Dekomposisi Web Service pada Enterprise Resource Planning Pondok Pesantren

Fian Risdia Wulan[1)], Muhammad Ainul Yaqin[2)], Syahiduz Zaman[3)]
[1,2,3]Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang
Jalan Gajayana 50 Malang, 65144 Telepon (0341) 551354, Faksimile (0341) 572533
[1]13650090@student.uin-malang.ac.id , [2]yaqinov@ti.uin-malang.ac.id, [3]syahid@ti.uin-malang.ac.id

Abstrak

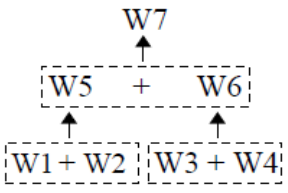
Web Service merupakan sistem software yang saling berinteraksi untuk tujuan pertukaran data melalui jaringan. Salah satu aplikasi yang menggunakan teknologi web service adalah Enterprise Resource Planning Pondok Pesantren. Web service yang digunakan dalam penelitian ini merupakan web service pada ERP Pondok Pesantren yang dibangun dalam proyek pengembangan ERP Pondok Pesantren. Web service yang dibangun merupakan web service komposit, yaitu web service yang memiliki beberapa method atau function. Web service komposit kurang mencerminkan operasi yang dimilikinya, sehingga sulit untuk menemukan web service yang memiliki fungsi spesifik. Dengan demikian, web service perlu disedehanakan melalui proses dekomposisi. Dekomposisi web service merupakan suatu cara memecah web service komposit menjadi atomik. Web service atomik memiliki satu method atau function. Masing-masing operation akan disimpan dan dijadikan function. Setiap function yang terbentuk akan dicari tipedata outputnya untuk selanjutnya di generate menjadi WSDL baru. Hasil dari dekomposisi web service adalah WSDL web service atomik yang memiliki satu operation. Dengan demikian, web service atomik lebih spesifik terhadap operasi yang digunakan.

Kata kunci : *dekomposisi, web service, wsdl*

1. Pendahuluan

Menurut W3C [1] *web service* adalah sebuah sistem *software* yang dirancang untuk mendukung interaksi dari mesin-ke-mesin melalui jaringan. *Web service* menyediakan standar untuk melakukan interaksi antar aplikasi yang berbeda yang berjalan pada berbagai *platform* dan atau *framework.* Sistem lain berinteraksi dengan *web service* melalui cara yang telah ditentukan dan di deskripsikan menggunakan pesan SOAP, biasanya disampaikan menggunakan HTTP dengan serialisasi XML bersama dengan standar *web* lainnya**.**

Elemen-elemen yang terdapat dalam WSDL antara lain, *types, message, porttype, binding,* dan *service. Types* mendefinisikan tipedata yang digunakan dalam *web service. Message* mendeskripsikan pesan yang dikomunikasikan. *Porttype* mendeskripsikan operasi-operasi yang dapat dijalankan oleh *web service. Binding* mendefinisikan bagaimana pesan ditransmisikan. *Service* mendeskripsikan alamat *web service* [2]. *Web service* dapat dibedakan menjadi dua macam, yaitu *web service composit* dan *web service* tunggal (atomik). *Web sevice composit* merupakan *web service* yang dibangun dari layanan-layanan *web service* tunggal [3]. Berikut adalah gambaran *web service composite* yang dibangun menggunakan layanan-layanan *web service* tunggal :



Gambar 1. *Web Service* Komposit

Pada Gambar 1 terlihat bahwa *web service* W1 dan W2 bergabung membentuk *web service* W5. *Web service* W3 dan W4 bergabung membentuk *web service* W6. Dan *web service* W5 dan W6 bergabung membetuk *web service* W7. Sehingga dapat disimpulkan bahwa *web service* W1, W2, W3, dan W4 merupakan *web service* tunggal, dan *web service* W5, W6, dan W7 merupakan *web service composite* karena terbentuk dari gabungan beberapa *web service* baik *web service* tunggal ataupun *web service composite* [3].

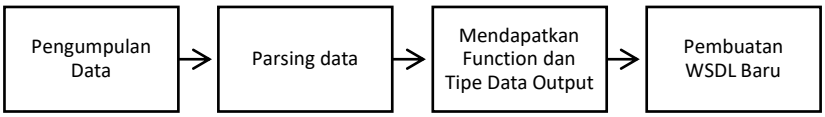
Dekomposisi web service merupakan proses untuk mengurai suatu web service komposit menjadi web service atomik. Tujuan dari dekomposisi web service ini adalah untuk mengurangi kompleksitas waktu pencocokan dan mengurangi penggunaan sumberdaya waktu mengkonsumsinya. Michiaki Tatsubori dan Kenichi Takahashi [4] menjelaskan penggunan framework yang bernama H2W yang dapat digunakan untuk membangun *web*

service wrapper yang ada, *multi-paged web application.* Penelitian yang dilakukan mengusulkan model dekomposisi berdasarkan *page-transition* dan model abstraksi akses halaman dengan propagasi konteks. Dengan dekomposisi dan abstraksi yang diusulkan, *developer* dapat membangun *web service wrapper* dengan fleksibel. Junichi Tatemura dan Wang Pin Hsiun [5] menggambarkan arsitektur layanan dekomposisi melalui komputasi tepi untuk meningkatkan *cacheability* layanan web *e-commerce*. Dengan arsitektur tersebut diusulkan layanan *cache* yang mendekomposisi konten *web service* menjadi objek. S. Lagraa, *et.al* [6] menyajikan dua dekomposisi berdasarkan metode *web service matchmaking.* Keuntungan utama mendekomposisi *graph web service* menjadi sub-struktur yang lebih kecil adalah untuk mengurangi kompleksitas waktu pencocokan. Algoritma yang diusulkan memperhitungkan karakteristik utama dari *graph web service : directed edges* dan menggunakan *graph matching tool* yang efisien. Pada *structural matching* ditambahkan perhitungan *semantic similarity* untuk meningkatkan presisi pencocokan.

Dalam penelitian ini *web service* yang digunakan merupakan *web service* yang bangun pada ERP Pondok Pesantren dalam proyek pengembangan ERP Pondok Pesantren [7] [8] [9] [10] [11] [12] [13]. *Web service* pada ERP Pondok Pesantren merupakan *web service* komposit karena memiliki beberapa *function/ method. . Web service* komposit kurang mencerminkan operasi yang dimilikinya, sehingga sulit untuk menemukan web service yang memiliki fungsi spesifik. Dengan demikian, *web service* perlu di sedehanakan melalui proses dekomposisi. Proses dekomposisi bertujuan untuk menyederhanakan *web service* agar *web service* dapat diketahui fungsinya dengan mudah. Dekomposisi *web service* dilakukan dengan mencari *function* dengan mengolah atribut *operation name* WSDL dan mencari dengan tipe data *output web service* berdasarkan atribut *type* dalam elemen *message. Function* dan tipe data yang telah didapat digunakan untuk *generate* WSDL baru. Masing-masing *function* di *generate* menjadi satu WSDL, sehingga setiap WSDL akan memiliki satu buah elemen *operation.*

2. Metode Penelitian

Data yang digunakan dalam penelitian ini berasal dari WSDL *web service* Pondok Pesantren yang telah dibangun pada penelitian sebelumnya. Berikut adalah blok diagram proses dekomposisi web service:



Gambar 2. Blok Diagram Metode Penelitian

2.1 Pengumpulan Data

Data yang digunakan diambil dan diolah adalah dokumen WSDL. Dokumen WSDL memuat deskripsi dan informasi yang ada di dalam *web service.* Dokumen WSDL diperoleh dengan menambahkan "?wsdl" pada URL *web service.* Dokumen WSDL yang telah didapatkan kemudian disimpan dalam file berekstensi xml. Hal ini diperlukan agar dokumen WSDL dapat dikalukan proses *parsing* untuk mengambil elemen-elemen dan atribut-atribut yang dibutuhkan.

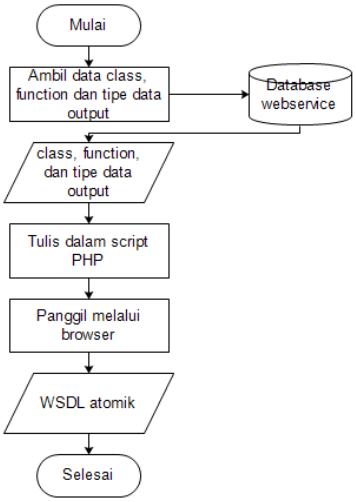
```
<definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:server"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="urn:server">
<types>
<xsd:schema targetNamespace="urn:server">
 <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
 <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
 <xsd:complexType name="getMapel">
....
```

Gambar 3. Dokumen WSDL

Tabel 1. Sumber Data dan URL WSDL *Web Service*

No.	Nama *Web Service*	URL WSDL
1.	*Web Service* Kurikulum	http://192.168.1.105/ws/_kurikulum_anik/mapel.php?wsdl
2.	*Web Service* Kegiatan	http://192.168.1.105/ws/_kegiatan_havit/kegiatan_services.php?wsdl

3.	*Web Service* Akuntansi	http://192.168.1.105/ws/_akuntansi_aziz/akuntansi.php?wsdl
4.	*Web Service* Kepegawaian	http://192.168.1.105/ws/_kepegawaian_ubhai/pegawai.php?wsdl
5.	*Web Service* Kesantrian	http://192.168.1.105/ws/_kesantrian_udin/santriParameter.php?wsdl
6.	*Web Service* Keuangan	http://192.168.1.105/ws/_keuangan_om/keuangan/keuangan_execute.php?wsdl
7.	*Web Service E-Document*	http://192.168.1.105/ws/_edocument_nurika/insert_bidang_jabatan.php?wsdl
8.	*Web Service* Perencanaan Produksi	http://192.168.1.105/ws/_pp_pipit/sipp.php?wsdl
9.	*Web Service* Pengadaan	http://192.168.1.105/ws/_pengadaan_dewi/pengadaan.php?wsdl



Gambar 4. *Parsing* Data

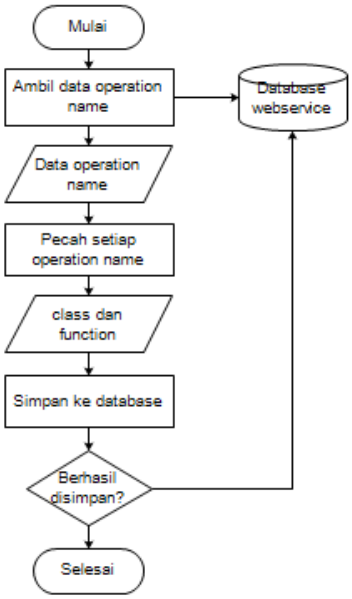
2.2 Parsing Data

Tahap *parsing* merupakan proses pemilahan elemen-elemen yang terdapat pada WSDL untuk selanjutnya dilakukan pengolahan data. Pada penelitin ini elemen-elemen yang ada di *filter* dan diambil elemen *message*, *porttype*, dan *operation*. Proses *parsing* dalam penelitian ini menggunakan ekstensi PHP yaitu *simpleXML* untuk mendapatkan data dari file xml. *SimpleXML* merupakan *parser* berbasis pohon yang sesuai dengan struktur dokumen WSDL.

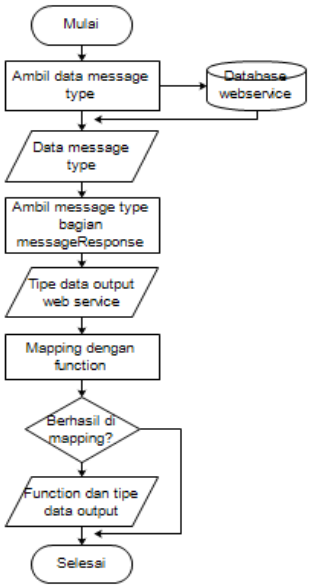
Sebelum elemen *message*, *portType*, dan *operation* disimpan ke dalam *database*, dilakukan pengecekan terhadap kelengkapan elemen pada dokumen WSDL terlebih dahulu. Proses pengecekan ini memastikan bahwa dokumen WSDL yang diinputkan memiliki elemen-elemen yang lengkap. Dalam penelitian ini, WSDL dianggap lengkap jika memiliki elemen *types*, *message*, *portType*, *binding*, dan *service*.

2.3 Mendapatkan *Function* dan Tipe Data *Output Web Service*

WSDL di *generate* otomatis menggunakan NuSOAP berdasarkan nama *function dan* tipe data *output web service* yang dihasilkan. Dokumen WSDL mendeskripsikan suatu *web service*. Elemen *operation* WSDL mengandung informasi tentang *class* dan *function-function web service* seperti pada Gambar 7, sedangkan elemen *message* mengandung informasi tentang tipe data untuk pesan *input* dan *output* yang digunakan *web service*. Proses dekomposisi didasarkan pada hubungan antara fungsi/method dengan tipe data output yang digunakan. Proses untuk mendapatkan *class* dan *function* dapat dilihat pada Gambar 5. Proses untuk mendapatkan tipe data *output* dapat dilihat pada Gambar 6.



Gambar 5. *Flowchart* Mendapatkan *Function*



Gambar 6. *Flowchart* Mendapatkan Tipe Data *Output*

Operation Name	
kegiatan.getTahunAjaran	
kegiatan	getTahunAjaran
Class	***Function***

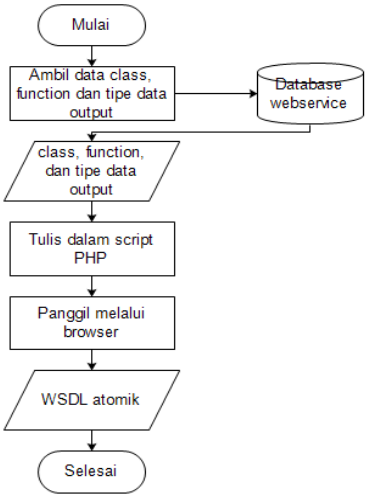
Gambar 7. Memecah *Operation Name* Menjadi *Class* dan *Function*

Message Name	***Message Type***
kegiatan.getTahunAjaranResponse	tns:TahunAjaranArray
Tipe Data Output	Array

Gambar 8. Mengambil Tipe Data dari *Message Type*

2.4 Penulisan WSDL Baru

Function dan tipe data *output* yang telah didapatkan pada proses sebelumnya digunakan sebagai input data untuk proses *generate* WSDL atomik. Setiap pasangan *function* dan tipe data *output* akan di *generate* WSDLnya. Penelitian ini menggunakan *library* NuSOAP untuk *generate* WSDL secara otomatis. *Function* dan tipe data *output* yang telah berpasangan kemudian dituliskan ke dalam script PHP untuk di *generate* menjadi WSDL baru dengan memanggil *method* `configureWSDL()` yang ada di dalam NuSOAP untuk menyediakan nama dan *namespace service*. Ketika URL ditambahkan "?wsdl", maka WSDL akan terbentuk secara otomatis. Proses penulisan WSDl baru dapat dilihat pada Gambar 9.



Gambar 9. *Flowchart* Penulisan WSDL Baru

3. Hasil dan Pembahasan

Hasil proses pengumpulan data berupa dokumen WSDL yang telah disimpan ke dalam format *.xml.* Penyimpanan data ke dalam format *xml* berfungsi untuk mempermudah proses *parsing.* Hasil dari proses *parsing* adalah data yang diambil dari atribut elemen *portType, operation,* dan *message.* Gambar 10 berisi elemen *portType* dan *operation* yang ada di dalam *web service* Kurikulum. Atribut yang disimpan berupa *PortType name, operation name, input message,* dan *output message*.

```
<portType name="Web Service kurikulum BY ANIKPortType">
 <operation name="kurikulum.getMapel">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.getMapelRequest"/>
  <output message="tns:kurikulum.getMapelResponse"/>
 </operation>
 <operation name="kurikulum.getMapelByDepartemen">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.getMapelByDepartemenRequest"/>
  <output message="tns:kurikulum.getMapelByDepartemenResponse"/>
 </operation>
 <operation name="kurikulum.getSilabus">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.getSilabusRequest"/>
  <output message="tns:kurikulum.getSilabusResponse"/>
 </operation>
 <operation name="kurikulum.getSilabusAll">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.getSilabusAllRequest"/>
  <output message="tns:kurikulum.getSilabusAllResponse"/>
 </operation>
 <operation name="kurikulum.getSks">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.getSksRequest"/>
  <output message="tns:kurikulum.getSksResponse"/>
 </operation>
 <operation name="kurikulum.rppbyidmapel">
  <documentation>Fetch array</documentation>
  <input message="tns:kurikulum.rppbyidmapelRequest"/>
  <output message="tns:kurikulum.rppbyidmapelResponse"/>
 </operation>
</portType>
```

Gambar 10. Elemen *PortType* dan *Operation Web Service* Kurikulum

Pada tahap mendapatkan *function* dan tipe data *output,* nama *class* yang *function* yang digunakan dalam *web service* diperoleh dari pengolahan data atribut *operation name.* Tipe data *output* diperoleh dari pengolahan data atribut *type messageResponse.* Hasil tahap ini dapat dilihat pada Tabel 2.

Tabel 2. Data *Function* untuk *Web Service Atomik*

No.	*Class*	*Function*	Tipe Data
1.	kurikulum	getMapel	array
2.	kurikulum	getMapelByDepartemen	array
3.	kurikulum	getSilabus	array
4.	kurikulum	getSilabusAll	array
5.	kurikulum	getSks	array
6.	kurikulum	rppbyidmapel	array

Tahap penulisan WSDL baru menghasilkan WSDL atomik menggunakan *generate* otomatis menggunakan *library* NuSOAP. WSDL atomik terbentuk sesuai dengan pemanggilan *function* yang dilakukan. *Script* untuk melakukan *generate* WSDL dapat dilihat pada Gambar 11.

```
<?php
include ("folder/file.php");
$SOAP = new class('nama class');

$SOAP->registerFunction('function',"output");
$SOAP->execute();
?>
```

Gambar 11. *Script* untuk *Generate* WSDL dengan NuSOAP

Hasil dekomposisi berupa WSDL atomik yang memiliki satu buah *operation.* WSDL atomik hasil *generate* mendeskripsikan *web service* atomik. Pada *web service* Kurikulum dapat dipecah menjadi enam *web service* atomik seperti pada Tabel 3.

Tabel 3. Hasil Dekomposisi *Web Service* Kurikulum

No.	Keterangan	*Operation Name*
1.	Web Service Mata Pelajaran	kurikulum.getMapel
2.	Web Service Mata Pelajaran Departemen	kurikulum.getMapelByDepartemen
3.	Web Service Silabus	kurikulum.getSilabus
4.	Web Service Silabus 2	kurikulum.getSilabusAll
5.	Web Service SKS	kurikulum.getSks
6.	Web Service Rencana Pembelajaran	kurikulum.rppbyidmapel

Array ([0] => Array ([id_pelajaran] => 1011 [nm_pelajaran] => Pendidikan Agama Islam [id_departemen] => 11) [1] => Array ([id_pelajaran] => 1012 [nm_pelajaran] => Bahasa Indonesia [id_departemen] => 11) [2] => Array ([id_pelajaran] => 1013 [nm_pelajaran] => Bahasa Inggris [id_departemen] => 11) [3] => Array ([id_pelajaran] => 1014 [nm_pelajaran] => Ilmu pengetahuan Alam [id_departemen] => 11) [4] => Array ([id_pelajaran] => 1015 [nm_pelajaran] => Matematika [id_departemen] => 11) [5] => Array ([id_pelajaran] => 1021 [nm_pelajaran] => Pendidikan Agama Islam [id_departemen] => 12) [6] => Array ([id_pelajaran] => 1022 [nm_pelajaran] => Bahasa Indonesia [id_departemen] => 11) [7] => Array ([id_pelajaran] => 1023 [nm_pelajaran] => Teori Graph [id_departemen] => 12) [8] => Array ([id_pelajaran] => 1027 [nm_pelajaran] => Bahasa Indonesia [id_departemen] => 12) [9] => Array ([id_pelajaran] => 1031 [nm_pelajaran] => Pendidikan Agama Islam [id_departemen] => 13) [10] => Array ([id_pelajaran] => 1032 [nm_pelajaran] => Bahasa Indonesia [id_departemen] => 11) [11] => Array ([id_pelajaran] => 1033 [nm_pelajaran] => Sistem Informasi Berorientasi Object [id_departemen] => 13) [12] => Array ([id_pelajaran] => 1036 [nm_pelajaran] => Riset Operasi [id_departemen] => 12) [13] => Array ([id_pelajaran] => 1037 [nm_pelajaran] => Sistem Berkas [id_departemen] => 13) [14] => Array ([id_pelajaran] => 1041 [nm_pelajaran] => Sistem Informasi Manajemen [id_departemen] => 14) [15] => Array ([id_pelajaran] => 1042 [nm_pelajaran] => Sistem Pendukung Keputusan [id_departemen] => 14) [16] => Array ([id_pelajaran] => 1043 [nm_pelajaran] => Sistem Berbasis Pengetahuan [id_departemen] => 14) [17] => Array ([id_pelajaran] => 1046 [nm_pelajaran] => Pendidikan Agama Islam [id_departemen] => 11) [18] => Array ([id_pelajaran] => 1048 [nm_pelajaran] => Sejarah [id_departemen] => 11) [19] => Array ([id_pelajaran] => 1049 [nm_pelajaran] => Kimia [id_departemen] => 11) [20] => Array ([id_pelajaran] => 1051 [nm_pelajaran] => coba e learning [id_departemen] => 11) [21] => Array ([id_pelajaran] => 1052 [nm_pelajaran] => fiqih [id_departemen] => 17))

Gambar 12. Hasil *Request Client* pada *Web Service* Mata Pelajaran

Hasil dekomposisi dilakukan pengujian menggunakan program *client* sederhana. Program *client* berfungsi untuk mengecek apakan fungsi yang ada di dalam *web service* bekerja atau tidak dengan memanfaatkan WSDL. C*lient* dapat mengakses file WSDL dengan mengakses URL yang berisi alamat diletakkannya file WSDL tersebut. Jika proses *request* berhasil, maka WSDL atomik hasil proses dekomposisi berfungsi dengan baik dapat digunakan kembali sesuai kebutuhan. Hasil *request client* dapat dilihat pada Gambar 2.12.

Proses dekomposisi *web service* membuat *web service* komposit menjadi lebih sederhana dengan menjadikannya sebagai *web service* atomik. *Web service* atomik memiliki WSDL yang memiliki informasi lebih spesifik tentang operasi yang digunakan karena hanya memiliki satu elemen *operation.*

4. Simpulan

Berdasarkan penelitian, dekomposisi *web service* komposit menjadi *web service* atomik berhasil dilakukan. Dekomposisi dilakukan dengan mengambil atribut *operation name* yang mengandung informasi *class* dan *function* atau *method*, serta mengambil atribut *message type* sebagai tipe data *output web service. Web service* atomik memiliki satu fungsi yang di deskripsikan di dalam satu elemen *operation* WSDL atomik. Proses dekomposisi meyederhanakan *web service* komposit menjadi *web service* atomik yang memiliki satu operasi, sehingga *web service* dapat diketahui fungsinya secara lebih spesifik.

Daftar Pustaka

[1] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris and D. Orchard, "Web Services Architecture," 11 February 2004. [Online]. Available: https://www.w3.org/TR/ws-arch/. [Accessed 2017].

[2] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana, "W3C," 2001. [Online]. Available: https://www.w3.org/TR/2001/NOTE-wsdl-20010315. [Accessed 14 October 2017].

[3] K.-K. Lau and C. Tran, "Composite Web Service," *Emerging Web Services Technology*, vol. 2, pp. 77-95, 2008.

[4] M. Tatsubori and K. Takahashi, "Decomposition and Abstraction of Web Applications for Web Service Extraction and Composition," in *Web Services, 2006. ICWS'06. International Conference on*, IEEE, 2006, pp. 859-868.

[5] J. Tatemura and W.-P. Hsiung, "Web service decomposition: Edge computing architecture for cache-friendly e-commerce applications," *Electronic Commerce Research and Applications*, vol. 5, pp. 57-65, 2006.

[6] S.Lagraa, H.Seba, R.Khennoufaand and H.Kheddouci, "A graph decomposition approach to web service matchmaking," in *7th International Conference on Web Information Systems and Technologies (WEBIST)*, 2011, pp. 31-40.

[7] A. Ma'rifah, "Integrasi Sistem Informasi Kurikulum Pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[8] F. M. Alfaidah, "Integrasi Sistem Informasi Perencanaan Produksi pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[9] A. Fajar, "Integrasi Sistem Informasi Akuntansi pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[10] M. F. Pratama, "Integrasi Sistem Informasi Keuangan pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[11] M. E. Suprianto, "Integrasi Sistem Informasi Akademik pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[12] B. Syah, "Integrasi Sistem Informasi Kesantrian pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[13] M. Ubaidillah, "Integrasi Sistem Informasi Kepegawaian pada Enterprise Resource Planning Pondok Pesantren Tipe D Menggunakan Service Oriented Architecture," Malang, 2017.

[14] K. Gottschalk, G. Graham, H. Kreger and J. Snell, "Introduction to Web Service Architecture," *IBM SYSTEMS JOURNAL*, vol. 41 , pp. 170-177, 2002.