PERANCANGAN APLIKASI PENGATURAN *WIRELESS SENSOR NETWORK* BERBASIS *WEB*

Nico Kurniawan [*)], Sukiswo, and Enda Wista Sinuraya

Jurusan Teknik Elektro, Universitas Diponegoro Semarang
Jl. Prof. Sudharto, SH, kampus UNDIP Tembalang, Semarang 50275, Indonesia

[*)] *Email: owahn@yahoo.com*

Abstrak

Wireless Sensor Network (WSN) adalah salah satu bidang riset yang sedang berkembang belakangan ini. Dengan meningkatnya penggunaan WSN, penyajian data hasil bacaan sensor dari setiap node menjadi bagian penting dalam WSN untuk keperluan analisa. Selain itu, penting untuk mendiagnosa keadaan suatu node seperti: mendapatkan nilai temperatur/suplai tegangan suatu node. Masalahnya adalah posisi WSN yang biasanya sulit dijangkau atau berada jauh dari operator. Untuk itu dibutuhkan sebuah sistem untuk mempermudah kegiatan tersebut. Penelitian ini membangun aplikasi berbasis web yang dapat menampilkan data dan mengendalikan/mengkonfigurasi suatu node pada WSN melalui internet. Sistem ini dibangun menggunakan bahasa pemrograman PHP, MySQL sebagai basisdata untuk menyimpan data sensor, dan Highcharts untuk menampilkan data tersebut dalam bentuk grafik garis. WSN berkomunikasi dengan sistem menggunakan HTTP melalui internet. Pengujian dilakukan dengan menggunakan sebuah WSN dengan tiap node menggunakan perangkat transreceiver XBee yang terdiri dari dua node dan satu koodinator, dan server dengan spesifikasi RAM 256MB dan CPU 200Mhz. Hasil pengujian menunjukan kapasitas sistem dalam menampilkan data dengan throughput 2 request/detik dan mengkonfigurasi suatu node.

Kata kunci: Wireless Sensor Network, Sistem Monitoring, Web

Abstract

Wireless Sensor Network (WSN) is one of the emerging field of research lately. With the increasing use of WSN, the presentation of data from sensor readings from each node becomes an important part in WSN for analysis. It is necessary to diagnose condition of a node such as: get its temperature, voltage supply, etc. The problem is the position of WSN are usually difficult to reach or are away from the operator. That requires a system to facilitate such activities. This study develops web-based application that can presents data and control/configure a node in the WSN via the internet. This system is built using the programming language PHP, MySQL as a database to store sensor data, and Highcharts to present those data in the form of a line graph. WSN uses HTTP to communicate with the system via the GSM network. Tests carried out using a WSN which consists of two nodes and one coordinator using XBee as its transreceiver module, and a server with 256MB of RAM and 200 MHz CPU. The test results show the capacity of the system in presenting data with throughput 2 request/second and configure a node.

Keywords: Wireless Sensor Network, Monitoring System, Web

1. Pendahuluan

Wireless Sensor Network (WSN) adalah salah satu bidang riset yang sedang berkembang belakangan ini. Dengan meningkatnya penggunaan WSN, data hasil bacaan sensor dari setiap *node* yang merupakan bagian penting dalam WSN, perlu untuk dikirim ke mesin yang mampu untuk melakukan visualisasi data tersebut untuk keperluan analisa lebih lanjut. Selain itu penting untuk mengetahui keadaan internal suatu *node* seperti: nilai temperatur, suplai tegangan; dan melakukan *reconfiguration* contohnya mengubah periode pengambilan data.

Terdapat beberapa standar dan teknologi untuk pengiriman data sensor dan melakukan konfigurasi, yaitu *Observations and Measurements (O&M)* [1] dan *XBee Remote AT Command*. O&M mendefinisikan format standar data observasi *node* berbasis XML. Fitur *Remote AT Command* pada XBee memungkinkan operator untuk melakukan konfigurasi *node* melalui program XCTU [2].

O&M menggunakan XML untuk mendefinisikan data observasi. XML adalah format data tekstual sehingga memerlukan *byte* ekstra untuk merepresentasikan suatu nilai. Hal tersebut kurang cocok untuk diimplementasikan untuk sensor *node* yang memiliki kapasitas memori yang kecil. XCTU membutuhkan operator berada pada lokasi WSN untuk melakukan pengaturan *node*. Hal tersebut akan menyulitkan operator jika lokasi WSN berada pada daerah yang sulit dijangkau atau berada jauh dari posisi operator.

Pada penelitian ini dirancang metode untuk format data observasi seminimal mungkin, dan metode untuk melakukan pengaturan *node* melalui *web*, sehingga operator tidak harus berada pada lokasi WSN. Data observasi yang diterima *web server* dapat diakses atau di-*streaming* oleh pengguna melalui *web*.

2. Metode

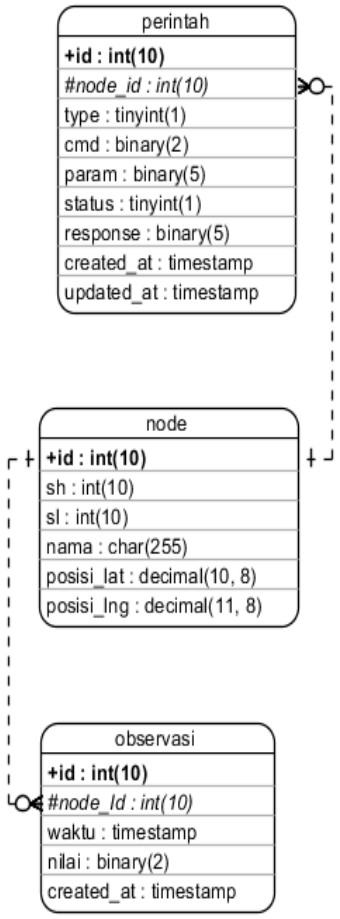
Sistem terdiri dari WSN dan *web server*. Fungsi utama sistem ini yaitu menyimpan data hasil observasi dari WSN, mengirim perintah ke *node* tertentu pada WSN, dan mengirim hasil dari perintah tersebut kembali ke *web server*. *Web server* berkomunikasi dengan WSN melalui Internet menggunakan HTTP. Data yang dikomunikasikan di-*encode*/*decode* terlebih dahulu. Perancangan ini meliputi tiga aspek yaitu: basisdata, struktur data, dan antarmuka grafis.

2.1.1 Basisdata

Perancangan basisdata bertujuan untuk mendapatkan rancangan yang efisien untuk menyimpan data observasi dan perintah. Tahap ini terdiri dari 4 langkah yaitu analisa kebutuhan, desain konseptual, desain logis, dan desain fisik. Analisa kebutuhan adalah proses menentukan kebutuhan atau kondisi yang diinginkan pada sistem ini. Desain konseptual adalah menentukan entitas-entitas yang ada pada sistem dan hubungannya berdasarkan hasil dari analisa kebutuhan. Desain logis adalah penentuan atribut setiap entitas. Desain fisik adalah penentuan tipe data pada atribut-atribut tersebut.

Hasil dari tahap perancangan ini adalah berupa *entity relation diagram* yang ditunjukkan pada Gambar 1.

Entitas perintah terdapat dua kolom tambahan yaitu *created_at* yang mengindikasikan kapan perintah pertama kali dimasukkan dan *updated_at* untuk dan kapan perintah terakhir di perbaharui. Entitas observasi terdapat kolom tambahan yaitu *created_at* yang mengindikasikan kapan data observasi diterima di *web server*.



Gambar 1 ERD Sistem

2.1.2 Struktur Data

Struktur data observasi terdiri dari sederetan N node. Setiap node dapat menampung hingga D data. Setiap data terdiri dari 4 byte timestamp dan V byte untuk menampung nilai observasi. Sebelum data observasi dikirim, data tersebut akan ditambahkan *header* untuk membantu *web server* mem-*parsing* data observasi. Contoh implementasi struktur data ini dapat dilihat pada Tabel 1. Besar memori yang harus dialokasikan koordinator WSN adalah sebesar:

$$(1 + N) + N\big(8 + D\,(4 + V)\big) \qquad (1)$$

Struktur data perintah terdiri dari sederetan N *node*. Setiap *node* dapat menampung hingga P perintah terdiri dari C *byte* nilai perintah, 1 *byte* id perintah, dan P *byte* parameter perintah. Setiap data terdiri dari 4 *byte timestamp* dan V *byte* untuk menampung nilai observasi. Sebelum data perintah dikirim, data tersebut akan ditambahkan *header* untuk membantu WSN mem-*parsing* data perintah. Contoh implementasi struktur data ini dapat dilihat pada

Tabel 2. Besar memori yang harus dialokasikan koordinator WSN adalah sebesar:

$$(1 + N) + N\big(8 + P\,(4 + C + M)\big) \qquad (2)$$

Data respons perintah adalah bagian dari struktur data perintah yang dapat yang ditambah data respons, oleh karena itu besar maksimum yang harus dialokasikan WSN adalah sebesar:

$$(1 + N) + \big(8 + P(4 + C + M + R)\big) \qquad (3)$$

Respons dikirim balik ke *web server* dalam bentuk sederetan pasangan dan id perintah. Contoh data yang akan dikirim ke *web server* dapat dilihat pada

Tabel 3.

Tabel 1 Contoh data observasi

Offset	*Nilai*	*Keterangan*
Header		
00	01	Jumlah *node*
01	03	Jumlah data *node* ke 1
***Node* Ke-1**		
02 - 05	13A20040	SH *node* ke-1
06 - 09	40AD1933	SL *node* ke-1
Data Ke-1		
10 – 13	554B6910	Waktu observasi data ke-1
14 – 15	01EA	Nilai observasi data ke-1
Data Ke-2		
16 – 19	554B6910	Waktu observasi data ke-2
20 – 21	01EA	Nilai observasi data ke-2
Data Ke-3		
22 – 25	554B6910	Waktu observasi data ke-3
26 – 27	01EA	Nilai observasi data ke-3

Tabel 2 Contoh data perintah

Offset	*Nilai*	*Keterangan*
Header		
00	02	Jumlah *node*
01	02	Jumlah perintah *node* ke 1
02	01	Jumlah perintah *node* ke 2
***Node* Ke-1**		
03 ~ 06	13A20040	SH *node* ke-1
07 ~ 10	40AD1933	SL *node* ke-1
Perintah Ke-1		
11	03	id perintah ke-1
12	17	*type* perintah ke-1
13 ~ 14	4442	Nilai perintah ke-1
15 ~ 19	0	*parameter* perintah ke-1
Perintah Ke-2		
20	04	id perintah ke-2
21	10	type perintah ke-2
22 ~ 23	0000	Nilai perintah ke-2
24 ~ 28	030000000	*parameter* perintah ke-2
***Node* Ke-2**		
29 ~ 32	13A20040	SH *node* ke-2
33 ~ 36	40AD1934	SL *node* ke-2
Perintah Ke-1		
37	05	id perintah ke-1
38	17	*type* perintah ke-1
39 ~ 40	2556	Nilai perintah ke-1
41 ~ 45	0	*parameter* perintah ke-1

Tabel 3 Contoh data respons perintah

Offset	*Nilai*	*Keterangan*
Respons Perintah Ke-1		
00	01	Task id ke-1
01 ~ 05	00000000	Respons perintah ke-1
Respons Perintah Ke-2		
06	02	Task id ke-2
07 ~ 11	0000000F	Respons perintah ke-2
Respons Perintah Ke-3		
12	03	Task id ke-3
13 ~ 17	0000003C	Respons perintah ke-3
Respons Perintah Ke-4		
18	04	Task id ke-4
19 ~ 23	00000002	Respons perintah ke-4
Respons Perintah Ke-5		
24	05	Task id ke-5
25 ~ 29	00000001	Respons perintah ke-5

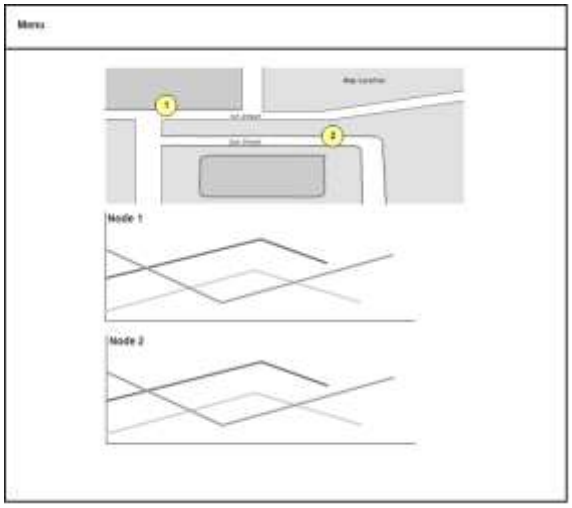
2.1.3 Encoding Data

WSN dan *web server* berkomunikasi melalui HTTP. Data sensor dan perintah diletakkan pada HTTP *message body*. Penelitian ini Base64 digunakan untuk memudahkan *parsing* data pada sisi WSN. Ukuran data yang telah di-*encoding* akan mengalami *overhead* sekitar 33%.

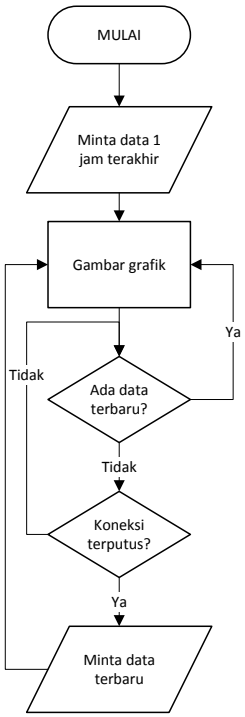
2.1.4 Antarmuka Grafis

Antarmuka grafis pada sistem ini terdiri dari *Monitoring* dan Pengaturan. *Monitoring* berisi data hasil observasi dari WSN yang ditampilkan dalam bentuk grafik, dan peta yang menunjukkan lokasi dari setiap *node*. Pengaturan *berisi* tampilan untuk mengirim perintah ke *node* tertentu dan log perintah. *Layout* antarmuka menggunakan *framework* CSS *Bootstrap*.

Data dari WSN ditampilkan pada halaman *web* dalam bentuk grafik garis menggunakan *Highcharts*. Saat pertama kali halaman dikunjungi, halaman tersebut membuka koneksi ke *web server* dan melakukan permintaan ke *web server* untuk mendapatkan data observasi 1 jam terakhir. Selanjutnya *web server* akan mengirim data observasi setiap 5 detik sekali ke *web browser*. Jika koneksi antara *web browser* dan *web server* terputus, *web browser* hanya akan meminta data observasi berdasarkan data terakhir yang sudah diterima, dengan ini data yang ada pada grafik akan tetap konsisten. Pengguna tidak perlu me-*refresh* halaman *monitoring* untuk melihat data terbaru. Diagram alir ditunjukkan pada
Gambar 3.



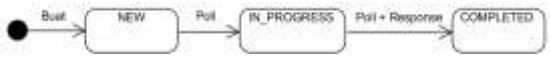
Gambar 2 Rancangan halaman *web monitoring*



Gambar 3 Diagram alir monitoring

Pengguna dapat mengirim perintah ke *node* tertentu melalui halaman *web* (

Gambar 5). Perintah yang baru dimasukkan pengguna berstatus NEW. Setelah WSN melakukan *polling*, perintah tersebut berstatus **IN_PROGRESS**. Setelah WSN mengirim respons, status dari perintah tersebut menjadi **COMPLETED**. State diagram perintah dapat dilihat pada
Gambar 4. *State diagram* perintah dapat dilihat pada.



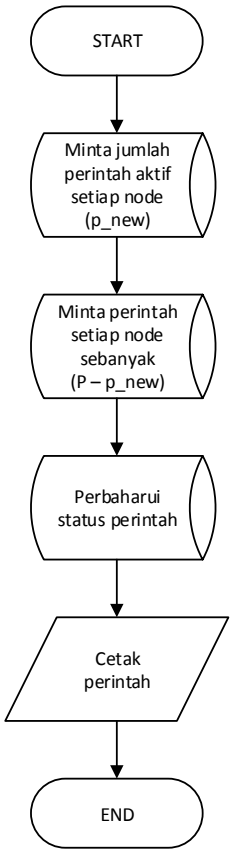
Gambar 4 *State diagram* perintah

Gambar 5 Rancangan halaman *web* Pengaturan

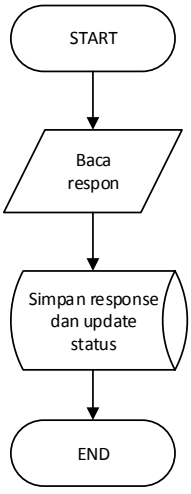
WSN menerima perintah dari *web server* dengan cara melakukan *polling* oleh koordinator setiap periode waktu yang ditentukan. Ketika *polling* dilakukan, *web server* mengirim *query* ke basisdata untuk meminta jumlah perintah (**p_active**) yang ditujukan untuk *node* yang memiliki jumlah perintah berstatus IN_PROGRESS dibawah **P** (jumlah maksimal perintah per *node*). Dari hasil *query* tersebut dapat diketahui jumlah slot perintah yang tersedia untuk setiap *node*. *Query* selanjutnya adalah meminta perintah untuk setiap *node* sebanyak slot yang tersedia yaitu **P – p_active**. Status setiap perintah yang didapat dari *query* tersebut diperbaharui menjadi **IN_PROGRESS**.

Setiap kali *web server* menerima respons sebuah perintah dari WSN, status perintah tersebut diperbaharui menjadi COMPLETED dan responsnya disimpan di basisdata. Diagram alir dari proses ini ditunjukkan pada Gambar 6.

Koordinator WSN melakukan proses *decoding* setelah data perintah diterima. Setelah itu koordinator mengirim perintah ke *node* tujuan. Id perintah digunakan untuk menentukan *FrameId* paket perintah. *FrameId* paket *request* pada XBee sama dengan *FrameId* paket *respon*-nya [3]. Paket yang datang ke koordinator dicek *FrameId*-nya, jika cocok dengan id salah satu perintah maka perbaharui status perintah tersebut dan simpan respons perintah tersebut.



Gambar 6 Diagram alir pengambilan perintah



Gambar 7 Diagram alir simpan respons perintah

3 Hasil dan Analisa

Pengujian ini menggunakan data *dummy* yang dihasilkan program *dummy-data* yang ditulis menggunakan bahasa pemrograman Python sebagai pengganti WSN nyata. Program ini mengirim data observasi secara periodik. Program ini juga dapat menerima perintah dari *web server* dan menanggapinya mirip dengan jaringan XBee. Pengujian ini membatasi jumlah perintah yang ada menjadi dua: perintah AT%V untuk mendapatkan nilai suplai tegangan pada *node* tertentu [4], dan perintah untuk mengubah periode pengambilan data suatu *node*. Nilai observasi yang dihasilkan program ini ini ditentukan secara acak dengan batas bawah dan batas atasnya. Informasi yang ditampilkan program *dummy-data* adalah: data observasi yang diterima dari *end node*, perintah yang dikirim dari koordinator, dan data yang dikirim ke *server* berserta tanggapannya.

3.1 Pengujian Menggunakan Data *Dummy*

Pengujian ini adalah pengujian *blackbox* yaitu perangkat lunak diuji kebenarannya berdasarkan keluaran dari masukan yang diberikan. Pengujian ini terdiri dari beberapa *test cases* yaitu: penerimaan data observasi, mengirim perintah AT%V, dan mengirim perintah untuk mengubah tingkah laku *node*. Dua *test case* terakhir merupakan bagian dari fitur pengaturan *node* dari *web server*. Pengujian ini terdapat dua *end node* dan satu koordinator. Properti dari setiap *node* ditunjukkan pada Tabel 4. **SH** adalah 4 *byte* pertama alamat *node*. **SL** adalah 4 *byte* terakhir alamat *node*.

Tabel 4 Properti node pengujian dengan data *dummy*

Properti	*Node 1*	*Node 2*
SH	0x0013A200	0x0013A200
SL	0x40AD1933	0x40AD1934
Nilai Sensor	10 – 200	50 – 100
Periode Data	5 detik	20 detik
Suplai Tegangan	3 V	5 V

3.1.1 Pengujian Monitoring

Pengujian ini untuk memastikan struktur data observasi yang telah dirancang dapat merepresentasikan data dari sensor dengan benar. Pengujian ini juga memastikan metode untuk mengubah struktur data tersebut menjadi SQL dapat dieksekusi untuk menyimpan data observasi ke basisdata pada *web server*.

Pertama-tama tabel observasi dikosongkan terlebih dahulu, lalu program *dummy-data* dijalankan hingga mengirim data ke *server* dan mendapatkan responsnya, lalu isi dari tabel observasi dicocokkan dengan keluaran yang dihasilkan program tersebut, terakhir keluaran yang dihasilkan dicocokkan juga dengan grafik garis yang ada pada antarmuka grafis *Monitoring*. Hasil pengujian ini menunjukkan bahwa grafik garis pada halaman *web* sesuai dengan data *dummy*.

3.1.2 Pengujian Pengaturan

Pengujian ini bertujuan untuk memastikan struktur data perintah dan metode untuk mengubah respons perintah dapat bekerja sesuai dengan harapan. Terdapat dua perintah dalam pengujian ini, yaitu mendapatkan suplai tegangan (AT%V) dan mengganti periode pengambilan

data. Pengiriman perintah dapat dilakukan pengguna melalui halaman Pengaturan.

Untuk menguji apakah perintah AT%V bekerja sesuai dengan harapan, dapat dicocokkan respons perintah dengan nilai tegangan *node* yang bersangkutan. Nilai tegangan setiap *node* telah ditetapkan sebelumnya pada Tabel 4. Berikut adalah formulir pada halaman Pengaturan *Node* yang telah diisi untuk pengujian perintah AT%V. Hasil pengujian ini menunjukkan respons perintah dapat diterima oleh *web server* dan nilainya sesuai dengan Tabel 4.

3.2 Pengujian *Throughput*

Pengujian *throughput* dilakukan menggunakan perangkat lunak *jMeter*. Sistem dipasang pada *server* dengan spesifikasi RAM 256MB dan CPU 200 Mhz dan terpasang sistem operasi Ubuntu 14.04, PHP 5.5, MySQL 5.5, dan Apache 2.2. Aplikasi *monitoring* melakukan *delay* 10 detik setiap kali mengirim data ke *web browser* dan menutup koneksi 20 detik setelah *request* pertama kali dikirim *web browser*, dan mengirim maksimal 60 data terbaru setiap *delay* sehingga aplikasi paling banyak mengirim 120 data setiap *request*.

Pengujian dilakukan dengan mensimulasikan 100 pengguna yang mengakses halaman *Monitoring* secara bersamaan. Pengujian dilakukan 10 kali dengan memvariasikan jumlah *node* yang di-*streaming*. Tabel 5 menunjukkan hasil pengujian.

Tabel 5 Hasil pengujian *throughput*

Node	*Err. (%)*	*Throughput*	*Waktu respons (ms)*		
			MAX	*MIN*	*AVG*
1	0	2,2	44327	20239	25989
2	28	3.1	30723	785	16845
3	0	2.1	45060	20392	26328
4	0	2.2	44470	20096	26047
5	1	2,1	45024	4965	26510
6	9	2,2	44479	4907	24564
7	2	2	47983	8543	30292
8	0	2,1	45892	20339	27816
9	6	2,1	45226	2620	25287
10	11	2,1	44704	6501	25794

Error yang terjadi selama pengujian adalah ketika MySQL tidak dapat mengalokasi memori. Eksekusi program dihentikan ketika terjadi *error* sehingga respons lebih cepat dikirim. Hal tersebut dapat mengurangi rata-rata waktu respons atau meningkatkan *throughput*.

Dari Tabel 5 dapat diketahui bahwa aplikasi *monitoring* dapat memproses 2 permintaan pengguna setiap detiknya. Nilai tersebut didapat dari rata-rata pengujian yang *error*-nya 0 persen. Setiap *request* menghasilkan *response* paling banyak 120 data observasi, berarti sistem ini mampu menghasilkan 240 data/detik. Selain itu dapat diketahui bahwa semakin banyak jumlah *node* yang di-*streaming*, semakin besar *delay* yang dihasilkan.

4 Kesimpulan

Beberapa kesimpulan yang dapat diambil dari hasil perancangan dan pengujian pada penelitian ini adalah :

1. Integrasi WSN dengan internet menggunakan skema *Front-end solution* dapat digunakan untuk WSN yang tidak dapat diakses publik.
2. Skema *encoding* Base64 dapat menghilangkan karakter non ASCII suatu data tetapi mengubah ukuran data tersebut menjadi sekitar 133% dari ukuran aslinya.
3. Aplikasi *monitoring* mampu melayani 2 permintaan per detik menggunakan komputer RAM 256MB dan CPU 200 Mhz.
4. Semakin banyak jumlah *node* yang di-*streaming*, semakin besar *delay* yang dihasilkan.
5. Jumlah *node* yang di-*streaming* tidak mempengaruhi besar *throughput* aplikasi.

Referensi

[1]. Open Geospatial Consortium, Inc., *Observations and Measurements (O&M)*, http://www.ogcnetwork.net/om, Juni 2015.

[2]. Digi International, *XCTU*, http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu, Juni 2015.

[3]. Faludi, R., *Building Wireless Sensor Network: A Practical Guide To The Zigbee Mesh Networking Protocol*, California, O'Reilly Media, 2011.

[4]. Digi International, *XBee Command Reference Tables*, http://examples.digi.com/wp-content/uploads/2012/07/XBee_ZB_ZigBee_AT_Commands.pdf, Juni 2015.