industri 4.0, sebagai inisiatif dan inspirasi pertama untuk inisiatif lainnya, berasal dari Jerman dan akan dibahas secara rinci dalam makalah ini. Konsep serupa yang diprakarsai di negara lain segera dipresentasikan di benua ini. Konsep Internet Industri telah berkembang di Amerika Utara oleh

General Electric pada akhir 2012. Hal ini dilihat sebagai integrasi ketat dunia fisik dan digital yang menggabungkan analisis data besar dengan *Internet of Things*. Konsep ini mengasumsikan area aplikasi yang jauh lebih luas seperti Industri 4.0 dan mencakup pembangkit tenaga listrik dan distribusi, perawatan kesehatan, manufaktur, sektor publik, transportasi dan pertambangan. Dalam konsorsium Internet Industri yang didirikan oleh *General Electric* dan beberapa perusahaan lain, diperkirakan bahwa 46% ekonomi global dapat memperoleh keuntungan dari Industrial Internet.

Di Prancis, konsep 'Industrie du futur' diperkenalkan sebagai inti masa depan Kebijakan industri Perancis Hal ini didasarkan pada kerjasama industri dan sains dan dibangun di atas lima pilar:

(1) teknologi terdepan termasuk manufaktur aditif, pabrik virtual, IoT, dan augmented reality,
(2) mendukung perusahaan Perancis, terutama yang kecil sampai menengah, untuk menyesuaikan untuk teknologi baru,
(3) pelatihan karyawan yang ekstensif,
(4) memperkuat kerja sama internasional seputar standar industri dan
(5) promosi industri Prancis di masa depan. Inisiatif serupa lainnya 'Made in China 2025' diperkenalkan pada tahun 2015.

Dilanjutkan oleh Kementerian Industri dan Teknologi Informasi China bekerja sama dengan banyak ahli dari Akademi Teknik China. Tujuan utama dari inisiatif ini adalah untuk secara komprehensif meng-upgrade industri China dengan menarik inspirasi langsung dari konsep Industry 4.0 Jerman dan menyesuaikannya dengan kebutuhan China. Manufaktur yang berubah harus didorong inovasi. Juga elemen lain seperti pembangunan berkelanjutan dan energi hijau dipertimbangkan. Sepuluh sektor prioritas diidentifikasi mulai dari teknologi informasi,

robotika dan mesin alat otomatis. Tujuan jangka panjangnya adalah untuk mereformasi industri manufaktur China, untuk beralih dari tingginya jumlah produk berbiaya rendah ke produk berkualitas tinggi dan untuk mengambil alih dominasi Jepang dan Jepang di bidang manufaktur sampai tahun 2035, untuk berkembang ke dunia industri. adidaya sampai tahun 2049.

Makalah ini akan berfokus pada konsep Industri 4.0 yang diperkenalkan oleh pemerintah Jerman yang bertujuan pada sistem produksi industri. Latar belakang

konsep, rencana pengembangan dan keadaan saat ini akan dibahas. Beberapa masalah latar belakang teknologi perangkat lunak, yang mencerminkan aspek penting dari konsep Industri 4.0, akan hadir.

Makalah ini disusun sebagai berikut. Bagian kedua menyajikan gagasan inti Industry 4.0, asal usul, tujuan dan elemennya serta sistem produksi Industri 4.0 (pabrik cerdas). Juga dukungan IT / *software* ditangani. Pada bagian ketiga Model Arsitektur Referential RAMI 4.0 yang menjadi dasar kegiatan standardisasi dijelaskan. Pada bagian keempat kesiapan perusahaan untuk Industri 4.0 dibahas dan contoh konkret sebuah perusahaan yang telah mengadopsi sebagian besar elemen Industri 4.0 disajikan. Pada bagian akhir kesimpulan ditarik dan topik umum dibahas (Rojko, 2017).

2.1.2 Definisi Internet Of Things

Dengan kemajuan terus menerus dalam teknologi sebuah inovasi potensial, IoT turun dari jalan yang berkembang sebagai jaringan komputasi global di mana-mana dimana setiap orang dan segala sesuatu akan terhubung ke Internet. IoT terus berkembang dan merupakan topik penelitian yang populer dimana peluangnya tidak terbatas. Imajinasi tidak terbatas yang menempatkannya di ambang membentuk kembali bentuk internet saat ini menjadi versi yang dimodifikasi dan terintegrasi. Jumlah perangkat yang memanfaatkan layanan internet meningkat setiap hari dan setelah semuanya terhubung dengan kabel atau nirkabel akan memberi sumber informasi yang hebat di ujung jari kita. Konsep untuk memungkinkan interaksi antara mesin cerdas adalah teknologi terdepan namun teknologi yang menyusun IoT bukanlah

sesuatu yang baru bagi kita. IoT, seperti yang bisa Anda tebak namanya, adalah pendekatan data konvergen yang diperoleh dari berbagai jenis hal ke *platform* virtual pada infrastruktur internet yang ada.

Konsep IoT berasal dari tahun 1982 ketika sebuah mesin coke yang dimodifikasi terhubung ke Internet yang dapat melaporkan minuman yang terkandung dan apakah minuman itu dingin. Kemudian, pada tahun 1991, visi kontemporer tentang IoT dalam bentuk di mana-mana komputasi pertama kali diberikan oleh Mark Weiser. Namun pada tahun 1999, Bill Joy memberi petunjuk tentang komunikasi Device to Device dalam taksonomi internetnya. Pada tahun yang sama, Kevin Ashton mengajukan istilah "*Internet of Things*" untuk menggambarkan sistem perangkat yang saling terhubung.

Ide dasar dari IoT adalah untuk memungkinkan pertukaran informasi berguna secara otomasi antara perangkat dunia nyata yang unik yang dapat dikenali secara berbeda di sekitar kita, didorong oleh teknologi terdepan seperti *Radio Frequent Identification* (RFID) dan *Wireless Sensor Networks* (WSNs) yang dirasakan oleh sensor dan diproses lebih lanjut untuk pengambilan keputusan, berdasarkan mana tindakan otomatis dilakukan.(Farooq & Waseem, 2015)

2.1.3 *Cloud Computing*

Dengan jutaan perangkat yang diperkirakan akan datang pada tahun 2020, *Cloud* tersebut tampaknya merupakan satu-satunya teknologi yang dapat menganalisis dan menyimpan semua data secara efektif. Ini adalah teknologi komputasi cerdas di mana jumlah *server* berkumpul di satu *platform Cloud* untuk memungkinkan berbagi

sumber daya antara satu sama lain yang dapat diakses di kapan saja dan di mana saja. *Cloud computing* adalah bagian terpenting dari IoT, yang tidak hanya menyatu dengan *server* tetapi juga memproses peningkatan kekuatan pemrosesan dan menganalisis informasi berguna yang diperoleh dari sensor dan bahkan menyediakan kapasitas penyimpanan yang baik. Tapi ini hanyalah awal dari melepaskan potensi sebenarnya dari teknologi ini.

Cloud computing yang dihubungkan dengan objek cerdas yang menggunakan jutaan sensor berpotensi sangat bermanfaat dan dapat membantu IoT untuk pengembangan skala sangat besar sehingga pencarian ulang dilakukan karena IoT akan sangat bergantung pada *Cloud Computing*. (Farooq & Waseem, 2015).

Gambar 2.4 *Cloud Computing*

2.1.4 Transmisi Data

1) HTTP *Long Polling*

HTTP *Long Polling* adalah upaya untuk meminimalkan latensi dan penggunaan sumber daya pemrosesan dan jaringan dibandingkan dengan pemungutan suara XHR dengan hanya merespons permintaan HTTP begitu sebuah peristiwa, status, atau batas

waktu terjadi. Saat menerima tanggapan, klien akan segera mengirim permintaan baru atau setelah penundaan dalam periode latensi yang dapat diterima. Perbedaan utama dari *Polling* XHR adalah ketika menggunakan HTTP *Long Polling*, tidak perlu klien terus mengirimkan permintaan sering untuk mengetahui apakah *server* memiliki update baru. Sebuah *Long* Polling tunggal dapat dibagi menjadi empat bagian (Loreto, Saint-Andre, Salsano, & Wilkins, 2011) :

1. Klien mengajukan permintaan awal dan kemudian menunggu tanggapan.
2. *Server* menolak responsnya sampai ada pembaruan atau sampai status atau batas waktu telah terjadi.
3. Bila pembaruan tersedia, *server* akan mengirimkan tanggapan lengkap kepada klien.
4. Klien biasanya mengirimkan permintaan jajak pendapat baru, segera setelah menerima tanggapan atau setelah jeda untuk mengizinkan periode latensi yang dapat diterima

Karena timeout akan memaksa siklus permintaan baru tanpa informasibermanfaat yang dipertukarkan, maka diinginkan untuk menjaga nilai timeout setinggi mungkin untuk meminimalkan jumlah timeout. Timeout dapat terjadi baik oleh *browser* klien yang menghentikan koneksi karena tidak ada tanggapan yang masuk dari *server*, namun lebih umum terjadi pada *server* web atau perantara (seperti proxy) untuk memulai batas waktu dengan mengirimkan respons HTTP 408 atau 504 karena mereka biasanya memiliki pengaturan batas waktu default yang lebih rendah. Sebuah jajak pendapat yang panjang akan tetap hidup umumnya sampai 120 detik tapi

30 detik adalah nilai yang lebih aman untuk menghindari *server* web atau perantara untuk membatasi *Long Polling* (Loreto et al 2011).

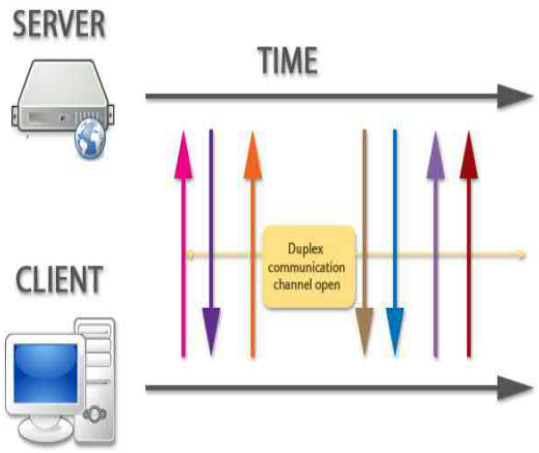
Ini berarti bahwa paling baik akan ada satu respons respons HTTP yang lengkap per *server* update. Jika frekuensi pesan *server* kurang dari pengaturan batas waktu lingkungan, *Polling* yang panjang akan menjadi sedikit kurang efisien karena batas waktu yang "tidak perlu" dan pengiriman ulang permintaan.

Teknologi ini dalam banyak hal seperti *Polling* XHR. Alih-alih klien mengirim misalnya sepuluh permintaan dimana hanya yang terakhir berisi informasi bermanfaat untuk klien, klien mengirimkan satu permintaandan *server* tidak meresponsnya sebelum ada sesuatu yang baru untuk klien. Hal ini menyebabkan lebih sedikit pesan dikirim antara *server* dan klien dibandingkan dengan *Polling* XHR tergantung seberapa sering *server* memiliki data baru untuk klien dan frekuensi pemungutan suara. Jika frekuensi *Polling* XHR sama dengan frekuensi pesan *server*, *Polling* XHR dan *Long Polling* harus secara teori memiliki dampak kinerja yang hampir sama.(Örnmyr, n.d.).

2) *Websocket*

Websocket adalah protokol yang memungkinkan komunikasi dupleks penuh antara klien dan web *server* Protokol ini bekerja melalui TCP (seperti HTTP) dan dibuat sebagai alternatif untuk *Polling* berulang (Fette & Melnikov, 2011). Sesi komunikasi interaktif terbentuk antara klien dan *server* web - saat terbuka, baik *server* dan klien dapat mengirim pesan satu sama lain tanpa bergantung pada struktur permintaan-respons. Lihat gambar 3 untuk ilustrasi bagaimana *Websocket* bekerja. Karena *Websocket* bekerja melalui koneksi TCP tunggal yang terus menerus, hal itu

dapat dimanfaatkan untuk secara efisien dan efektif memproses arus data antara klien dan *server* dengan jumlah overhead minimal dan sambil memberikan skalabilitas tingkat tinggi (Zhao, Xia, & Le, 2013).



Gambar 2.5 Ilustrasi *Websocket*

Websocket [RFC6455] mencakup protokol JS API dan IETF yang ditetapkan oleh W3C (World Wide Web Consortium). API *Websocket* belum distandarisasi. Meski begitu, banyak web *browser* memasukkannya ke dalam bentuk akhirnya. Ini adalah protokol jaringan yang berjalan pada TCP (Transmission Control Protocol). Ini dirancang untuk dijalankan di *browser* web dan *server* web, namun dapat digunakan untuk tujuan lain juga.

Meskipun independen dari protokol HTTP, ia memiliki kesamaan dalam hal proses kerja sama yang diperlukan untuk membuat koneksi. *Server Websocket* menggunakan port yang sama (TCP 80 atau 443) dengan HTTP. *Websocket* berjalan di atas satu koneksi TCP, dan memiliki kesamaan dengan metode koneksi TCP. Pertama, klien membuat permintaan HTTP ke *server*. Permintaan ini memiliki informasi header

"Upgrade *Websocket*". Ini berarti permintaan HTTP adalah permintaan jabat tangan dengan *Websocket*. Kemudian, *server* menerima permintaan HTTP, menganalisis informasi yang terdapat dalam permintaan, dan menghasilkan kunci keamanan untuk klien. Sambungan *Websocket* didirikan pada saat ini, dan sekarang kedua belah pihak (*server* dan klien) dapat mengirimkan data satu sama lain melalui koneksi ini. Setelah koneksi dibuat, nomor sesi akan dikirim ke *server*. Nomor ini hanya menampilkan Socket tertentu. *Server* menyimpan daftar nomor sesi untuk mengelola semua klien yang terhubung. Saat *browser* sisi klien ditutup, *server* menerima pesan peringatan dan kemudian *browser* dan sumber terkait dikeluarkan dari daftar klien *server*.

Alasan sebenarnya di balik *Websocket* adalah mengganti Comet dan HTTP *Polling* (dan HTTP *Long Polling*), dan memperbaiki status aplikasi *real-time* web biasa. Dibandingkan dengan metode komunikasi umum di HTTP, perbedaan utama *Websocket* adalah tidak mengikuti standar permintaan / tanggapan tradisional. *Websocket* membutuhkan beberapa sumber daya operasi untuk pengolahan data. *Websocket* menciptakan lalu lintas jaringan yang lebih rendah dalam kasus dimana potongan data kecil perlu ditransfer secara berkala. *Websocket* dapat mengurangi lalu lintas jaringan dengan rasio 500: 1, dibandingkan dengan metode HTTP biasa, Ini karena, ukuran setiap frame adalah 2 byte dalam koneksi *Websocket*. Ini melebihi 1 KB dalam *Long Polling*. Di *Websocket*, tidak ada koneksi TCP baru yang dibuat untuk mengirimkan setiap pesan HTTP. Oleh karena itu, tidak ada latency yang dibutuhkan untuk koneksi baru.

Websocket telah digunakan secara luas meningkat dari hari ke hari. Dalam sistem rumah pintar, perangkat IoT (Internet of Things) mengirimkan informasi yang dikumpulkan dari sensor mereka. *Websocket* dapat digunakan untuk transfer data pada perangkat ini. Selain itu, metode ini dapat digunakan dalam game multiplayer berbasis web dengan HTML5 agar dapat tersinkron, transmisi data pengukuran energi, dan pesan instan dan komunikasi kelompok berbasis web. (Comert, 2016).

2.1.5 *Javascript Object Notation* (JSON)

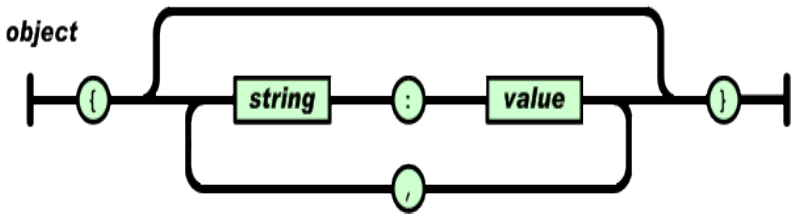
JSON adalah format pertukaran data berbasis teks terbuka (lihat RFC 4627). Seperti XML, ini bisa dibaca oleh manusia, *platform* independen, dan memiliki banyak implementasi. Data yang diformat sesuai dengan standar JSON yang ringan dan bisa diurai dengan implementasi JavaScript dengan mudah luar biasa, menjadikannya format pertukaran data ideal untuk aplikasi web Ajax. Karena ini terutama format data, JSON tidak terbatas hanya pada aplikasi web Ajax, dan dapat digunakan di hampir semua skenario di mana aplikasi perlu menukar atau menyimpan informasi terstruktur sebagai teks.

Ini (Atif Aziz, 2007).JSON (JavaScript Object Notation) adalah format data-interchange yang ringan. Mudah bagi manusia untuk membaca dan menulis. Mudah bagi mesin untuk mengurai dan menghasilkan. Hal ini didasarkan pada subset dari Bahasa Pemrograman JavaScript, Standard ECMA-262 3rd Edition - Desember 1999. JSON adalah format teks yang benar-benar bahasa independen namun menggunakan konvensi yang familiar bagi pemrogram keluarga C-bahasa, termasuk C, C ++, C #,

Java, JavaScript, Perl, Python, dan banyak lainnya. Properti ini menjadikan JSON sebagai bahasa pertukaran data yang ideal.

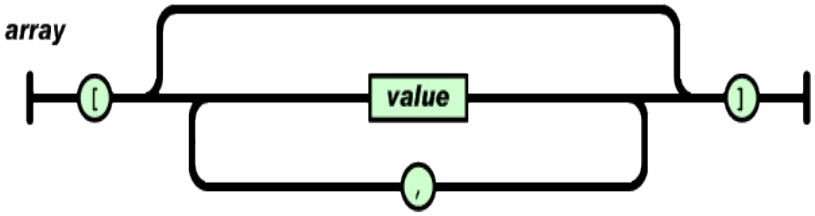
Kumpulan pasangan nama / nilai. Dalam berbagai bahasa, ini diwujudkan sebagai objek, catatan, struct, kamus, tabel hash, daftar kunci, atau array asosiatif. Daftar nilai yang terurut. Dalam kebanyakan bahasa, ini diwujudkan sebagai array, vektor, daftar, atau urutan.Ini adalah struktur data universal. Hampir semua bahasa pemrograman modern mendukung mereka dalam satu bentuk atau bentuk yang lain. Masuk akal bahwa format data yang bisa dipertukarkan dengan bahasa pemrograman juga didasarkan pada struktur ini.

Objek adalah kumpulan pasangan nama / nilai yang tidak berurutan. Objek dimulai dengan {(penjepit kiri) dan diakhiri dengan} (penjepit kanan). Setiap nama diikuti oleh: (titik dua) dan pasangan nama / nilai dipisahkan oleh, (koma).



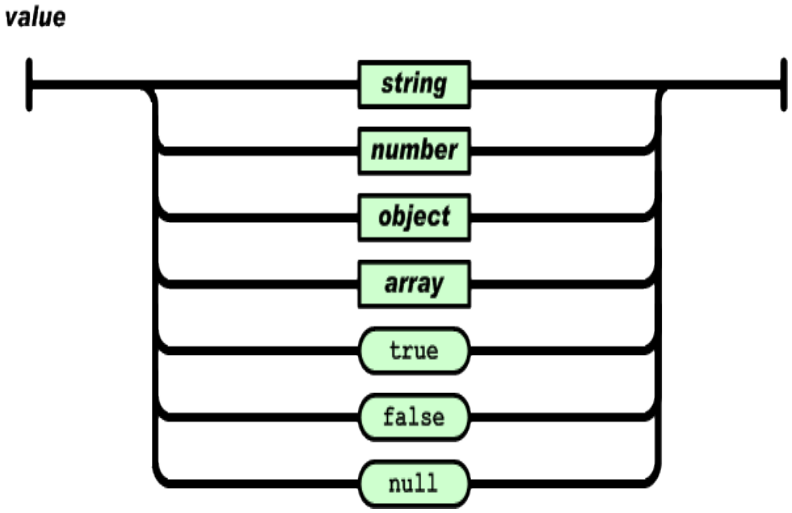
Gambar 2.6 Object array dalam JSON

Array adalah kumpulan nilai yang terurut. Sebuah array dimulai dengan [(braket kiri) dan diakhiri dengan] (braket kanan). Nilai dipisahkan oleh, (koma).



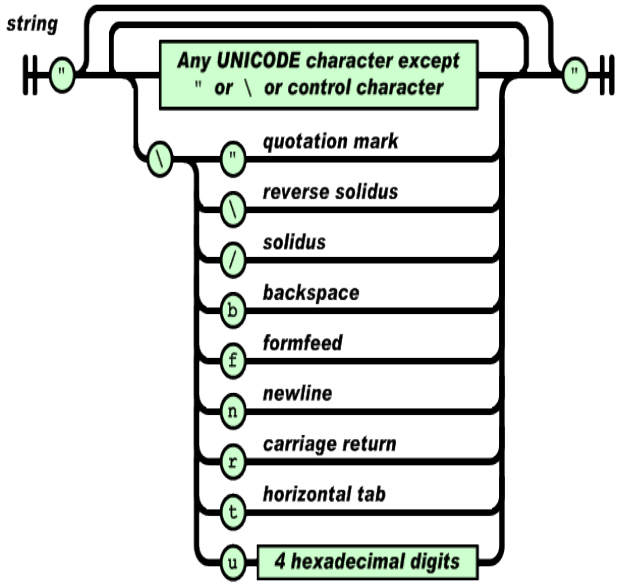
Gambar 2.7 Array dalam JSON

Nilai bisa berupa string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau objek atau array. Struktur ini bisa disarangkan.



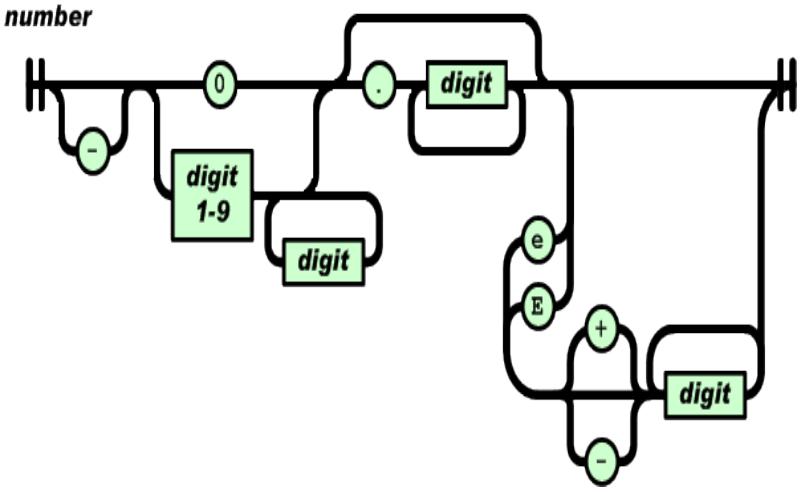
Gambar 2.8 Jenis data dalam JSON

String adalah urutan nol atau lebih karakter *Unicode*, dibungkus dengan tanda petik ganda, menggunakan pelepasan garis balik terbalik. Karakter diwakili sebagai string karakter tunggal. String sangat mirip dengan string C atau Java.



Gambar 2.9 Karakter Unicode dalam JSON

Angka sangat mirip dengan nomor C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.



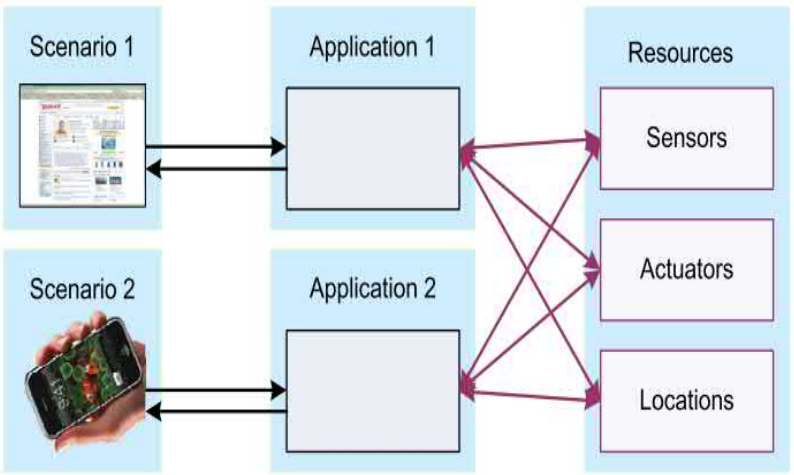
Gambar 2.10 Jenis data angka dalam JSON

Whitespace dapat disisipkan di antara sepasang token. Kecuali beberapa detail pengkodean, yang benar-benar menggambarkan bahasa.

2.1.6 Webservice RESTful API

Sistem web dapat dirancang dengan berbagai cara, REST berfokus untuk menghindari status aplikasi, pembuatan yakin bahwa konsep penting dari skenario aplikasi direpresentasikan sebagai sumber daya yang diidentifikasi *Uniform Resource Identifier* (URI), dan bahwa semua interaksi dengan klien melalui *server* berisi semua informasi keadaan yang diperlukan, sehingga *server* tidak perlu mempertahankan status sesi dengan klien. Arsitektur ini memiliki keunggulan besar dibandingkan aplikasi berbasis negara, yang biasanya bekerja dengan baik pada tahap awal mereka, namun kemudian mengalami masalah dengan pengujian, skalabilitas, dan integrasi dengan aplikasi lain. Gambar 2.10 menunjukkan skenario di mana desain sumber daya yang tenang memungkinkan beberapa aplikasi berinteraksi.

dengan sumber daya yang sama, tanpa perlu ada interaksi berbasis API antara aplikasi. Jika sumber daya diidentifikasi dan dikelola dengan cara yang memungkinkan interaksi paling penting dengan skenario model, maka arsitektur REST menghasilkan sistem pasangan *Long*gar dimana sumber daya dapat berinteraksi secara individual, tanpa memerlukan satu hambatan utama untuk menangani semua kemungkinan interaksi.



Gambar 2.11 Ilustrasi RESTFul API

Tentu saja, akhirnya pasti ada beberapa entitas pengendali yang misalnya memonitor lokasi, memang bisa untuk melaporkan semua sensor yang aktif di lokasi itu dan dapat diperiksa, dan menangani akses ke aktuator yang dapat digunakan di lokasi tersebut. Namun, gaya arsitektur REST memungkinkan decoupling terbaik dari skenario aplikasi spesifik di lingkungan ini, dan penanganan dasar sumber daya di dalamnya. Jadi, sementara satu aplikasi hanya mengizinkan akses ke sensor untuk tujuan pemantauan, aplikasi lain juga dapat memungkinkan akses ke aktuator untuk bertindak di lingkungan itu, dan kedua aplikasi dapat hidup berdampingan dengan menggunakan metode interaksi tanpa kewarganegaraan yang sama dengan sumber daya yang tersedia. (Wilde, 2007).

BAB V

KESIMPULAN

5.1. Kesimpulan

Berdasarkan hasil analisis dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa kecepatan respon dari metode pengiriman *Websocket* lebih cepat dibandingkan dengan *Long Polling*. Dalam HTTP *Long Polling*, server tidak segera mengirim respons kosong apabila tidak ada pesan baru yang tersedia untuk klien. Sehingga, server menahan permintaan sampai pesan baru tersedia atau hingga batas waktu habis. Berbeda dengan *Websocket*, server dapat megirimkan data tanpa harus ada permintaan dari pengguna, *Websocket* memungkinkan server mengirimkan data ke semua *client* yang terhubung melalui satu channel tanpa harus me-*request* setiap saat.

Penggunaan internet pada Webscoket juga lebih hemat dibandingkan dengan HTTP *Long Polling*. *Websocket* mempunyai rentang waktu rendah dalam kirim- terima data dan *Websocket* juga dapat mengurangi lalu lintas jaringan yang tidak penting. Menurut (Comert, 2016), Pengiriman data *Websocket* lebih efisien dibandingkan dengan HTTP *Long Polling*, begitu pula dalam hal bandwidth *Websocket* lebih efisien dibandingkan dengan HTTP *Long Polling*, dikarenakan HTTP *Long Polling* perlu mengirimkan permintaan koneksi ke server untuk memperbarui data.

Namun dalam penelitian yang dilakukan, penggunaan memori pada *Websocket* lebih besar yang dilakukan pada perangkat *Single Board Computer* Rasperry Pi 2 dengan NodeJS.

Berbeda dengan beban CPU dari masing-masing perangkat, beban CPU pada *Long Polling* sangat tidak stabil jika dibandingkan dengan *Websocket* karena request tersebut dilakukan berulang kali. *Long Polling* mengupdate perintah dengan cara permintaan berkala, sedangkan *Websocket* tidak hanya dapat mengirimkan *request* kepada server, tetapi juga menerima data dari server tanpa harus mengirimkan *request* terlebih dahulu. Hal ini sangat berpengaruh pada kecepatan respon, besar bandwidth dan beban CPU yang mana akan lebih di unggulkan oleh *Websocket*. Lalu dengan penelitian ini dapat disimpulkan bahwa penggunaan *Websocket* pada Internet of thing akan sangat berguna bagi dunia perindustrian dan rumah pintar yang mana dunia akan berkembang dengan penerapan mikrokontroller pada rumah pintar dan industri.

Sebagai contoh penerapan CCTV dengan *face recognition* pada masing-masing jalan raya yang berfungsi sebagai keamanan masyarakat, control perangkat jarak jauh seperti robot kecil maupun besar, system monitoring suhu dan kontroling lampu led (Hidayat et al., 2018) dan sistem pemantauan *real*-time berbasis Webscoket untuk bangunan cerdas yang memudahkan pengontrolan atau pemantauan aktivitas sensor secara real-time(Ma & Sun, 2013). Bahkan dapat dikatakan penggunaan internet akan semakin meningkat seiring berjalannya jaman. Maka dari itu jenis pengiriman data

akan sangat diperlukan untuk menanggulangi masalah efisiensi big data dan transmisi data yang *realtime*.

Dengan demikian, pengembangan IoT sebagai sistem cerdas dapat berjalan untuk interoperabilitas, energi berkelanjutan, privasi, dan keamanan. IoT telah menjadi tren tak terhindarkan dari perkembangan industri informasi, yang pasti akan membawa perubahan baru dalam kehidupan kita(Chen, et al, 2014).

5.2. Saran

Berdasarkan kesimpulan diatas, saran yang perlu dipertimbangkan ialah Pengujian dan penerapan Webscoket dapat langsung diimplementasikan dalam skala yang lebih besar agar dapat mengetahui kestabilan jaringan jika digunakan oleh banyak perangkat. Karena manfaat dan penerapannya didunia industri akan mencakup perangkat yang sangat banyak.

DAFTAR PUSTAKA

Ali, M., Alfonsus Vlaskamp, J. H., Eddin, N. N., Falconer, B., & Oram, C. (2013). Technical development and socioeconomic implications of the Raspberry Pi as a learning tool in developing countries. 2013 5th Computer Science and Electronic Engineering Conference (CEEC), 103–108. https://doi.org/10.1109/CEEC.2013.6659454

Atif Aziz, S. M. (2007). An Introduction to JavaScript Object Notation (JSON) in JavaScript and .NET. *Msdn.Microsoft.Com*, (February), 22. Retrieved from https://msdn.microsoft.com/en-us/library/bb299886.aspx

Boronczyk, T., Naramore, E., Gerner, J., Scouarnec, Y. Le, & Stolz, J. (2009). *Beginning PHP 6, Apache, MySQL 6 Web Development*. Retrieved from http://books.google.com/books?id=8CPSZACKJ_AC&pgis=1

Bradley, J. (2015). Json Web Token (JWT), 1–30.

Chen, S., Xu, H., Liu, D., Hu, B., & Wang, H. (2014). A vision of IoT: Applications, challenges, and opportunities with China Perspective. *IEEE Internet of Things Journal*, *1*(4), 349–359. https://doi.org/10.1109/JIOT.2014.2337336

Comert, O. (2016). The Comparison of New Web Communication Method WebSocket with Traditional Methods 2 . Web Communication Techniques, 1–8.

Fahrurrozi, I., & Azhari, S. N. (2012). Proses Pemodelan Software Dengan Metode Waterfall dan Extreme Programming: Studi Perbandingan. *Jurnal Online STMIK*

EL RAHMA, 1–10.

Farooq, M. U., & Waseem, M. (2015). A Review on Internet of Things (IoT). *International Journal of Computer Applications (0975 8887)*, *113*(1), 1–7. https://doi.org/10.5120/19787-1571

Hidayat, [2], Imam, L., Bhawiyuga, Adhitya, Siregar, & Andria, R. (2018). Implementasi Protokol Websocket Pada Perangkat Non IP Berbasis NRF24l01. *J-Ptiik*, *2*(6), 2058–2066.

Liu, Q., & Sun, X. (2012). Research of Web Real-Time Communication Based on Web Socket. *International Journal of Communications, Network and …*, *2012*(December), 797–801. https://doi.org/10.4236/ijcns.2012.512083

Loreto, S., Saint-Andre, P., Salsano, S., & Wilkins, G. (2011). Known Issues and Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP, 1–19. https://doi.org/10.17487/rfc6202

Ma, K., & Sun, R. (2013). Introducing websocket-based real-time monitoring system for remote intelligent buildings. *International Journal of Distributed Sensor Networks*, *2013*. https://doi.org/10.1155/2013/867693

Örnmyr, O. (n.d.). Performance comparison of XHR polling , Long polling , Server sent events and Websockets.

Paper, C., Jain, S., & Hcl, A. V. (2014). Raspberry Pi based interactive home automation system through E-mail Raspberry Pi based Interactive Home Automation System through E-mail. *Optimization, Reliabilty, and*, (November

2015), 277–280. https://doi.org/10.1109/ICROIT.2014.6798330

Rakhunde, S. M. (2014). Real Time Data Communication over Full Duplex Network Using Websocket. *IOSR Journal of Computer Science*, *5*, 15–19.

Roblek, V., Meško, M., & Krapež, A. (2016). A Complex View of Industry 4.0. *SAGE Open*, *6*(2), 215824401665398. https://doi.org/10.1177/2158244016653987

Rojko, A. (2017). Industry 4.0 Concept: Background and Overview, *11*(5), 77–90. https://doi.org/10.3991/ijim.v11i5.7072

Shuang, K., & Feng, K. (2013). Research on server push methods in web browser based instant messaging applications. *Journal of Software*, *8*(10), 2644–2651. https://doi.org/10.4304/jsw.8.10.2644-2651

Soehartono, I. (2003). Metodologi Penelitian, 20–37.

Sugiyono (2014). Metode Penelitian Kuantitatif, Kualitatif, dan R&D. Alfabeta CV Bandung, 979-8433-71-8.

Wilde, E. (2007). Putting Things to REST. *Transport*, *15*(November), 1–13. Retrieved from http://dret.net/netdret/publications#wil07n

Yu, H. R. (2015). Design and implementation of web based on Laravel framework. *Atlantis Press*, (Iccset 2014), 301–304. https://doi.org/10.2991/iccset-14.2015.66

Zanella, a, Bui, N., Castellani, a, Vangelista, L., & Zorzi, M. (2014). Internet of Things for Smart Cities. *IEEE Internet of Things Journal*, *1*(1), 22–32. https://doi.org/10.1109/JIOT.2014.2306328