Kinerja Web Service pada Proses Integrasi Data

Yogiswara, Wijono, dan Harry Soekotjo Dahlan

Abstrak—Layanan web (*Web Service*) banyak di-implementasikan pada proses integrasi presentasi dan juga dapat digunakan untuk integrasi data. tiga metode layanan web. *Extensible Markup Language Remote Procedure Call (XML-RPC), Simple Object Access Protocol (SOAP) dan Representational State Transfer (REST)* memiliki spesifikasi yang berbeda dan memerlukan algoritma yang berbeda dalam implementasinya.

Penelitian ini difokuskan untuk mengembangkan sebuah model integrasi data yang mengimplementasikan seluruh metode layanan web pada aplikasi sistim informasi terpisah serta melakukan analisis dengan mengevaluasi kinerja untuk mendapatkan konfigurasi terbaik. Proses evaluasi dilakukan dengan mengukur kecepatan layanan web pada semua jenis transaksi. Parameter yang digunakan dalam mengevaluasi kinerja adalah pengukuran lama waktu yang diperlukan dari pengiriman permintaan oleh klien sampai klien menerima hasil permintaan tersebut dari server atau biasa disebut *user perceived latency*.

Hasil penelitian menunjukkan bahwa layanan web dapat digunakan untuk mengembangkan model integrasi data. Dan konfigurasi klien "Apache", server "Nginx" dengan menggunakan metode "*REST*" memiliki kecepatan terbaik untuk diimplementasikan dalam proses integrasi data.

***Kata Kunci*—integrasi data, *latency*, metode *xmlrpc soap rest, web service.*.**

I. PENDAHULUAN

PROSES integrasi sistim informasi terjadi seiring dengan perkembangan proses bisnis serta kebutuhan pengguna informasi dalam sebuah organisasi. Terdapat empat tipe integrasi dalam sistim informasi yaitu integrasi data, integrasi aplikasi, integrasi proses bisnis dan integrasi presentasi (1). Integrasi data merupakan solusi awal dalam proses integrasi sistim informasi proses ini dilakukan dengan pertukaran data antar basis data yang terdapat pada aplikasi terpisah. Permasalahan pada proses integrasi data adalah perbedaan struktur basis data yang akan diintegrasikan serta perbedaan platform aplikasi.is

Yogiswara adalah Dosen Politeknik Negeri Jember dan Mahasiswa Program Studi Magister Teknik Elektro Universitas Brawijaya, Malang, Indonesia (email: yogipoltek@gmail.com)

Wijono adalah Ketua Program Studi Magister Elektro, Fakultas Teknik Universitas Brawijaya, Malang, Indonesia, (Telp: 081555788082, email: wijono@ub.ac.id)

Harry Soekotjo Dachlan adalah dosen Teknik Elektro Universitas Brawijaya, Malang, Indonesia(Telp.08155555811; email : harrysd@brawijaya.ac.id).

Teknologi integrasi sistim informasi sebelumnya seperti *Common Object Request Broker Architecture* (CORBA), *Distributed Component Object Model* (DCOM) dan *Remote Method Invocation* (RMI) memiliki ketergantungan terhadap platform sistim operasi dan tidak memiliki kemampuan interoperabilitas sehingga memerlukan infrastruktur yang kompleks untuk diimplementasikan pada proses integrasi data (2). *webservice* dapat menyelesaikan permasalahan interoperabilitas antar aplikasi yang berbeda platform baik perangkat lunak aplikasi, sistim operasi maupun bahasa pemrograman (1).

Isu utama teknologi layanan web adalah permasalahan kualitas layanan atau Qos (*Quality of service*). matrik kualitas layanan pada layanan web terbagi dalam beberapa hal antara lain *accessibility, reliability, performance (3).* Pengukuran kinerja layanan web dapat dilakukan berdasarkan *latency* yang merupakan lama waktu yang diperlukan dari pengiriman permintaan oleh klien sampai menerima hasil permintaan tersebut dari server (4).

Terdapat tiga metode yang telah distandarisasi konsorsium web dalam melakukan pertukaran informasi menggunakan layanan web yaitu *Extensible Markup Language Remote Procedure Call* (XML-RPC), *Simple Object Access Protocol* (SOAP) dan *Representational State Transfer* (*REST).* Aspek lain yang mempengaruhi kinerja layanan web adalah platform web server. Implementasi layanan web pada web server yang berbeda platform akan menghasilkan kinerja yang berbeda (5).

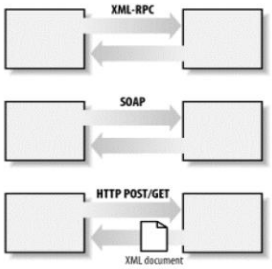
Untuk menghasilkan konfigurasi proses integrasi berbasis layanan web yang optimal. Model integrasi, platform web server dan penerapan metode pertukaran data menggunakan teknologi layanan web yang terukur memerlukan sebuah kajian yang sistematis. Oleh karena itu penulis membuat prototipe model integrasi data yang menghasilkan penilaian kinerja tiga metode layanan web pada tiga platform web server berbeda.

II. DASAR TEORI

A. Web Service

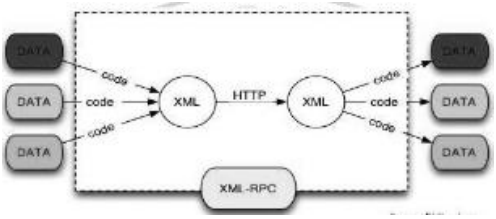
Layanan web (*Web Service*) adalah Suatu aplikasi yang programmable, dapat diakses sebagai komponen melalui protokol standard web menggunakan protokol standard web. karakteristik layanan web diantaranya pesan dengan standar XML dan tidak terikat pada satu sistem operasi atau bahasa pemrograman. Ada beberapa 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 mencakup dokumentasi yang dapat terbaca manusia sehingga pengembang mudah menggunakan layanan tersebut.



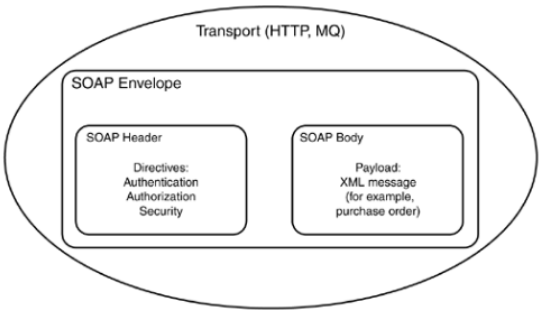
Gambar. 1. Metode pengiriman pesan XML

XML-RPC adalah sebuah implementasi protokol pemanggilan prosedur jarak jauh yang menggunakan XML untuk mengkodekan panggilan dan HTTP sebagai mekanisme transportasi yang bisa membuat perangkat lunak berjalan pada sistem operasi yang berbeda dan lingkungan yang berbeda melalui internet atau sederhananya adalah *cross-platform* komputasi terdistribusi yang berbasis pada standar Internet. XML-RPC bekerja dengan mengirimkan permintaan HTTP ke server yang dikirimkan dalam bentuk XML yang kemudian dieksekusi oleh prosedur pada server dan nilai balikannya juga diformat dalam bentuk XML.



Gambar. 2. metode pengiriman pesan XML

SOAP merupakan suatu protokol pemaketan pesan (*message*) antar aplikasi yang telah distandardisasi. Spesifikasi format pesan tersebut didefinisikan menjadi semacam suatu amplop berbasis XML yang dikirim beserta aturan-aturan atau cara untuk menerjemahkan representasi data dari XML.



Gambar. 3. Ilustrasi SOAP Message

Suatu pesan XML tersebut dapat merepresentasikan data apapun, 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*. Bagian *header* menyimpan informasi yang berhubungan dengan cara memproses *message* ini 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*.

REST (*Representational State Transfer*) sebenarnya bukan sebuah arsitektur tapi lebih mendekati kumpulan *constraint* (batasan) yang ketika diterapkan pada desain sebuah sistem, akan menjadi jenis arsitektur perangkat lunak. REST service harusnya memenuhi batasan-batasan seperti:

1. *Resource Identification*: Semua sumberdaya serta statusnya yang berhubungan dengan aplikasi diberikan identifikasi yang unik dan identifikasi tersebut harus bersifat global.
2. *Uniform Interface*: REST service menampilkan semua sumberdaya dan interaksinya dengan antarmuka yang seragam. Dalam layanan web REST untuk antarmuka seragam ini menggunakan *Uniform Resource Identifier* (URI).
3. *Self-Describing Message*: untuk setiap interaksi dengan sumberdaya melalui interface yang seragam, REST membutuhkan representasi dari sumberdaya yang menggambarkan semua aspek penting yang dimiliki oleh sumberdaya tersebut. Representasi dari sumberdaya sendiri adalah semua hal yang dikirim antara klien dan server.
4. *Stateless Interaction*: Setiap interaksi antara klien dan server harus memiliki status sendiri (atau dengan kata lain tidak dipengaruhi sesi klien). Jadi server hanya akan memantau status sumberdaya bukan sesi klien.

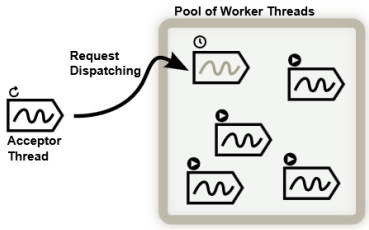
Semua batasan tersebut tidak berpengaruh dengan teknologi yang akan digunakan untuk implementasi. Batasan tersebut hanya mendefinisikan bagaimana data ditransfer antar komponen dan keuntungan apa yang didapat dan tidak perlu mencari teknologi atau protokol jaringan baru untuk mengimplementasi-kannya.

B. Arsitektur 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. Didalam penelitian ini terdapat tiga produk web server yang digunakan. Ketiga web server tersebut memiliki perbedaan dalam arsitekturnya.

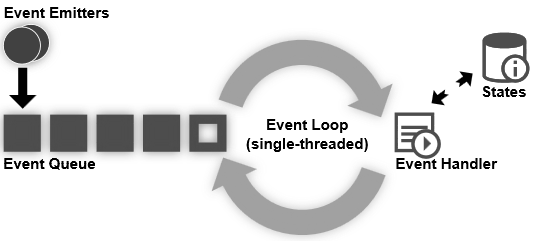
Web server apache bekerja dengan model *thread-based*. Web server ini menggunakan proses (*thread*) untuk menerima dan merespon permintaan. Web server NginX terinspirasi oleh pengembangan perangkat lunak

berkelanjutan berbasis event. Sehingga bentuk yang dihasilkan adalah modular, *event-driven, asynchronous, single-threaded, non-blocking arsitektur.*



Gambar. 4. multi thread model

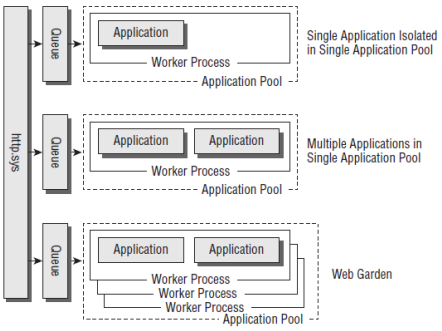
Model *event-based* hanya menggunakan sebuah thread untuk memproses permintaan dan mengelola *event* dengan menggunakan multiplexing dengan banyak notifikasi *event*. Setiap koneksi diproses dalam sejumlah proses *single threaded* yang disebut *workers*. Dalam setiap nginx, *workers* dapat menangani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



Gambar. 5. Event Driven Model

Gambar diatas adalah ilustrasi internal dari arsitektur event-driven. Sebuah *single-threaded* mengkonsumsi event dari antrian yang berurutan dan dieksekusi oleh *event handler* proses eksekusi tersebut disimpan dalam sebuah states sebagai catatan notifikasi.

IIS atau Internet Information Services adalah sebuah HTTP web server yang digunakan dalam sistem operasi server Windows. Layanan ini berfungsi sebagai pendukung protokol TCP/IP yang berjalan dalam lapisan aplikasi. Secara struktur model kinerja arsitektur IIS melayani permintaan melalui application pool yang merupakan komponen *host worker process*.

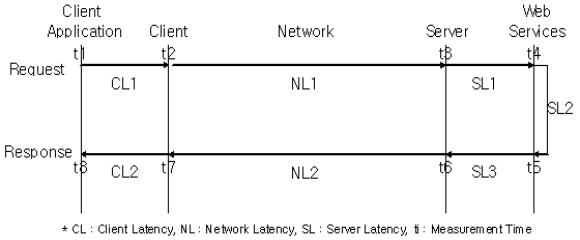


Gambar. 6. Jenis konfigurasi application pool

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. *Latency*

Latency adalah waktu antara pengiriman suatu permintaan dan menerima tanggapan (3). *Latency* merupakan salah satu parameter pengukuran kinerja yang digunakan pada aplikasi perangkat lunak. Terdapat dua jenis latency. *Latency* koneksi dan latency permintaan. *latency* koneksi mencerminkan waktu yang dibutuhkan untuk membuat sambungan, sementara 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. Tiga komponen latency dalam pengukuran latency terlihat pada gambar 7.

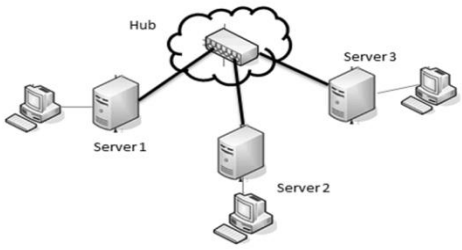


Gambar 7. komponen latency

CL (*Client Latency*) waktu latency yang diperlukan oleh klien saat mengirim ke jaringan dan menrima 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.

III. METODE PENELITIAN

Terdapat dua aktivitas dalam penelitian ini. Pertama adalah mengembangkan model integrasi data untuk menunjukkan layanan web dapat menyelesaikan permasalahan perbedaan struktur basis data pada aplikasi yang akan berintegrasi. Kedua adalah membuat pengujian kinerja layanan web melalui pengukuran *latency* untuk mendapatkan konfigurasi terbaik. Gambar dibawah ini mengilustrasikan rencana topologi yang akan dilbuat pada penelitian ini.



Gambar. 8. Diagram *Use Case* Model Integrasi Data

A. Model Integrasi Data

Langkah awal dalam perancangan model prototipe integrasi data dengan melakukan analisis kebutuhan model yang diinginkan kemudian dilanjutkan

perancangan basis data dan proses interaksi antar basis data, setelah itu membuat antarmuka yang dipasang pada masing masing web server. Setiap web server akan memiliki sebuah layanan web yang dapat diakses aplikasi lain. Selain itu web server tersebut juga berfungsi sebagai klien yang dapat mengeksekusi layanan web yang dimiliki web server lainnya. Fokus utama prototipe ini adalah mengimplementasikan metode pertukaran data (*xmlrpc* ,*soap*, *rest*) sebagai media pertukaran data pada tiga aplikasi terpisah yang dipasang pada tiga web server berbeda platform.

B. Evaluasi Kinerja *Web Service*

Pada setiap aplikasi dikembangkan sebuah modul klien untuk mengakses layanan aplikasi lain dan sebuah modul server yang menyediakan layanan untuk diakses klien. Modul klien akan berisi sebuah skema permintaan berupa penambahan data, perubahan data, pemilihan data dan penghapusan data pada sebuah tabel di server. Waktu *latency* akan dicatat oleh klien saat klien mengirim permintaan sampai mendapatkan respon dari web server. Data *latency* yang diambil terdiri dari 18 konfigurasi yang dikembangkan dalam model integrasi ini. Konfigurasi terebut ditunjukkan pada tabel 1. Nilai *latency* yang dicatat untuk masing masing pengujian terdiri dari 50 nilai latency berdasarkan beban data yang ditransaksikan yaitu 1 sampai 50 set data.

TABEL I
KONFIGURASI PEGUJIAN EVALUASI KINERJA

No	Server	Klien	Metode	Transaksi
1	Iis 7	apache	xmlrpc	4
2	Iis 7	apache	soap	4
3	Iis 7	apache	rest	4
4	Iis 7	nginx	xmlrpc	4
5	Iis 7	nginx	soap	4
6	Iis 7	nginx	rest	4
7	apache	Iis 7	xmlrpc	4
8	apache	Iis 7	soap	4
9	apache	Iis 7	rest	4
10	apache	nginx	xmlrpc	4
11	apache	nginx	soap	4
12	apache	nginx	rest	4
13	nginx	apache	xmlrpc	4
14	nginx	apache	soap	4
15	nginx	apache	rest	4
16	nginx	Iis 7	xmlrpc	4
17	nginx	Iis 7	soap	4
18	nginx	Iis 7	rest	4
		Jumlah Data Seri		**72**

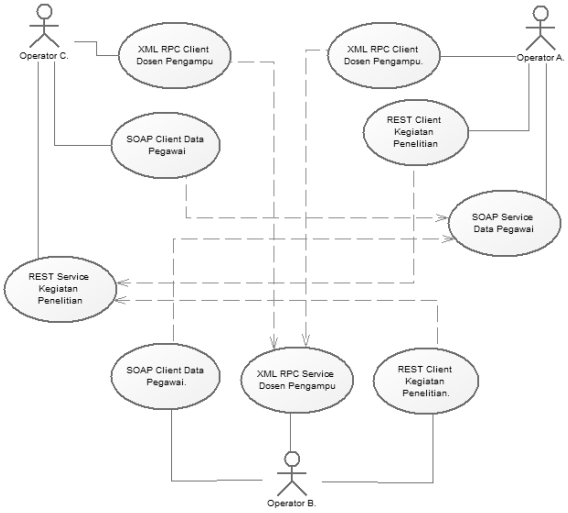
Dari 18 konfigurasi tersebut akan akan ditemukan konfigurasi terbaik yang ditetapkan menggunakan analisis deskriptif.

IV. HASIL DAN PEMBAHASAN

A. Model Integrasi data

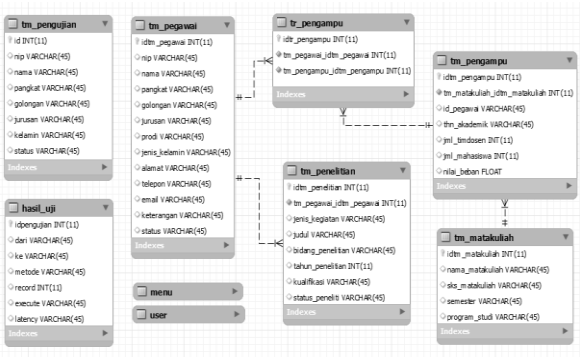
Dalam pengembangan model integrasi data dalam penelitian ini menggunakan studi kasus sebuah organisasi perguruan tinggi yang memiliki aplikasi terpisah yaitu aplikasi kepegawaian, aplikasi penelitian dan aplikasi akademik. Masing masing aplikasi akan memiliki layanan web yang datanya dapat digunakan oleh aplikasi lain.

Use case model integrasi data menggambarkan fungsi-fungsi dalam sebuah skema integrasi data yang mengimplementasikan tiga metode teknologi layanan web yaitu *XMLRPC*, *SOAP* dan *REST* menjadi media pertukaran data antara basis data. Didalam studi kasus penelitian ini dibuat tiga buah basis data.

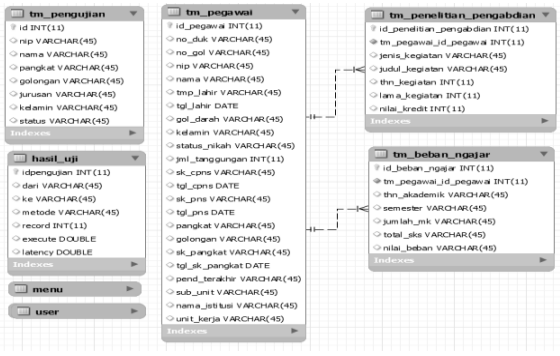


Gambar. 9. Diagram *Use Case* Model Integrasi Data

Dari gambar diatas ditunjukkan bahwa masing masing aktor memiliki sebuah layanan web yang melayani permintaan data bagi aplikasi lainnya sehingga aplikasi tersebut berperan sebagai *provider*. Selain itu masing masing aktor juga memiliki modul untuk meminta data kepada aplikasi lain yang menunjukkan peran operator sebagai klien. Pada gambar tersebut model direncanakan agar tiga aplikasi tersebut menerapkan metode yang berbeda.



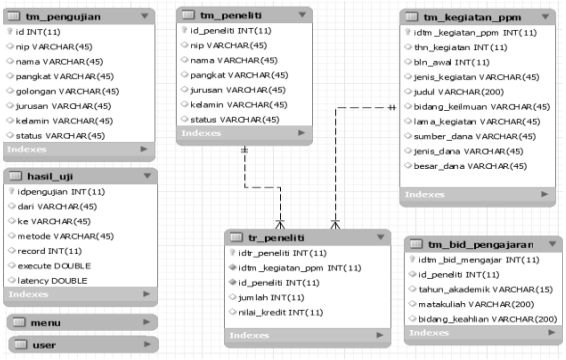
Gambar. 10 Basis Data Aplikasi Akademik



Gambar. 11. Basis Data Aplikasi Kepegawaian

Dalam studi kasus yang dikembangkan pada model integrasi data ini Layanan aplikasi kepegawaian direncanakan menggunakan data pegawai sebagai yang dapat dimanfaatkan oleh aplikasi lainnya. Layanan

aplikasi akademik menggunakan data dosen pengampu yang berisi data matakuliah yang diajar sebagai data yang menjadi rujukan, sedangkan layanan aplikasi penelitian Menggunakan data hasil penelitian sebagai data rujukan aplikasi lainnya. Adapun desain basis data yang ditunjukkan pada gambar 10 untuk aplikasi akademik , gambar 11 untuk apalikasi kepegawaian dana gambar 12 untuk aplikasi penelitian. Ketiga basis data tersebut memiliki struktur data yang berbeda.



Gambar. 12. Basis Data Aplikasi Penelitian

Penetapan field kunci (*Unique Field Key*) digunakan untuk menetapkan skema integrasi. Aplikasi kepegawai menggunakan NIP pegawai sebagai field kunci yang bisa digunakan oleh aplikasi lain. Aplikasi Akademik menggunakan gabungan NIP, Tahun Akademik dan kode matakuliah sebagai field kunci dan Aplikasi Penelitian menggunakan NIP, thn kegiatan dan judul penelitian sebagai field kunci.

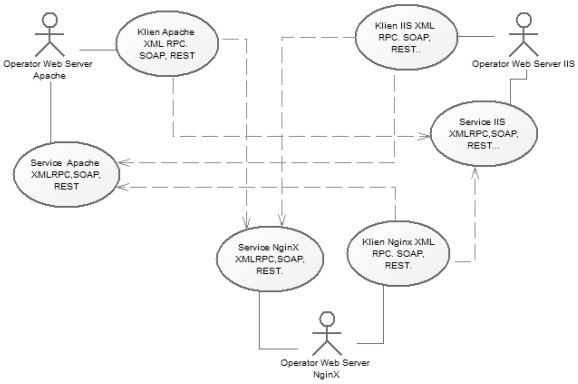
Hasil implementasi model integrasi data menunjukkan bahwa *web service* dapat digunakan sebagai media pertukaran data. Dari sisi pemrograman metode soap merupakan metode paling mudah diimplementasikan sedangkan metode rest memerlukan aplikasi berbasis obyek. Jenis transaksi pemilihan data menjadi mode yang paling aman untuk meng-integrasikan data.

B. Evaluasi kinerja *Web Service*

Dengan menggunakan aplikasi yang dikembangkan pada model integrasi data. Untuk evaluasi kinerja *web service* dilakukan penambahan modul pada masing masing aplikasi dan ditambahkan tabel pengujian pada server sebagai layanan yang dapat diakses oleh klien untuk transaksi penambahan, perubahan, pemilihan serta penghapusan data. Transaksi tersebut dilakukan melalui permintaan dari klien dengan menggunakan metode *web service*. Data pengukuran disimpan didalam tabel hasil pengujian yang terdapat pada masing masing aplikasi.

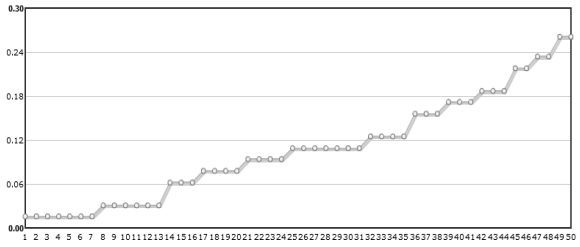
use case model yang direncanakan untuk model pengujian kinerja layanan web bertujuan untuk mengukur *latency* ketiga metode layanan web pada tiga jenis web server yang memiliki perbedaan arsitektur. Pada gambar 13 dapat dilihat masing masing web server memiliki modul klien dan modul server layanan web yang dibuat dengan menerapkan semua metoda (*XML-RPC, SOAP, REST*).

Kebutuhan pengujian adalah menghasilkan keluaran berupa data *latency* dalam satuan mili detik untuk semua fungsi yang dimiliki masing-masing aktor. Didalam implementasinya. Tabel yang digunakan untuk melakukan transaksi dimasukkan dalam masing masing basis data yang telah dibuat pada model integrasi.



Gambar. 13. Diagram Use Case Evaluasi Kinerja *Web Service*

Pengukuran waktu *latency* dilakukan melalui program klien. Program tersebut mengirimkan permintaan berupa penambahan data, perubahan data, pemilihan data dan penghapusan data, permintaan tersebut akan dilayani oleh server dan server mengeksekusi permintaan kedalam basis data dan mengirimkan respon data ke klien. Pengiriman permintaan dilakukan berulang ulang berdasarkan rancangan beban data. Beban data berupa sebuah data array yang terdiri baris data yang dirancang dari 1 baris data sampai 50 baris data. Sehingga untuk satu pengujian evaluasi kinerja akan didapatkan 50 nilai latency dari beban 1 baris data hingga 50 baris data. Nilai latency tersebut disimpan dalam basis data dan ditampilkan oleh aplikasi dalam bentuk sebuah grafik nilai latency terhadap jumlah baris data. Berikut salah satu bentuk keluaran grafik yang dihasilkan oleh aplikasi.



Gambar. 14. Grafik nilai latency terhadap beban data

Dari pengamatan grafik yang dihasilkan berdasarkan konfigurasi yang ditetapkan menunjukkan adanya pengaruh beban data terhadap lamanya waktu latency. Hal ini ditunjukkan pada setiap penambahan jumlah beban data menghasilkan nilai latency yang semakin besar.

Tabel II dan tabel III adalah urutan data nilai latency penambahan data dan perubahan data berdasarkan 18 konfigurasi yang telah ditetapkan. no urutan pada tabel tersebut merupakan nilai latency tercepat. Tabel IV dan V adalah data urutan nilai latency untuk jenis transaksi, pemilihan data dan penghapusan data. Tabel tersebut berisi data hasil analisis deskriptif menggunakan metode *tendency central* yang menampilkan nilai latency terkecil dan terbesar serta rata rata nilai latency dari 50 data yang dicatat. Dari hasil analisis deskriptif tersebut dapat diamati dan disimpulkan bahwa metode rest

memiliki kinerja lebih baik dari metode lainnya, sedangkan konfigurasi yang memiliki nilai latency tercepat yang dihasilkan adalah konfigurasi dengan web server "Apache" sebagai klien, web server "Nginx" sebagai pemilik service atau sebagai server, sedangkan metode terbaik yang digunakan untuk proses integrasi data adalah metode rest.

TABEL II
STATISTIK DESKRIPTIF NILAI LATENCY PENAMBAHAN DATA

No	server-klien-metode	N	Min	Max	Mean
1	nginx-apache-rest	50	0.002	0.016	0.0038
2	apache-nginx-rest	50	0.009	0.076	0.01384
3	apache-iis-rest	50	0.009	0.016	0.01562
4	nginx-iis-rest	50	0.001	0.031	0.0187
5	iis-nginx-rest	50	0.014	0.047	0.02238
6	iis-apache-rest	50	0.014	0.047	0.0236
7	apache-nginx-xmlrpc	50	0.016	0.162	0.0416
8	nginx-iis-xmlrpc	50	0.016	0.182	0.04234
9	nginx-apache-xmlrpc	50	0.007	0.192	0.04288
10	nginx-apache-soap	50	0.01	0.16	0.0436
11	apache-iis-xmlrpc	50	0.016	0.094	0.04986
12	iis-apache-xmlrpc	50	0.03	0.098	0.06414
13	apache-nginx-soap	50	0.032	0.11	0.06618
14	nginx-iis-soap	50	0.016	0.125	0.0668
15	iis-nginx-xmlrpc	50	0.03	0.121	0.07004
16	apache-iis-soap	50	0.031	0.139	0.08328
17	iis-apache-soap	50	0.057	0.891	0.48714
18	iis-nginx-soap	50	0.044	1.417	0.52428

TABEL III
STATISTIK DESKRIPTIF NILAI LATENCY PERUBAHAN DATA

No	server-klien-metode	N	Min	Max	Mean
1	nginx-apache-rest	50	0.006	0.188	0.09472
2	nginx-iis-rest	50	0.016	0.187	0.10144
3	apache-nginx-rest	50	0.016	0.261	0.10812
4	nginx-apache-soap	50	0.012	0.246	0.12626
5	nginx-iis-xmlrpc	50	0.016	0.25	0.12948
6	nginx-apache-xmlrpc	50	0.009	0.38	0.13828
7	nginx-iis-soap	50	0.016	0.265	0.14696
8	apache-iis-rest	50	0.016	0.328	0.15666
9	apache-nginx-xmlrpc	50	0.031	0.31	0.15994
10	apache-nginx-soap	50	0.024	0.352	0.18222
11	apache-iis-xmlrpc	50	0.031	0.343	0.1825
12	apache-iis-soap	50	0.031	0.39	0.21154
13	iis-nginx-rest	50	0.033	1.041	0.48192
14	iis-apache-xmlrpc	50	0.049	0.965	0.50638
15	iis-apache-soap	50	0.058	0.963	0.50946
16	iis-nginx-soap	50	0.054	1.047	0.51032
17	iis-apache-rest	50	0.031	1.025	0.5166
18	iis-nginx-xmlrpc	50	0.05	1.197	0.53012

TABEL IV
STATISTIK DESKRIPTIF NILAI LATENCY PEMILIHAN DATA

No	server-klien-metode	N	Min	Max	Mean
1	nginx-apache-rest	50	0.009	0.023	0.01596
2	nginx-iis-rest	50	0.016	0.031	0.0262
3	apache-nginx-rest	50	0.031	0.042	0.03848
4	apache-iis-rest	50	0.031	0.062	0.0425
5	apache-nginx-xmlrpc	50	0.016	0.075	0.0425
6	nginx-iis-xmlrpc	50	0.016	0.078	0.04272
7	nginx-apache-xmlrpc	50	0.009	0.086	0.04672
8	nginx-apache-soap	50	0.011	0.094	0.05236
9	iis-nginx-xmlrpc	50	0.016	0.086	0.0526
10	apache-iis-xmlrpc	50	0.016	0.094	0.05486
11	iis-apache-xmlrpc	50	0.022	0.087	0.05494
12	nginx-iis-soap	50	0.016	0.109	0.0658
13	apache-nginx-soap	50	0.029	0.107	0.07178
14	iis-nginx-rest	50	0.047	0.086	0.072
15	apache-iis-soap	50	0.031	0.109	0.07298
16	iis-apache-rest	50	0.071	0.088	0.0755
17	iis-nginx-soap	50	0.03	0.181	0.1033
18	iis-apache-soap	50	0.033	0.187	0.10916

Konfigurasi web klien apache, web server nginx dan metode rest berlaku pada semua jenis transaksi yaitu transaksi penambahan data, perubahan data, pemilihan data dan penghapusan data.

TABEL V
STATISTIK DESKRIPTIF NILAI LATENCY PENGHAPUSAN DATA

No	server-klien-metode	N	Min	Max	Mean
1	nginx-apache-rest	50	0.005	0.015	0.00552
2	iis-nginx-rest	50	0.009	0.015	0.01235
3	iis-apache-rest	50	0.01	0.015	0.01246
4	nginx-apache-xmlrpc	50	0.007	0.022	0.01376
5	nginx-iis-rest	50	0.001	0.031	0.0145
6	nginx-apache-soap	50	0.013	0.025	0.01626
7	apache-iis-rest	50	0.016	0.031	0.0163
8	nginx-iis-xmlrpc	50	0.008	0.031	0.01944
9	apache-nginx-res	50	0.016	0.031	0.0244
10	nginx-iis-soap	50	0.016	0.047	0.02982
11	apache-nginx-xmlrpc	50	0.029	0.031	0.03048
12	apache-iis-xmlrpc	50	0.016	0.031	0.0307
13	apache-nginx-soap	50	0.031	0.052	0.03442
14	iis-nginx-soap	50	0.037	0.054	0.03904
15	iis-nginx-xmlrpc	50	0.03	0.077	0.04274
16	iis-apache-xmlrpc	50	0.031	0.063	0.04516
17	apache-iis-soap	50	0.031	0.062	0.0466
18	iis-apache-soap	50	0.044	0.058	0.05084

Berdasarkan data tersebut secara umum perbedaan arsitektur web server menunjukkan pengaruh yang beragam terhadap nilai latency pada semua jenis transaksi.

V. KESIMPULAN

Kunci field unik (*Unique key field*) pada tabel-tabel yang akan melakukan pertukaran data menjadi dasar analisis dan desain dalam pengembangan sebuah model integrasi data.

Web server dan metode pertukaran data memiliki pengaruh terhadap lamanya proses pertukaran data. Konfigurasi web server "Nginx" dengan klien "Apache" menggunakan metode "REST" merupakan konfigurasi dengan nilai latency terbaik untuk diimplementasikan dalam proses integrasi data.

Sebagai saran untuk penelitian lebih lanjut, penambahan parameter throughput, respon time dan execution time sebagai parameter pengujian perlu dilakukan untuk menghasilkan analisis kinerja yang lebih mendalam. Selain itu modul integrasi data disempurnakan dengan modul yang dapat menerima mendeteksi metode pertukaran data sebagai bentuk fleksibilitas teknologi *web service*.

DAFTAR PUSTAKA

[1] Juric, M.B., Loganathan, R., Sarang, P., dan Jennings, F. 2007. SOA Approach to Integration, Packt Publishing, Birmingham, B27 6PA, UK.

[2] Eko Budi C. 2007. Model dan Teknologi Komputasi Terdistribusi dengan platform Web Service, Publikasi Ilmiah Universitas Muhammadiyah Malang.

[3] Ladan Mohamad Ibrahim,. Ph.D. 2011. *Web Services Metrics A Survey and A Classification*, International Conference on Network and Electronics Engineering IPCSIT vol.11 (2011) © (2011) IACSIT Press, Singapore

[4] Kuyoro Shade. 2012. Quality of Service (Qos) Issues in Web Services, IJCSNS International

[5] Toyotaro Suzumaru .2008. Performance Comparison of Web Service Engines in PHP, Java, and CG. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.