PERANCANGAN *PROTOTYPE* APLIKASI *MOBILE* UNTUK PENGAKSESAN *WEB SERVICE*

Theophilus Wellem
Fakultas Teknologi Informasi Universitas Kristen Satya Wacana
Jl. Diponegoro 52-60, Salatiga 50711, Indonesia
Email: erman_wellem@yahoo.com

Abstrak
Tulisan ini membahas perancangan prototype aplikasi mobile menggunakan Java Micro Edition (Java ME) untuk mengakses Web service. Untuk memungkinkan pengaksesan Web service oleh aplikasi Java ME, digunakan Web Service Application Programming Interface (WSA) untuk Java ME, yang menyediakan fungsi untuk melakukan parsing XML dan Remote Procedure Call. Sebagai contoh, Web service yang diakses oleh aplikasi yang dirancang adalah Web service pada suatu Sistem Infromasi Akademik. Web service ini mengembalikan Indeks Prestasi Semester, Indeks Prestasi Kumulatif, dan Nilai suatu Matakuliah berdasarkan parameter yang diberikan kepada Web Service tersebut oleh aplikasi. Hasil pengujian terhadap prototype aplikasi menunjukkan bahwa aplikasi bekerja sesuai dengan yang diharapkan. Adanya WSA membuat pemrograman Web Service client pada mobile device berbasis Java ME tidak perlu lagi melakukan hal-hal yang bersifat low-level, seperti manipulasi SOAP, HTTP, dan pemetaan tipe data antara tipe data WSDL dan tipe data pada Java, karena semuanya telah ditangani oleh WSA.

***Keyword** : Web Services, Java ME, Web Services API*

1. PENDAHULUAN

Dengan perkembangan yang pesat pada teknologi komunikasi dan informasi, operator seluler, perusahaan, maupun organisasi nirlaba, seperti pemerintah dan institusi pendidikan dapat menyediakan berbagai macam layanan kepada pelanggannya atau masyarakat secara umum. Saat ini telah terdapat banyak layanan yang dapat diakses dari *mobile device*, misalnya untuk mengetahui hasil pertandingan sepak bola, berita, *download ringtone*, *game*, gambar, dan sebagainya. Layanan-layanan *mobile* ini pada awalnya berupa aplikasi *browser-based*, dimana *mobile device* mengakses informasi di Internet melalui aplikasi *microbrowser* dan memanfaatkan WAP (*Wireless Application Protocol*). Dikenalkannya teknologi pemrograman aplikasi pada mobile device seperti *Java Micro Edition* (Java ME), khususnya *Mobile Information Device Profile* (MIDP), Microsoft .NET Compact Framework, dan Symbian C++ membuat *mobile device* dapat mengakses informasi melalui aplikasi yang langsung memanfaatkan *Hypertext Transfer Protocol* (HTTP) dan koneksi *General Packet Radio System* (GPRS). Dengan teknologi-teknologi ini dapat dibuat aplikasi *client* pada *mobile device* yang dapat mengakses layanan *mobile*.

Teknologi pemrograman pada web dan Internet dalam beberapa tahun terakhir ini juga berkembang dengan pesat. Salah satunya adalah dikenalkannya *Web service*. Menurut W3C, *Web service* merupakan suatu *software* sistem yang mendukung interaksi yang *interoperable* dari *machine to machine* melalui jaringan (World World Wide Consortium). Dalam tulisan yang lain, *Web service* didefinisikan sebagai *"loosely coupled, reusable software components that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols"* (Stencil Group). Dengan suksesnya *Web service* sebagai suatu standar teknologi *software*, memberikan peluang yang besar untuk pengembangan aplikasi terdistribusi melalui Internet. Saat ini *Web service* tidak hanya dapat diakses melalui komputer saja, tetapi juga dapat diakses melalui *mobile device*, seperti telepon seluler dan PDA, sehingga memungkinkan diciptakannya layanan *mobile* menggunakan *Web service* dan aplikasi *mobile* yang menggunakan *Web service* ini.

Saat ini di Indonesia bertumbuh bisnis aplikasi *mobile* dan layanan *mobile content*, tetapi umumnya layanan ini adalah bisnis milik perusahaan swasta. Dalam bidang pendidikan, universitas atau institusi pendidikan juga dapat menyediakan layanan mobile untuk mahasiswanya. Salah satu contohnya adalah layanan *mobile* Sistem Informasi Akademik, yang dapat memberikan informasi kepada kepada mahasiswa mengenai nilai matakuliah, indeks prestasi, dan informasi lainnya. Sistem Informasi Akademik bukan merupakan suatu hal yang baru, hampir semua institusi pendidikan telah memanfaatkan komputer dan Internet untuk proses registrasi matakuliah, pendaftaran mahasiswa baru, hingga e-learning. Walaupun Sistem Informasi Akademik telah menjadi hal yang biasa dalam institusi pendidikan, tetapi umumnya memiliki format tampilan yang kurang cocok jika diakses menggunakan *mobile device low-end*, karena itu agar Sistem Informasi Akademik dapat diakses dari *mobile device*, dikembangkan Sistem Informasi Akademik menggunakan teknologi WAP.

Selain itu dikembangkan juga aplikasi *mobile* menggunakan Java ME untuk mengakses Sistem Informasi Akademik dari *mobile device* langsung dengan protokol HTTP, jadi tidak menggunakan WAP dan WAP Gateway sebagai perantara.

Dengan adanya teknologi *Web service*, dimana *Web service* juga dapat diakses melalui *mobile device*, maka dapat dikembangkan *Web service* untuk layanan mobile Sistem Informasi Akademik. Penelitian yang ada sebelumnya (Utomo, 2005a dan Utomo, 2005b) telah menunjukkan pengembangan *Web service* untuk mengembalikan daftar mahasiswa, dengan parameter input total SKS yang telah ditempuh dan indeks prestasi kumulatif. Pada kedua penelitian tersebut, *Web service* dikembangkan menggunakan teknologi .NET dan *client* dari *Web service* adalah komputer dengan *browser* Microsoft Internet Explorer.

Tulisan ini bertujuan untuk menjelaskan perancangan dan implementasi suatu prototype aplikasi mobile untuk mengakses/mengkonsumsi Web service menggunakan teknologi Java ME (MIDP) dan Java ME Web Services API (WSA). Web service yang akan diakses merupakan *Web service* pada suatu Sistem Informasi Akademik yang dibuat menggunakan teknologi *Java Enterprise Edition* (Java EE).

2. KAJIAN PUSTAKA

Beberapa hal yang dikaji dalam bagian ini adalah Java ME, *Web service*, dan WSA.

2.1. JAVA ME

Java ME merupakan bagian teknologi Java yang ditujukan untuk *consumer* dan *embedded device*. Arsitektur Java ME mendefinisikan *configuration*, *profile*, dan *optional package* yang membentuk *Java Runtime Environment* (JRE) untuk *mobile information device* (MID). *Configuration* mendefinisikan dasar dari J2ME *runtime environment*, terdiri dari *virtual machine* dan *class library* yang merupakan kumpulan API untuk menyediakan fungsi-fungsi dasar, seperti *network connectivity* dan *memory footprint* untuk kelompok device tertentu. Untuk *mobile phone*, configuration yang digunakan adalah *Connected Limited Device Configuration* (CLDC). *Profile* merupakan perluasan dari *Configuration*. *Profile* mendefinisikan *higher-level API* untuk *application domain* yang lebih spesifik. *Profile* yang digunakan untuk pengembangan aplikasi pada *mobile phone* adalah *Mobile Information Device Profile* (MIDP) (Sun Microsystems, 2001). *Mobile Information Device Profile* (MIDP) merupakan *profile* CLDC yang menyediakan fitur untuk membuat *User Inferface* (UI), fungsi Multimedia dan Game, Konektivitas ke jaringan dengan *HyperText Transfer Protocol* (HTTP), HTTPS, datagram, socket, server socket dan komunikasi dengan serial port (Sun Microsystems, 2001).

Selain *package* dan *class* yang disediakan oleh *Configuration* dan *Profile*, Java ME dapat juga menggunakan *package* yang disediakan untuk kebutuhan yang lebih spesifik. *Package* opsional menyediakan beberapa API yang dispesifikasikan dalam *Java Specification Request* (JSR) oleh JCP. WSA merupakan *optional package*, yang dispesifikasikan pada JSR-172 (Ellis dan Young, 2003). Aplikasi Java ME (CLDC/MIDP) dikenal dengan sebutan MIDlet.

2.2. *WEB SERVICE* DAN WSA

Web service mendefinisikan suatu fungsi tertentu yang dapat diakses oleh aplikasi yang lain melalui Internet dengan menggunakan protokol-protokol yang menjadi standard Internet, yaitu (Sun Microsystems, 2004):

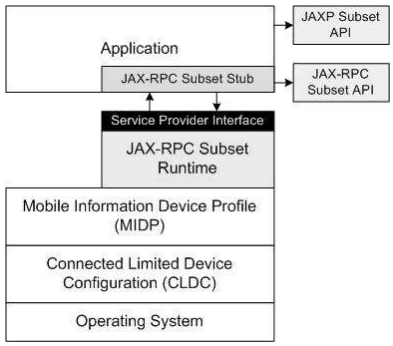
- *eXtensible Markup Language* (XML), merupakan *markup language* untuk dokumen, bersifat *portable*. Digunakan untuk mendeskripsikan konfigurasi dan informasi mengenai sistem yang dapat dipertukarkan antar aplikasi. SOAP dan WSDL ditulis menggunakan XML
- *Remote Procedure Call* (RPC), merupakan protocol yang memungkinkan pertukaran data di antara sistem-sistem yang berbeda melalui *network*/Internet.
- *Web Services Description Language* (WSDL), merupakan protokol berbasis XML yang digunakan untuk mendeskripsikan suatu *Web service* dan memfasilitasi *application-to-application communication*. Suatu *file* WSDL mendeskripsikan apa yang dilakukan oleh *service*, bagaimana untuk memanggil (*invoke*) operasi dari *service*, dimana *service* tersebut dapat ditemukan dalam jaringan/Internet, dan *interface-interface*-nya.
- *Simple Object Access Protocol* (SOAP), merupakan protokol berbasis XML yang menyediakan *envelope* untuk pertukaran obyek/data di Internet. SOAP menggunakan HTTP untuk mekanisme transportnya.

Dengan *Web service*, maka aplikasi dapat diakses oleh berbagai macam *client*, *independent* terhadap *platform* dimana *Web service* tersebut di-*publish*. Jika suatu *Web service* telah di-*publish*, maka untuk menggunakannya, *client* harus mencari *Web service* tersebut. Umumnya menggunakan suatu *network registry*, yaitu *Universal Description, Discovery and Integration* (UDDI).

WSA merupakan *optional package* Java ME untuk mendukung *Web service* dan *parsing* XML. WSA mempunyai dua *package* yaitu *Java API for XML Processing* (JAXP) yang menyediakan fasilitas XML parsing, dan *Java API for XML-based RPC* (JAX-RPC) yang menyediakan fasilitas *remote service invocation*. WSA JAXP merupakan subset dari *Simple API for XML Parsing version 2* (SAX2), sedangkan WSA JAX-RPC merupakan subset dari JAX-RPC milik Java SE (*Standard Edition*). Aplikasi *client* pada *mobile device* menggunakan JAXP subset API untuk menangani dokumen XML dan menggunakan JAX-RPC subset API untuk mengkonsumsi (*consume*) *Web service*. JAX-RPC subset mempunyai dua bagian, yaitu:

- Stub, merupakan *client-side proxy* yang akan dipanggil oleh aplikasi untuk melakukan koneksi ke *Web service*. Stub ini dihasilkan oleh Stub Generator berdasarkan *file* WSDL dari *Web service*.
- *Runtime* dan *Service Provider Interface* (SPI), digunakan oleh Stub untuk koneksi ke *Web service*.

Pada *mobile device*, biasanya aplikasi dan stub berada pada memori dari device, sedangkan bagian yang lain (*Runtime*, SPI, *profile*, dan *configuration*) berada (*embedded*) pada *device* (Ortis, 2004). Hubungan antara aplikasi, WSA, serta MIDP dan CLDC ditunjukkan pada Gambar 1.



Gambar 1. Organisasi dari Aplikasi Java ME yang menggunakan WSA

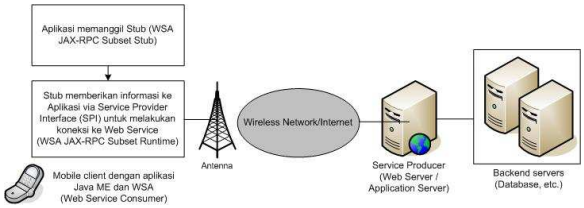
Secara umum arsitektur pengaksesan *Web service* dari *mobile device* mempunyai tiga elemen:

- Aplikasi yang akan menggunakan/mengkonsumsi web service pada *mobile device*. Aplikasi ini mempunyai WSA stub yang menggunakan WSA *runtime* untuk komunikasi ke jaringan.
- Jaringan *wireless* dan Internet, protokol-protokol pendukung (HTTP, SOAP), XML.
- Web server yang bertindak sebagai *service producer*. Umumnya *web server* ini berhubungan dengan *database server* atau *server* aplikasi lainnya.

Versi dari WSA saat ini adalah versi 1.0 dan dispesifikasikan melalui JSR-172. WSA 1.0 hanya mendukung pengaksesan/pemanfaatan *Web service*. Hal ini berarti suatu *mobile device* Java ME dapat menjadi *service consumer*, tetapi bukan *service producer*, dan tidak mendukung pembuatan dan *deployment service endpoint*. WSA juga tidak menentukan *Application Programming Interface* (API) untuk *service discovery* menggunakan UDDI (Rendon, 2005).

3. PENGEMBANGAN SISTEM

Teknologi yang digunakan untuk implementasi *Web service* adalah J2EE 1.4, dimana teknologi *Web service*-nya menggunakan JAX-RPC 1.1. Aplikasi pada sisi *client* yang akan digunakan sebagai contoh untuk mengakses *Web service* yang dibuat, merupakan sebuah aplikasi *mobile* yang diimplementasikan menggunakan teknologi Java ME dan WSA. *Mobile device* yang digunakan adalah telepon seluler yang mendukung MIDP versi 2.0 dan WSA versi 1.0. Arsitektur sistem dapat dilihat pada Gambar 2. Metode yang digunakan untuk mengembangkan sistem ini mengikuti langkah-langkah pada *System Development Life Cycle* (SDLC), yaitu Analisis Kebutuhan Sistem, Desain Sistem, Implementasi Sistem, dan Pengujian Sistem. Untuk pemodelannya digunakan pemodelan berorientasi obyek menggunakan *Unified Modeling Language* (UML).



Gambar 2. Arsitektur Sistem

Cara kerja sistem dapat dijelaskan sebagai berikut. User menjalankan aplikasi *client* pada *mobile device*, aplikasi *client* kemudian akan memanggil metode pada *Web* service yang sesuai dengan permintaan user. *Web service* mengembalikan data yang diminta, dan aplikasi *client* akan menampilkan data tersebut pada user. Aplikasi *client* dapat mangkonsumsi atau memanggil metode dari Web service setelah memperoleh *file* WSDL dari *Web service*. Informasi dari WSDL ini digunakan oleh Stub Generator untuk membuat stub. Aplikasi kemudian memanggil stub untuk melakukan koneksi dan memanggil metode pada *Web service*.

Model penggunaan *Web service* di sini adalah RPC (*Remote Procedure Call*), dimana metode pada *Web service* dipanggil seperti memanggil suatu prosedur. Saat aplikasi client memanggil metode pada *Web service*, aplikasi *client* mengirimkan parameter yang dibutuhkan oleh metode yang dipanggil dalam bentuk SOAP Request (dalam format XML). Aplikasi *server* kemudian akan mengirimkan data kembali ke *client* dalam suatu SOAP Response (dalam format XML). Aplikasi *client* akan melakukan pemrosesan yang dibutuhkan terhadap SOAP Response ini (menggunakan JAXP) dan menampilkan hasilnya pada user.

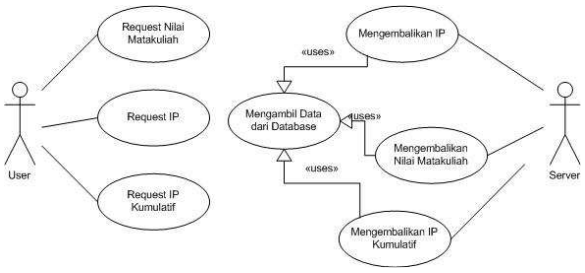
3.1. SPESIFIKASI SISTEM

Sistem ini mempunyai batasan dan spesifikasi sebagai berikut :

- Informasi yang diberikan dibatasi pada informasi nilai suatu matakuliah, indeks prestasi semester, dan indeks prestasi kumulatif.
- Aplikasi pada sisi *client* menggunakan Java ME dan WSA.
- Aplikasi sisi *server* menggunakan teknologi J2EE 1.4.
- *Server* yang digunakan adalah Sun Java Application Server (SJAS).
- *Database* dibuat menggunakan MySQL

3.2. DESAIN DAN IMPLEMENTASI SISTEM

Diagram *use case* dari sistem dapat dilihat pada Gambar 3.



Gambar 3. Diagram *Use Case*

3.2.1. PERANCANGAN *WEB SERVICE*

Web service yang dirancang mempunyai metode-metode sebagai berikut:

Tabel 1. Metode pada *Web Service*

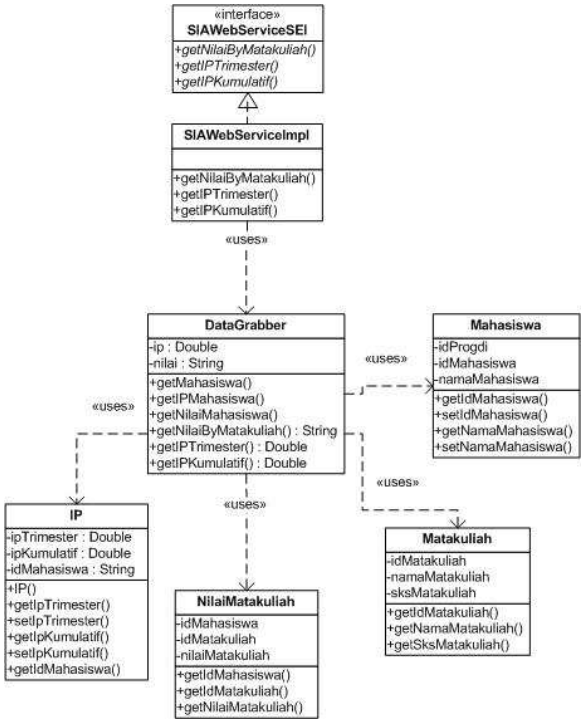
Metode	Deskripsi
`getNilaiByMatakuliah(String idMahasiswa, String idMatakuliah)`	Mengembalikan nilai untuk suatu matakuliah
`getIPTrimester(String idMahasiswa)`	Mendapatkan Indeks Prestasi Semester
`getIPKumulatif(String idMahasiswa)`	Mendapatkan Indeks Prestasi Kumulatif

Berikut ini merupakan potongan kode untuk metode `getNilaiByMatakuliah(String idMahasiswa, String idMatakuliah)`

```
public String getNilaiByMatakuliah(String idMahasiswa, String idMatakuliah)
throws java.rmi.RemoteException {

DataGrabber dg = new DataGrabber();
String str = dg.getNilaiByMatakuliah(idMahasiswa, idMatakuliah);
return str;
}
```

Aplikasi pada sisi *server* terdiri dari enam kelas dan satu *interface*. Diagram kelas dari aplikasi sisi *server* ditunjukkan pada Gambar 4. Diagram kelas yang ditunjukkan pada gambar hanya memuat atribut dan metode yang digunakan oleh aplikasi, jadi tidak semua atribut dan metode ditunjukkan.



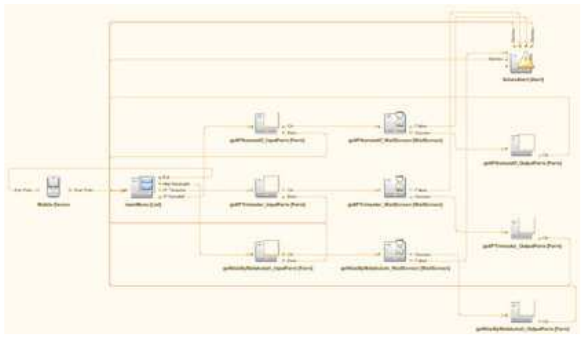
Gambar 4. Diagram Kelas Aplikasi Sisi *Server*

3.2.2. PERANCANGAN APLIKASI *CLIENT*

Aplikasi *client* terdiri dari tujuh kelas dan satu *interface* sebagai berikut:

- `GetIPKumulatif`
- `GetIPKumulatifResponse`
- `GetIPTrimester`
- `GetIPTrimesterResponse`
- `GetNilaiByMatakuliah`
- `GetNilaiByMatakuliahResponse`
- `SIAWebServiceSEI (interface)`
- `SIAWebServiceSEI_Stub`
- `SIASATWebServiceMIDlet`

Enam kelas yang pertama merupakan kelas pendukung untuk pemanggilan metode dan memproses hasil dari *Web service*. *Interface* `SIAWebServiceSEI` merupakan *interface* berisi metode-metode yang akan diimplementasikan oleh `SIAWebService_Stub` untuk melakukan pemanggilan metode pada *Web service*. `SIAWebService_Stub` merupakan stub yang dihasilkan oleh Stub Generator. Kelas utama dalam aplikasi *client*, adalah `SIASATWebServiceMIDlet`. Kelas inilah yang membuat *instance* dari stub dan kemudian menggunakan *instance* dari stub untuk melakukan pemanggilan metode pada *Web service*. *Flow diagram* dari aplikasi *client* ditunjukkan pada Gambar 5.



Gambar 5. *Flow Diagram* Aplikasi *Client*

3.3. PENGUJIAN SISTEM

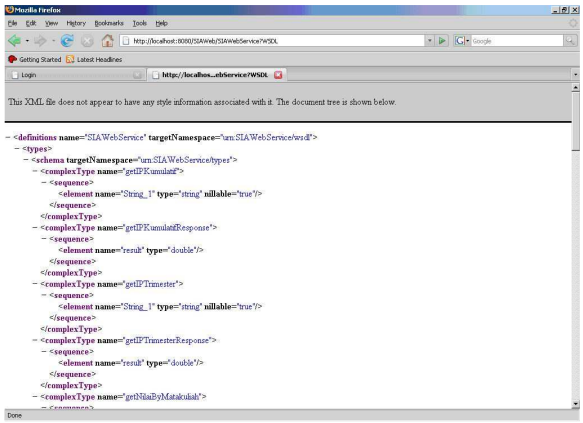
Pengujian sistem ini dilakukan dengan cara:

- Melakukan pengambilan file WSDL melalui browser pada komputer.
- Pemanggilan metode pada Web service dengan aplikasi selain aplikasi client Java ME, yaitu StrikeIron Web Service Analyzer (aplikasi client pada komputer)
- Pemanggilan metode pada Web service dangan aplikasi client Java ME.

Hasil dari pengujian menggunakan aplikasi pada komputer dan aplikasi pada *mobile device* akan dibandingkan apakah menghasilkan *response* yang sama atau tidak.

4. HASIL DAN PEMBAHASAN

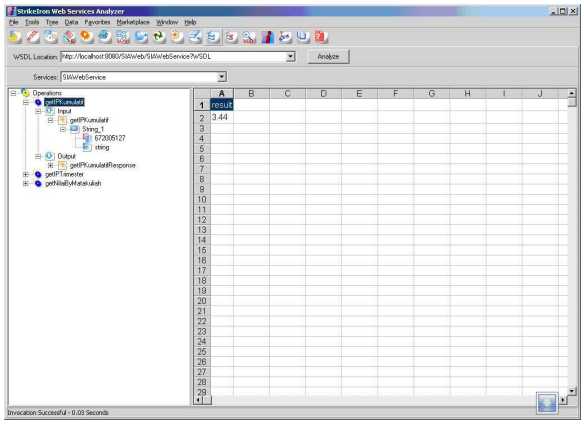
Hasil pengujian pengambilan *file* WSDL ditunjukkan pada Gambar 6.



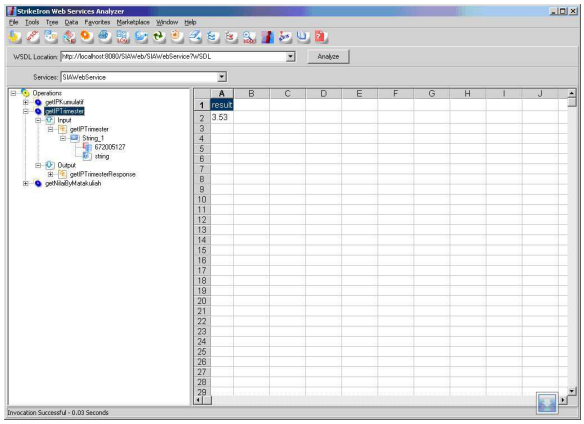
Gambar 6. File WSDL dari *Web Service* ditampilkan dengan Mozilla Firefox

Pengujian pemanggilan metode pada *Web service* menggunakan aplikasi biasa pada komputer ditunjukkan pada Gambar 7, 8, dan 9. SOAP Request dan SOAP Response dalam pemanggilan metode `getNilaiByMatakuliah()` dapat dilihat pada Gambar 10 dan 11.

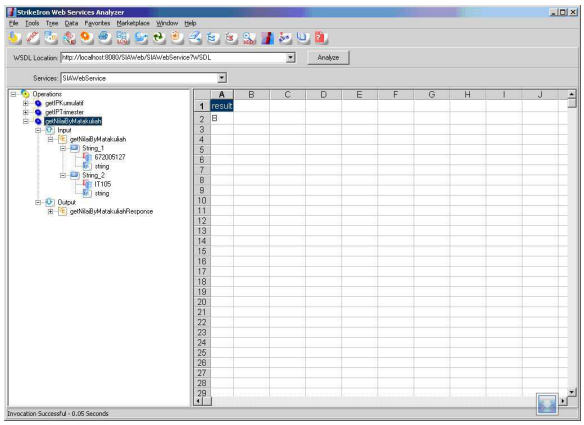
ISSN: 1979-2328



Gambar 7. Hasil Pemanggilan `getIPKumulatif()` menggunakan StrikeIron Web Service Analyzer



Gambar 8. Hasil Pemanggilan `getIPTrimester()` menggunakan StrikeIron Web Service Analyzer

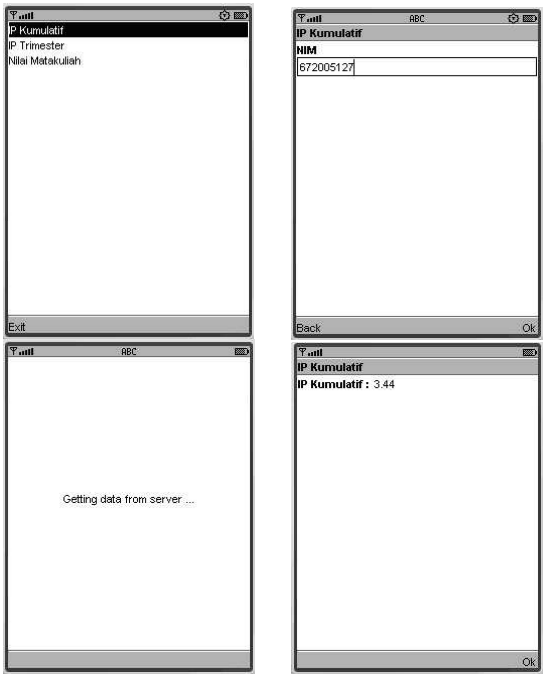


Gambar 9. Hasil Pemanggilan `getNilaiByMatakuliah()` menggunakan StrikeIron Web Service Analyzer

Gambar 10. SOAP Request `getNilaiByMatakuliah()`

Gambar 11. SOAP Response `getNilaiByMatakuliah()`

Pengujian pemanggilan metode `getIPKumulatif()` pada *Web service* dari *client* Java ME ditunjukkan pada Gambar 12.



Gambar 12. Hasil dari `getIPKumulatif()` oleh Aplikasi *Client* Java ME

Hasil pengujian Web service ditunjukkan pada Tabel 2.

Tabel 2. Hasil Pengujian *Web Service*

Aplikasi	Metode yang dipanggil	Hasil
StrikeIron Web Service Analyzer	`getNilaiByMatakuliah(672005127,IT105)`	B
	`getIPTrimester(672005127)`	3.53
	`getIPKumulatif(672005127)`	3.44
Client (Java ME)	`getNilaiByMatakuliah(672005127,IT105)`	B
	`getIPTrimester(672005127)`	3.53
	`getIPKumulatif(672005127)`	3.44

Berdasarkan hasil pengujian terhadap *Web service* yang dibuat dan aplikasi *client*, dapat dilihat bahwa *Web service* yang dibuat dapat berjalan dengan baik dan dapat diakses oleh aplikasi *client*, baik aplikasi *client* pada komputer (StrikeIron Web Service Analyzer) maupun aplikasi *client* pada *mobile device* yang dibuat menggunakan Java ME dan WSA. Hasil yang diperoleh dari pengujian menggunakan aplikasi *client* pada komputer dan aplikasi *client* pada *mobile device* mengembalikan hasil yang sama untuk pemanggilan metode pada *Web service* (dengan parameter yang sama).

5. KESIMPULAN

Web services saat ini merupakan salah satu teknologi yang digunakan untuk integrasi aplikasi terdistribusi dengan aplikasi *mobile*. Dalam tulisan ini telah ditunjukkan bagaimana merancang dan mengimplementasikan suatu *prototype* aplikasi *mobile* yang menggunakan Java ME dan WSA untuk mengakses/mengkonsumsi *Web service*, dengan contoh *Web service* pada Sistem Informasi Akademik. Dengan WSA yang menyediakan *programming interface* untuk pengaksesan *Web service* (JAX-RPC subset) dan *parsing* XML (JAXP subset) pada Java ME *platform*, maka pengembang aplikasi yang ingin membuat aplikasi *mobile* yang mengakses/mengkonsumsi *Web service* tidak perlu lagi melakukan hal-hal yang bersifat *low-level*, seperti manipulasi SOAP, HTTP, dan pemetaan tipe data antara tipe data WSDL dan tipe data pada Java, karena semuanya telah ditangani oleh WSA. Selanjutnya dapat dikembangkan aplikasi-aplikasi mobile yang mengkonsumsi *Web services* untuk tujuan tertentu, misalnya melihat stok barang, informasi cuaca, berita, dan sebagainya.

6. DAFTAR PUSTAKA

Ellis, John dan Young, Mark, 2003, *J2ME Web Service 1.0*, Available from: http://www.jcp.org.

Ortis, Enrique, 2004, *Introduction to J2ME Web Services*, Available from: http://developers.sun.com/techtopics/mobility/apis/articles/wsa/index.html.

Rendon, Oscar, 2005, *Architectures for Web Services Access from Mobile Devices*, IEEE Proceedings of the Third American Web Congress.

Stencil Group, *Defining Web Services*, Available from: http://whitepapers.sdnet.co.ok/0,39025945,60020104p-39000468q,00.htm

Sun Microsystems, 2001, *Java 2 Platform, Micro Edition Data Sheet*, Available from: http://java.sun.com/j2me/j2me-ds.pdf.

Sun Microsystems, 2004, *Java 2 Platform, Micro Edition Web Services*, Santa Clara, California, USA.

Sun Microsystems, 2001, *Mobile Information Device Profile Data Sheet*, Available from: http://java.sun.com/products/midp/midp-ds.pdf.

Utomo, Wiranto Herry, 2005a, *Prototyping Sistem Informasi Eksekutif Perguruan Tinggi Bidang Akademik menggunakan Web Service*, Aiti Jurnal Teknologi Informasi Vol. 2 No. 2, Fakultas Teknologi Informasi Universitas Kristen Satya Wacana, Salatiga, Agustus 2005.

Utomo, Wiranto Herry, 2005b, *Prototyping Web Service untuk E-Service (Studi Kasus: Rental Software Sistem Informasi Akademik)*, PAKAR Jurnal Teknologi Informasi dan Bisnis Vol. 6, No. 2, Fakultas Teknologi Informasi, Universitas Teknologi Yogyakarta, Agustus 2005.

World World Wide Consortium, *Web Service*, Available from: http://www.w3c.org