KINERJA KOMPUTASI TERDISTRIBUSI BERBASIS LAYANAN WEB MENGGUNAKAN METODE SOAP *SIMPLE OBJECT ACCESS PROTOCOL*

Yogiswara

Jurusan Teknologi Informasi, Politeknik Negeri Jember
E-mail: yogipoltek@gmail.com

ABSTRACT

Technology web services (Web Service) published to overcome the problems interoperabillitas between applications is a major obstacle in the process of data integration. This technology is a distributed computing technology that is widely used because of its simplicity and flexibility than previous technologies. There are three methods of data exchange which has been standardized in this technology, one of which is a SOAP (Simple Object Access Protocol) is a protocol for the exchange of information with a remote procedure call mechanism through HTTP protocol in the form of XML. These methods has been widely applied in many cases but still have not measured the performance of this method, especially in the execution of data. In this study, the method of exchange of SOAP tested its performance in a distributed environment with different server platforms. Performance measurement results show the method has the SOAP protocol that is easy to understand by the client for the service descriptions using the standard WSDL (web services description language), while a web server that has the best performance in executing SOAP is a web server NGINX

Kata Kunci:*Layanan Web, SOAP.*

PENDAHULUAN

Layanan web (web service) merupakan salah satu jenis layanan teknologi informasi yang berkembang pesat dibandingkan dengan teknologi lainnya. Layanan Web telah memberi dampak perubahan yang nyata pada pembuatan dan pengembangan aplikasi berbasis web. Hal ini terlihat jelas pada kemunculan sejumlah aplikasi berbasis web yang kaya akan isi serta kemampuan mengintegrasikan data yang cukup handal. Layanan Web memungkinkan sistem yang berbeda platform baik sistem operasi, arsitektur aplikasi dan bahasa pemrograman untuk berkomunikasi satu sama lain. Mekanisme yang digunakan dalam berkomunikasi menggunakan pesan dengan format XML (*Extensible Mark Up Language*). XML terbentuk dalam format teks yang memiliki standar dimana semua sistem dapat memahami format pesan tersebut. Sebagai contoh, dalam aplikasi pesan pribadi (*chatting, messenger*), teks yang diketik oleh pengguna di kemas dalam bentuk pesan dengan format XML dan masing-masing platform aplikasi menerjemahkan pesan tersebut sesuai protokol yang telah didesain oleh pengembang perangkat lunak, terdapat tiga macam metode dalam melakukan proses tersebut salah satunya adalah SOAP.

SOAP (*Simple Object Access Protocol)* telah banyak diimplementasikan dalam berbagai hal diantaranya untuk mengintegrasikan data antar server dalam sebuah organisasi [1],[2] dan digunakan untuk melakukan transaksi data antar aplikasi [3]. Isu utama dalam penerapan layanan web adalah masalah kualitas layanan. Standar kualitas pada layanan web terbagi dalam beberapa hal diantaranya adalah kemampuan layanan web dalam melayani permintaan klien

(*accessibility*), kemampuan layanan web untuk melaksanakan fungsinya pada interval waktu tertentu (*reliability*), dan kemampuan kinerja (*performance*) [4]. Kemampuan kinerja layanan web juga direpresentasikan dengan kecepatan layanan dalam menyelesaikan permintaan klien dimana Pengukuran tersebut dapat dilakukan berdasarkan *response time* yaitu lama waktu yang dibutuhkan untuk menyelesaikan permintaan, *throughput* yaitu jumlah permintaan yang dapat diselesaikan dalam periode tertentu, dan *latency* yang merupakan lama waktu yang diperlukan dari pengiriman permintaan oleh klien sampai menerima hasil permintaan tersebut dari server [5]. *Respon time dan throughput* lebih banyak dipengaruhi faktor kualitas perangkat keras dan jaringan sedangkan *latency* lebih mengarah pada pengukuran kinerja aplikasi perangkat lunak.

Berdasarkan hal tersebut maka penelitian ini bertujuan untuk menilai kinerja web service menggunakan metode SOAP dalam melakukan transaksi data berupa transaksi penambahan, perubahan, penghapusan dan pemilihan data

TINJAUAN PUSTAKA

2.1. Layanan Web

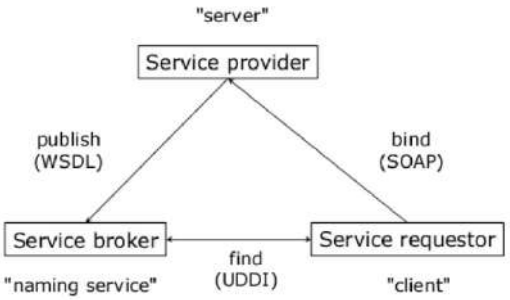
Layanan web (*Web Service*) adalah layanan yang tersedia melalui Internet yang menggunakan sistem pesan dengan standar XML dan tidak terikat pada satu sistem operasi atau bahasa pemrograman. Ada beberapa alternatif untuk menjalankan pesan XML yaitu dengan menggunakan *Remote Procedure Calls XML* (XML-RPC) atau SOAP atau bisa menggunakan HTTP GET/POST yang secara sistem akan leluasa melewatkan segala bentuk dokumen XML. Meskipun tidak wajib, layanan web juga memiliki dua sifat. Pertama Sebuah layanan web harus *self-describing*, *intinya* Jika mempublikasikan sebuah layanan web baru, sebaiknya juga harus menerbitkan antarmuka publik ke layanan. Minimal, layanan anda harus mencakup dokumentasi yang dapat terbaca manusia sehingga pengembang lain dapat lebih mudah mengintegrasikan layanan tersebut. Jika kita telah membuat layanan SOAP, idealnya anda juga harus membuat antarmuka publik yang ditulis dalam tata bahasa XML umum. Tata bahasa XML dapat digunakan untuk mengidentifikasi semua metode umum, metode argumen, dan nilai-nilai timbal baliknya. Kedua, sebuah layanan web harus *discoverable*. Jika anda membuat sebuah layanan web, harus ada mekanisme yang sederhana untuk mempublikasikan data tersebut dan harus ada mekanisme yang sederhana agar pihak yang berkepentingan dapat menemukan layanan tersebut. karakteristik layanan web sebagai berikut:

- Service yang mempertukarkan data dalam format XML message yang non-binary melalui jaringan menggunakan HTTP
- Bersifat terbuka, platform dan bahasa program independent (bisa diakses oleh aplikasi web, desktop ataupun mobile)
- Penyedia berupa aplikasi yang tidak memiliki antar muka web
- Menerapkan salah satu teknologi XML-RPC, SOAP atau REST
- Memiliki sifat-sifat layanan pada umumnya yaitu *interoperability, self-describing, discoverable, reusable*

Arsitektur layanan web terdiri dari *Service provider*, merupakan pemilik layanan web yang berfungsi menyediakan kumpulan operasi dari layanan web. *Service requestor*, merupakan aplikasi yang bertindak sebagai klien dari layanan web yang mencari dan memulai interaksi terhadap layanan yang disediakan. *Service Broker*, merupakan tempat dimana *service provider* mempublikasikan layanannya. Pada arsitektur layanan web, *Service broker* bersifat optional. Teknologi layanan web memungkinkan kita dapat menghubungkan berbagai jenis software yang memiliki platform dan sistem operasi yang berbeda.

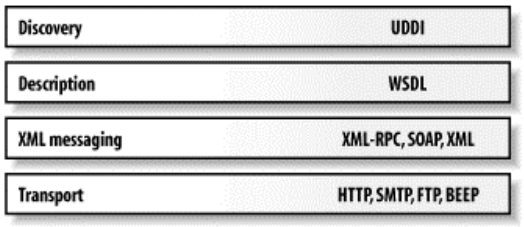
Selain itu berdasarkan lapisan protokolnya terdiri dari pertama, *XML messaging* yaitu layer yang bertanggung jawab untuk *encoding* pesan dalam format umum XML sehingga pesan dapat dipahami di kedua ujung. Sekarang di dalam layer ini termasuk di dalamnya

terdapat XML-RPC dan SOAP. Kedua, *service description* yaitu layer yang bertanggung jawab untuk mendeskripsikan layanan web tersebut dalam bentuk *public interface* menggunakan WSDL (*Web Service Description Language*) untuk dipublikasikan ke *service broker*. [6]



Gambar 1.Arsitektur Layanan Web

Ketiga adalah *service discovery* yang bertanggung jawab untuk mempublikasikan (mendaftarkan, menyimpan dan mengkategorikan) layanan ke dalam *service broker/registry* serta menyediakan fasilitas untuk pencarian layanan dan providernya. Saat ini, *service discovery* ditangani melalui Deskripsi Universal, Discovery, dan Integrasi (UDDI).



Gambar 2 lapisan layanan web

Secara umum mekanisme kerja layanan web terdiri dari dua aktivitas yaitu Aktivitas di sisi server sebagai berikut:

- Membuat fungsi utama/*core function*
- Membuat *service wrapper* berupa XML-RPC atau SOAP
- Membuat deskripsi layanan berupa WSDL atau instruksi integrasi XML-RPC (memuat semua method *public*, argumen dan return valuenya); ditambah dokumentasi yang *human readable*
- *Deploy* (rilis) *service*
- Daftarkan *service* tersebut melalui UDDI agar *discoverable*

Dan aktivitas di sisi klien adalah dengan langkah-langkah:

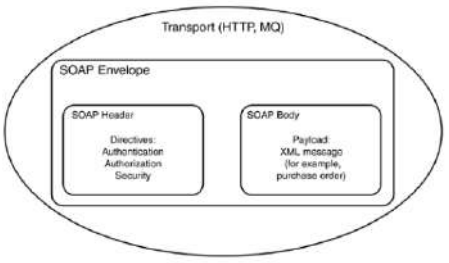
- Mencari service melalui UDDI
- Mengambil *service description file* berupa WSDL atau instruksi XML-RPC
- Membuat klien XML-RPC atau SOAP berupa fungsi lokal atau pesan XML untuk dikirim berdasarkan WSDLnya
- Memanggil remote service tersebut

2.2. *Simple Object Access Protocol*

merupakan suatu protokol pemaketan pesan (*message*) antar aplikasi yang telah distandarisasi. Spesifikasi format pesan tersebut didefinisikan menjadi semacam suatu amplop berbasis XML yang dikirim beserta aturan-aturan atau cara untuk menerjemahkan representasi data dari XML. Suatu pesan XML tersebut dapat merepresentasikan data apapun, misalnya:

- *Purchase Order*
- *Request* terhadap harga suatu produk,
- *Query* dari suatu mesin pencari,
- Dan informasi lainnya yang relevan terhadap suatu aplikasi

Spesifikasi XML yang telah di standardisasi di SOAP disebut juga sebagai *SOAP Message*, terdiri dari bagian *header* dan *body*. Sebuah *SOAP Message* boleh tidak memiliki *header* tetapi harus selalu memiliki *body*.

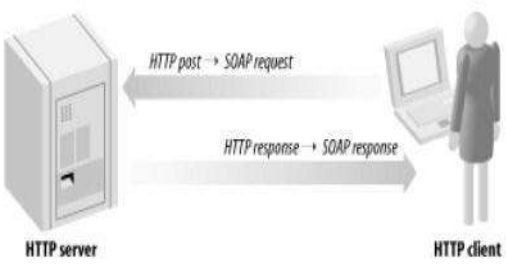


Gambar 3. Ilustrasi SOAP Message

Bagian *header* menyimpan informasi yang berhubungan dengan cara memproses *message* ini. Di dalamnya

termasuk informasi mengenai pengaturan pengiriman, autentikasi dan autorisasi, dan konteks transaksinya. Sedangkan bagian *body* menyimpan *message* yang akan diprosesnya. Sintaks XML apapun dapat dimasukan ke dalam bagian *body* ini.

SOAP Transport merupakan protokol pemaketan data yang berada di atas layer network dan transport. Sebagai suatu protokol pemaketan data, SOAP menjadi fleksibel dalam cara dan tempat ia digunakan. Contohnya, sebuah layanan web SOAP berbasis Perl yaitu *SOAP::Lite*, mendukung pertukaran *SOAP Message* di beberapa protokol diantaranya *HTTP, FTP, raw TCP, SMTP, POP3, MQSeries, dan Jabber*. Protokol HTTP merupakan protokol yang paling digunakan dalam mengirimkan *SOAP Message*. Sampai-sampai SOAP mendeskripsikan model pertukaran *message* khusus untuk di HTTP. Metoda ini cocok dengan model SOAP RPC (*Request-Response*) dikarenakan HTTP memang merupakan protokol bertipe *request-response.* Pesan *SOAP Request* di kirim ke server dengan *HTTP Request*, kemudian server menjawab dengan *HTTP Response* yang berisi pesan *SOAP Response*.



Gambar 4. Ilustrasi SOAP-over-HTTP Request-Response

Header HTTP SOAP Action didefinisikan oleh SOAP dan mengindikasikan tujuan dari *SOAP HTTP Request*. Nilainya sendiri dapat diisi sesuai kebutuhan dan ditujukan untuk memberitahukan *HTTP Server* yang harus dilakukan sebelum *HTTP server* tersebut mendekode *SOAP Message*. *SOAP Action* inipun dapat digunakan *HTTP Server* untuk memfilter permintaan yang tidak dapat diterima [7].

2.3. *Web Server*

Web server merupakan perangkat lunak untuk memberikan layanan data dengan menerima permintaan HTTP atau HTTPS dari klien yang dikenal dengan *web browser* dan mengirimkan kembali hasilnya dalam bentuk halaman-halaman web yang umumnya berbentuk dokumen HTML. Hubungan antara Web Server dan Browser Internet merupakan gabungan atau jaringan komputer yg ada di seluruh dunia. Setelah terhubung secara fisik, Protocol TCP/IP (networking protocol) memungkinkan semua komputer dapat berkomunikasi satu dengan yg lainnya. Pada saat browser meminta data web page ke server maka instruksi permintaan data oleh browser tersebut di kemas di dalam TCP yg merupakan protocol transport dan dikirim ke alamat menggunakan protocol berikutnya yaitu Hyper Text Transfer Protocol (HTTP). HTTP ini merupakan protocol yg digunakan dalam World Wide Web (WWW) antar komputer yg terhubung dalam jaringan di dunia ini.Data yg di passing dari browser ke web server disebut sebagai HTTP request. Data yg dikirim dari server ke browser disebut sebagai HTTP response. Jika data yang diminta oleh browser tidak ditemukan oleh si Web server maka akan menampilkan error yang sering anda lihat di web page yaitu *Error: 404 Page Not Found*. Di dalam penelitian ini terdapat tiga produk web server yang digunakan yaitu:

a. Apache

Secara umum arsitektur apache bekerja dengan model *thread-based* atau berbasis proses. Web server berbasis proses menggunakan proses (*thread*) untuk menerima dan merespon permintaan. Setiap permintaan akan diciptakan sebuah thread yang disimpan dalam sebuah pool pada alokasi memori tertentu dan dilakukan proses untuk menjawab, setelah proses dieksekusi kemudian hasil proses dikirimkan kembali ke klien serta dicatat dalam sebuah log

b. IIS 7

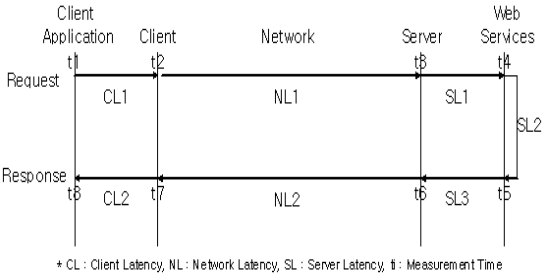
Secara struktur model kinerja arsitektur IIS melayani permintaan melalui application pool yang merupakan komponen *host worker process*. *Application pool* merupakan improvisasi dari arsitektur iis. Karena situs web yang terpisah dan aplikasi web yang terpisah dapat dilayani melalui konfigurasi *application pool*. Didalam application pool permintaan dari klien diproses oleh *event-handler* dan permintaan tersebut diterjemahkan dan dieksekusi dengan mengoleksi informasi yang diperlukan sebagai jawaban permintaan melalui ASPX (*active server page*). Proses dilanjutkan dengan mengirimkan respon ke klien dan mencatat log bahwa permintaan telah direspon.

c. NginX

Secara umum web server nginx menerapkan model *event-based*. Model *event-based* hanya menggunakan sebuah thread untuk memproses permintaan dan mengelola *event* dengan menggunakan multiplexing dan banyak notifikasi *event*. Setiap koneksi diproses secara sangat efisien dalam sejumlah proses *single threaded* yang disebut *workers*. Dalam setiap nginx, *workers* dapat menangani ribuan koneksi bersamaan dan permintaan per detik

2.4. *Latency*

latency adalah waktu antara pengiriman suatu permintaan dan menerima tanggapan. *Latency* merupakan salah satu parameter pengukuran kinerja yang digunakan pada aplikasi perangkat lunak [4]. Terdapat dua jenis *latency*. *Latency* koneksi dan *latency* permintaan. *latency* koneksi mencerminkan waktu yang dibutuhkan untuk membuat sambungan, *latency* permintaan mencerminkan waktu untuk menyelesaikan transfer data sekali sambungan. Dalam penelitian ini *latency* yang diukur menggunakan *user perceived latency* yang meliputi jumlah koneksi dari permintaan *latency*, ditambah *latency* jaringan karena koneksi wan (*wireless area network*) atau router.

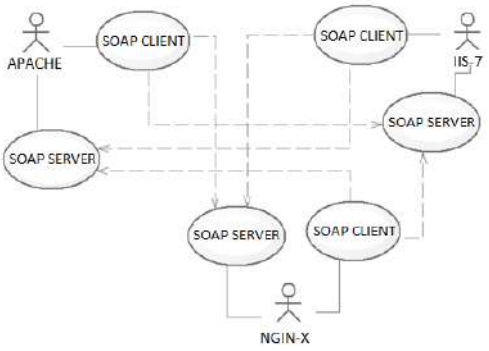


Gambar 5. komponen *latency*

Tiga komponen *latency* dalam pengukuran *latency* terlihat pada diatas. CL (*Client Latency*) waktu *latency* yang diperlukan oleh klien saat mengirim ke jaringan dan menerima data dari jaringan. NL (*Network Latency*) waktu yang diperlukan jaringan dalam mengirimkan data. SL (*Server Latency*) waktu yang diperlukan server menerima permintaan melayani permintaan dan mengirimkan hasil permintaan ke jaringan.

PERANCANGAN SISTEM

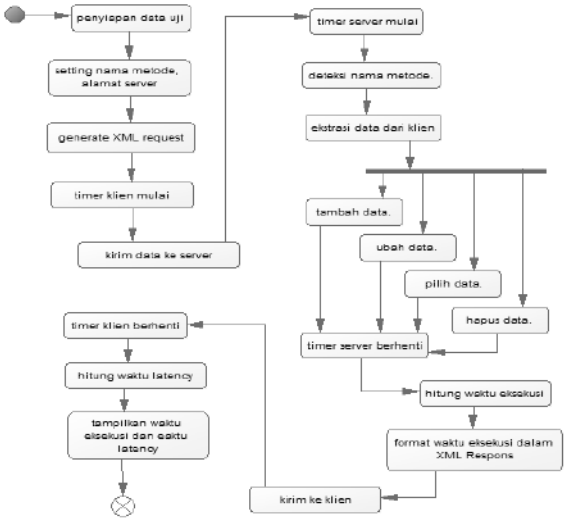
Sistem yang dirancang dalam penilaian kinerja SOAP dilakukan dengan membuat desain sistem dalam bentuk sebuah model aplikasi berbasis pesan yang dapat dieksekusi menggunakan aplikasi lain menggunakan metoda SOAP. Berikut rancangan model yang didesain untuk penilaian kinerja SOAP pada tiga buah server.



Gambar 6. Use Case Model Penilaian kinerja

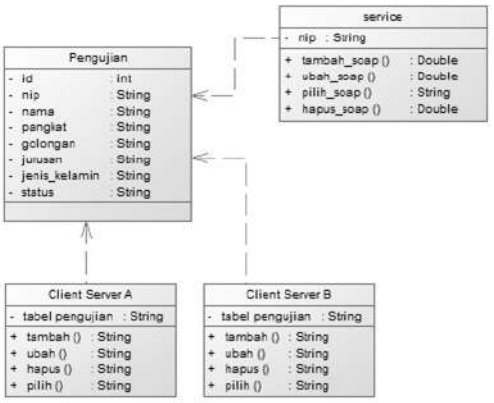
Use case model yang direncanakan bertujuan untuk mengukur *latency* ketiga metode layanan web pada tiga jenis web

server yang memiliki perbedaan platform dan arsitektur. Pada gambar 6 dapat dilihat masing-masing web server memiliki modul klien dan modul server layanan web yang dibuat dengan menerapkan metoda SOAP. Kebutuhan penilaian adalah menghasilkan keluaran berupa data *latency* dalam satuan mili detik untuk semua fungsi yang dimiliki masing-masing aktor. Secara lebih rinci aktivitas yang direncanakan pada setiap server atau actor dalam model use case dideskripsikan dalam diagram aktivitas yang terlihat pada gambar 7 di bawah ini.



Gambar 7. Activity diagram penilaian kinerja

berdasarkan diagram aktivitas tersebut, masing- masing aplikasi dirancang sebuah database dengan struktur yang sama sehingga setiap aplikasi akan memiliki layanan yang dapat diakses oleh server lain. Adapun transaksi antar server berupa penambahan, perubahan, pemilihan maupun penghapusan data pada tabel yang berada di server. Keluaran dari layanan adalah berupa data waktu eksekusi yang diperlukan oleh server dalam memproses instruksi dari klien. Representasi model database dirancang dalam sebuah model class diagram sebagai berikut



Gambar 8. Class Diagram penilaian kinerja

PENGEMBANGAN SISTEM

Hasil desain sistem yang dirancang melalui pemodelan *use case*, *activity* dan *class* pada bagian sebelumnya dikembangkan dengan membuat aplikasi yang dipasang pad ketiga server. Pengembangan dilakukan dengan menggunakan *CodeIgniter* yang merupakan framework aplikasi berbasis web dengan bahasa PHP. Sedangkan database yang dibangun menggunakan *mysql* adapun menu-menu yang dikembangkan pada aplikasi menyesuaikan pada kebutuhan masing-masing server. Berikut gambaran menu pada masing-masing server.

Tabel 1: Implementasi Menu Aplikasi Penilaian Kinerja Soap

Server	Menu
Apache	Client nginX-Tambah Data
	Client NginX-Ubah Data
	Client NginX-Hapus Data
	Client NginX-Pilih Data
	Client IIS-Tambah Data
	Client IIS-Ubah Data
	Client IIS-Hapus Data
	Client IIS-Pilih Data
IIS-7	Client nginX-Tambah Data
	Client NginX-Ubah Data
	Client NginX-Hapus Data
	Client NginX-Pilih Data
	Client Apache-Tambah Data
	Client Apache-Ubah Data
	Client Apache-Hapus Data
	Client Apache-Pilih Data

NginX	Client Apache-Tambah Data
	Client Apache-Ubah Data
	Client Apache-Hapus Data
	Client Apache-Pilih Data
	Client IIS-Tambah Data
	Client IIS-Ubah Data
	Client IIS-Hapus Data
	Client IIS-Pilih Data

Aplikasi yang dikembangkan menggunakan kerangka MVC *(Model–View–Controller)* di mana kodefikasi yang dikembangkan terdiri dari 2 modul yaitu modul klien dan modul service. Modul tersebut terpasang pada tiga server dan masing-masing server akan berfungsi sebagai klien dan sebagai server. Adapun struktur file dalam pengembangan modul ditunjukkan pada tabel 2.

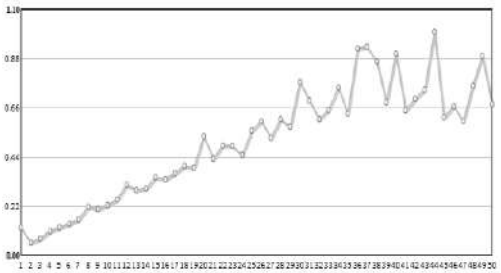
Tabel 2. Struktur File Pengembangan sistem penilaian kinerja SOAP

Modul	**Nama File**	**Fungsi**
Modul Klien	soap_tambah	Permintaan tambah data
	soap_ubah.php	permintaan perubahan data
	Soap_hapus.php	Permintaan penghapusan data
	Soap_pilih.php	Permintaan pemilihan data
Modul Server	Modul_uji_soap.php	Eksekusi layanan web berdasar permintaan klien

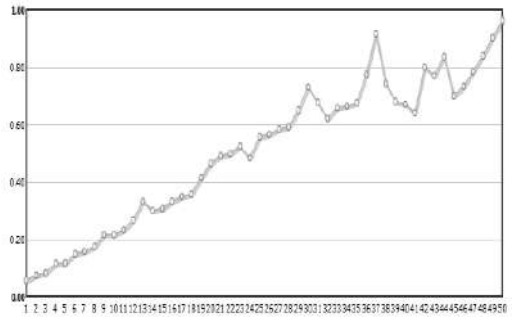
Data beban kerja yang dibuat terdiri dari tujuh field data dan besaran beban yang dibuat adalah jumlah record dalam data array yang dikirimkan untuk diinputkan pada basis data server penerima. Pada proses perubahan, data array yang dikirimkan dalam *xml request* adalah data perubahan tujuh kolom data dimana salah satu kolomnya menjadi parameter *record* yang dirubah. Besaran beban dalam proses perubahan adalah jumlah *record* yang mengalami perubahan data pada server penerima. Pada proses pemilihan data, data *xml request* hanya berisi satu data parameter sebagai bahan permintaan data dan respon yang akan dihasilkan adalah data lengkap tujuh kolom berdasarkan parameter permintaan. Besaran beban kerja didasarkan jumlah data parameter yang dikirimkan. Hal ini juga berlaku untuk proses penghapusan data.

HASIL PENGUJIAN

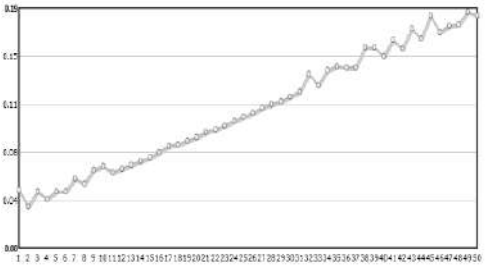
Hasil pengujian dalam penilaian kinerja berupa grafik nilai *latency* dari beban data 1 set data hingga 50 set data untuk masing-masing transaksi. Secara umum hasil penilaian pada tiap server memperlihatkan adanya pengaruh beban data terhadap nilai kecuali untuk proses penghapusan data *latency* Berikut beberapa contoh grafik data nilai *latency* pada salah satu server.



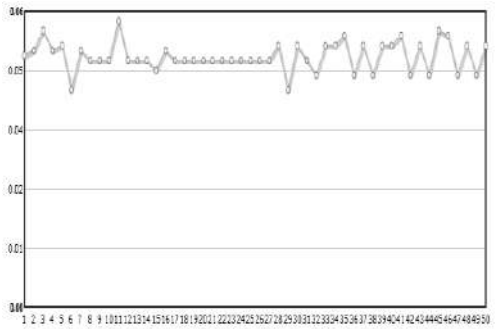
Gambar 9. Nilai *Latency* Tambah Data



Gambar 10. Nilai *Latency* Ubah Data



Gambar 11. Nilai *Latency* Ubah Data



Gambar 12. Nilai *Latency* Ubah Data

Proses penghapusan data memiliki nilai *latency* yang cenderung stabil hal ini disebabkan parameter yang dikirimkan kepada server adalah sama, dalam penelitian ini adalah kode unik nip. Dan dalam pengujian didapatkan eksekusi data hingga 50 data set tidak banyak mempengaruhi nilai *latency*. Selain itu hasil penililaian kemampuan kinerja server dalam mengeksekusi metoda SOAP diperlihatkan dalam tabel nilai *latency* berdasarkan server layanannya.

Tabel 3 Nilai *Latency* Penambahan Data

Descriptive Statistics				
	N	Min	Max	Mean
nginx-apache-soap	50	.010	.160	.04360
apache-nginx-soap	50	.032	.110	.06618
nginx-iis-soap	50	.016	.125	.06680
apache-iis-soap	50	.031	.139	.08328
iis-apache-soap	50	.057	.891	.48714
iis-nginx-soap	50	.044	1.417	.52428
Valid N (listwise)	50			

Tabel 4 Nilai *Latency* Perubahan Data

Descriptive Statistics				
	N	Min	Max	Mean
nginx-apache-soap	50	.012	.246	.12626
nginx-iis-soap	50	.016	.265	.14696
apache-nginx-soap	50	.024	.352	.18222
apache-iis-soap	50	.031	.390	.21154
iis-apache-soap	50	.058	.963	.50946
iis-nginx-soap	50	.054	1.047	.51032
Valid N (listwise)	50			

Tabel 5 Nilai *Latency* Pemilihan Data

Descriptive Statistics				
	N	Min	Max	Mean
nginx-apache-soap	50	.011	.094	.05236
nginx-iis-soap	50	.016	.109	.06580
apache-nginx-soap	50	.029	.107	.07178
apache-iis-soap	50	.031	.109	.07298
iis-nginx-soap	50	.030	.181	.10330
iis-apache-soap	50	.033	.187	.10916
Valid N (listwise)	50			

Tabel 6 Nilai *Latency* Penghapusan Data

Descriptive Statistics				
	N	Min	Max	Mean
nginx-apache-soap	50	.013	.025	.01626
nginx-iis-soap	50	.016	.047	.02982
apache-nginx-soap	50	.031	.052	.03442
iis-nginx-soap	50	.037	.054	.03904
apache-iis-soap	50	.031	.062	.04660
iis-apache-soap	50	.044	.058	.05084
Valid N (listwise)	50			

Berdasarkan keempat tabel tersebut dihasilkan bahwa web server ngin-x memiliki kecepatan akses yang lebih baik dari web server lainnya

KESIMPULAN

Dari Hasil Penilaian kinerja layanan web menggunakan metoda SOAP yang diujikan pada tiga buah server berbeda platform, dapat disimpulkan (1) Nilai *latency* akan menjadi bertambah sesuai dengan jumlah data yang ditransaksikan. Besar nilai *latency* ditentukan dari besarnya variabel data yang dikirimkan ke server. (2) Web Server Ngin-X

merupakan web server yang memiliki akses tercepat dalam mengeksekusi permintaan klien

DAFTAR PUSTAKA

[1] Susan Dian, 2008. *Web Service sebagai solusi integrasi data pada sistem informasi akademik universitas bina darma*

[2] Hartati Deviana, 2011, *Penerapan XML Web service pada Sistem Distribusi Barang*

[3] Eko budi cahyono, 2007, *Model dan Komputasi Terdistribusi dengan Platform Web Service*

[4] Mohamad Ibrahim Ladan, Ph.D, 2011. *Web Services Metrics A Survey and A Classification*

[5] Kuyoro Shade, 2012. *Quality of Service (Qos) Issues in Web Services*

[6] Ethan Ceramy 2002. *Web Service Esential.* Packt Publishing.

[7] Juric, M.B. 2007. *SOA Approach to Integration*. Packt Publishing.