Model Interoperabilitas Web Service Feeder PDDIKTI Menggunakan Enterprise Javabeans (EJB) dan REST-API

Febianto Arifien[1], Sutarno[2] dan Marti Riastuti[2]
[(1)]Universitas Gunadarma
Jl. Margonda Raya No. 100, Depok, Jawa Barat 16424
[(2)]Sistem Informasi, STMIK Jakarta STI&K
Jl. BRI No.17, Radio Dalam Kebayoran Baru Jakarta Selatan 12140
febianto@staff.gunadarma.ac.id, {p4kt4rno, tutimarti67}@gmail.com

ABSTRAK

Penggunaan aplikasi Feeder PDDIKTI yang diwajibkan bagi setiap perguruan tinggi baik negeri maupun swasta membutuhkan proses interoperabilitas dengan sistem informasi yang sudah berjalan di lingkungan perguruan tinggi.Data aktivitas akademik yang banyak dan bervariasi menjadikan kendala untuk dapat dikirim melalui sistem aplikasi yang telah disediakan oleh DIKTI. Aplikasi Feeder PDDIKTI menggunakan web service SOAP + WSDL dengan PHP. Proses pengiriman data tersebut harus melalui beberapa mekanisme proses integrasi atribut-atribut data, proses mapping, proses insert, update, delete dan yang terakhir proses syncronisasi.Penelitian ini memanfaatkan teknologi penggunaan REpresentational State Transfer – Application Programing Interface (REST-API) dan didukung oleh Enterprise Java Bean (EJB) yaitu teknologi untuk mengembangkan komponen di sisi server yang scalable , transactional dan secure untuk aplikasi enterprise.Hasil dari penelitian ini berupa model interoperabilitas antar dua sistem web service yaitu metode yang telah disediakan Feeder PDDIKTI dengan sistem informasi menggunakan JAVA EJB dengan teknologi REST-API. Berdasarkan hasil percobaan diperoleh hasil bahwa pada aplikasi web sevice yang dikembangkan berhasil menghilangkan masalah 'maximum exeed' pada aplikasi SOAP XML, dapat melakukan fungsi GET dan POST ke aplikasi Feeder PDDIKTI tanpa ada notifikasi erorr.

Kata Kunci : *Interoperabilitas, web service, REST-API, JAVA EJB*

PENDAHULUAN

Berdasarkan Undang-undang Republik Indonesia Nomor 12 Tahun 2012 tentang Pendidikan Tinggi dan Permenristekdikti Nomor 61 Tahun 2016 tentang Pangkalan Data Pendidikan Tinggi mengenai kewajiban perguruan tinggi untuk melaporkan data dan informasi penyelenggaran pendidikan tinggi secara benar dan tepat pada Pangkalan Data Pendidikan Tinggi (PDDIKTI) maka dikeluarkan Surat Edaran Feeder PDDIKTI Nomor : 0543/E1.2/PL/2015 mengenai penggunaan aplikasi Feeder PDDIKTI.Aplikasi Feeder PDDIKTI merupakan aplikasi yang digunakan untuk mengelola data mahasiswa dan data perkuliahan tiap Perguruan Tinggi. Tujuan penggunaan Feeder PDDikti adalah sebagai media pengumpan yang dikelola mandiri oleh masing-masing perguruan tinggi untuk diinput dan disimpan yang kemudian dikirim sehingga dapat ditampilkan pada aplikasi Forlap pada laman http://forlap.dikti.go.id/. Proses pelaporan data akademik pada aplikasi Feeder PDDIKTI dilakukan secara berkala dua kalisetiap semester di awal semester dan di akhir semester[1].

Data akademik yang terdiri dari data mahasiswa, mata kuliah, aktivitas mengajar dosen, riwayat status mahasiswa, aktivitas kuliah mahasiswa dan nilai semesteryang sangat banyak menjadi kendala saat pengisian melalui aplikasi Feeder PDDIKTI. Untuk menanggulangi permasalahan yang timbul,pihak DIKTI membuat jembatan perantara untuk proses pengiriman data dari instansi terkait ke DIKTI menggunakan web service. Web service adalah sistem perangkat lunak yang dirancang untuk mendukung interaksi mesin-ke-mesin yang saling beroperasi melalui jaringan. Sistem ini memiliki antarmuka yang dijelaskan dengan format tertentu (WSDL) dengan menggunakan metode Simple Object Access Protocol (SOAP)[2]. Proses integrasi data didahului oleh proses mapping tabel database yang ada di perguruan tinggi dengan tabel yang dibuat oleh web service PDDKTI Feeder. Model integrasi lainnya yaitu menyesuaikan fungsi-fungsi yang

disediakan web service PDDKTI Feeder seperti getToken untuk mendapatkan token agar bisa login ke web service Feeder, getRecord untuk mendapatkan satu record data dari tabel, InsertRecord untuk memasukkan satu record data ke dalam tabel, UpdateRecord untuk mengubah satu record data yang ada pada satu tabel dan DeleteRecord untuk meghapus satu record data. Permasalahan lain timbul jika data yang diproses merupakan data berskala besar, yaitu terjadi error 'exceeds maximum', artinya data yang dikirim terlalu besar sehingga harus mengurangi jumlah file yang dikirim atau mengubah file konfigurasi bahasa pemrograman. Hal ini mengakibatkan waktu proses yang dibutuhkan lebih lama.

Penelitian yang berkaitan dengan interoperabilitas perangkat lunak terhadap web service Feeder PDDIKTI antara lain:dengan dikembangkannya file excel yang berisi data siswa (profil dan status), data mata pelajaran dan penilaian, data kuliah (kelas kuliah, siswa KRS, dosen, mata pelajaran, dan nilai siswa), dan kegiatan kuliah dari siswa, namun muncul permasalahan dimana diperlukan pembuatan file excel terlebih dahulu dan diperlukan penyesuaian atribut yang disesuaikan dengan format web service Feeder PDDIKTI [3]. R. I. Perwira dan B. Santosa[4] mengusulkan memanfaatkan teknologi web service yang memiliki keunggulan dapat bekerja multi platfom sistem operasi. Modul web service yang diimplementasikan terdiri atas insert, update dan delete yang merupakan pengembangan replika sebelumnya sehingga pelaporan data tetap dapat dilakukan meskipun berbeda sistem. Hasil dari penelitian ini berupa sebuah tools web service yang terdiri tiga modul utama yang dapat mengakomodir kebutuhan-kebutuhan pelaporan data akademik ke DIKTI yang sudah berjalan. Tools ini telah digunakan dalam dua tahun terakhir untuk keperluan pelaporan, namun kembali masih memerlukan modifikasi script dari pengisian id satuan pendidikan (id_sp) sesuai dengan perguruan tinggi yang dituju, begitu juga dengan id wilayah (id_wil) dan kewarganegaraan. Pada Tahun 2012, F. Kapojos, H. F. Wowor, a M. Rumagit,dan a P. R. Wowor [5] mengusulkan sebuah model web service. Model ini menggunakan Service Oriented Architecture (SOA) yang juga dapat diimplementasikan sebagai aplikasi, untuk dapat diakses oleh aplikasi lain dengan menggunakan XML sebagai format pengiriman pesan[6], membuat suatu interface Web Service yang menerapkan SOA untuk diintegrasikan dengan aplikasi informasi akademik, namun dalam pengiriman data menggunakan web service SOA tidak maksimal untuk data berskala besar sering kali terjadi error 'exceeds maximum', artinya data yang dikirim terlalu besar sehingga harus mengurangi jumlah file yang akan dikirim atau mengubah file konfigurasinya.

Berdasarkan pada permasalahan yang masih timbul, perlu adanya solusi alternatif yang dapat mengintegrasi antara kebutuhan aplikasi Feeder PDDIKTI dengan sistem perguruan tinggi supaya berjalan secara efektif. Berkaitan dengan model interoperabilitas, saat ini dikenal teknologi Application Programing Interface (REST-API) menggunakan Enterprise Java Bean (EJB) sebagai solusi interoperabilitas berinteraksi antar dua atau lebih web service[6]. Tujuan Penelitian ini membangun interface dengan memanfaatkan teknologi penggunaan REpresentational State Transfer – Application Programing Interface (REST-API) dan didukung oleh Enterprise Java Bean (EJB) untuk mengintegrasi interoperabilitas web service Feeder PDDIKTI.

Interoperabilitas

Masalah interoperabilitas muncul ketika dua atau lebih sistem yang tidak kompatibel dimasukkan dalam hubungan.Secara umum, ketidakcocokan ini terkait dengan lapisan interoperabilitas. Misalnya, dua perusahaan dapat memiliki gaya manajemen yang berbeda, yang dapat menyebabkan masalah terkait dengan lapisan organisasi. Perusahaan juga dapat menggunakan konsep dan representasi yang berbeda untuk mengekspresikan makna yang sama, apa yang dapat menyebabkan masalah terkait dengan lapisan semantik[7].

Beberapa masalah dlm interoperabilitas seperti pada tabel 1.

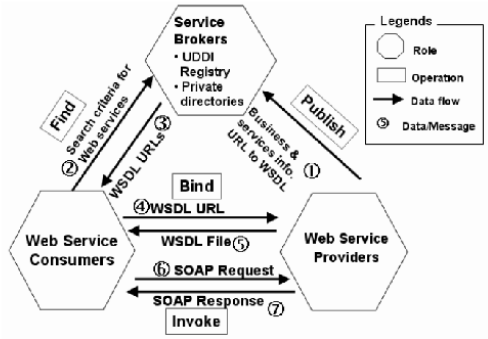
Tabel 1. *Contoh hambatan Interoperabilitas yang terkait dengan masalah perusahaan.*

		Interoperability barriers and layers		
		Conceptual (Syntactic and Semantic)	*Technological*	*Organisational (and Legal)*
Interoperability concerns	*Business*	- visions, strategies, cultures, understanding	- IT infrastructure	- organisation structures - legislations - business rules
	Process	- syntax and semantics of processes	- process interfaces and supporting tools	- procedures of work, - processes organisation
	Service	- semantics to name and describe services	- interface, architecture	- responsibility /authority to manage services
	Data	- data representation and semantics - data restriction rule	- data exchange formats	- responsibility/ authority to add/delete, change/ update data

(Sumber: G. da Silva Serapião Leal, W. Guédria, and H. Panetto, 2019)

Web Service

Web service adalah salah satu bentuk sistem perangkat lunak yang dirancang untuk mendukung interaksi antar mesin melalui jaringan.Web service memiliki antarmuka yang dijelaskan dalam format yang dapat dibaca oleh mesin. Sistem lain berinteraksi dengan web service menggunakan pesan SOAP umumnya dikirim melalui HTTP dibentuk XML. Arsitektur web servicesecara umum dapat dilihat pada gambar 1[4].



Gambar 1. *Arsitektur Web Service*
(Sumber: Rifki Indra Perwira, Budi Santosa, 2017)

Pada gambar 1, ada tiga besar peran dalam arsitektur web Service:

[1] . Penyedia layanan / Service Provider
Service provider adalah penyedia layanan web. Penyedia layanan mengimplementasikan layanan dan membuatnya tersedia di Internet.

[2] . Layanan Pemohon / Service Request
Ini adalah konsumen dari web Service. Peminta layanan yang mencari dan menemukanlayanan yang dibutuhkan serta menggunakan layanan tersebut.

[3] . Layanan registri / Service Registry
Berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/service yangtelah di-register[4].

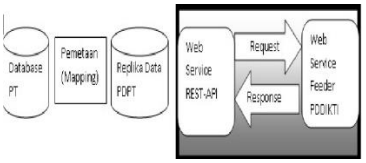
Lapisan dasar blok bangunan web service menyediakan fasilitas komunikasi jarak jauh antara dua aplikasi yang merupakan layerarsitektur web service.
Layer 1 : protokol internet standar yang digunakan sebagai sarana transportasi adalahHTTP dan TCP/IP.
Layer 2 : Simple Object Access Protocol (SOAP) berbasiskan XML dan digunakanuntuk pertukaran informasi antar sekelompok layanan.
Layer 3 : Web service Definition Language (WSDL) digunakan untuk mendiskripsikanattribute layanan.
Layer 4 : Universal Description Discovery and Integration, yang mana merupakandirektori pusat untuk deskripsi layanan.

Java Beans

Enterprise JavaBeans (EJB) merupakan salah satu teknologi Java EE penting untuk pengembangan sisi server komponen.EJB menggabungkan logika bisnis terdistribusi, transaksional, aman, dan portabel berbasis aplikasi berbasis Java EE. Namun, mereka relatifsederhana untuk dikembangkan dan digunakan[8].

METODE PENELITIAN

Gambaran penelitian yang difokuskan pada pembuatan web service REST-API yang dapat berinteraksi dengan web service Feeder PDDIKTI, dapat dijelaskan pada gambar 2.

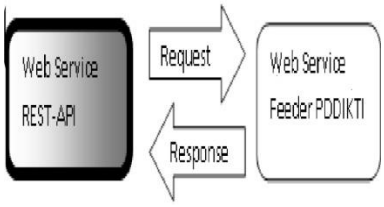


Gambar 2. *Gambaran Penelitian*

Keterangan gambaran penelitian adalah sebagai berikut:
1. Database PT : data mentah yang

berasal dari data internal perguruan tinggi.
2. Mapping : proses mapping data atau penyesuaian data dengan replika dataPDDIKTI (PDPT).
3. Replika Data PDPT : data yang sudah disesuaikan berdasarkan proses mapping.
4. Web Service REST-API : tahapan untuk membuat rest-api menggunakan Java Beans.
5. Web Service Feeder PDDIKTI : tahapan ini melakukan insert, update, detete data-data yang telah digabungsebelumnya untuk dapat dikirim ke aplikasi feeder.
6. Request: proses permintaan data dengan fungsi-fungsi sesuai format web service Feeder PDDIKTI.
7. Response: hasil keluaran yang didapat berupa data atau pernyataan error. Bagian ini bisa dibagi menjadi beberapa sub bab, tetapi tidak perlu mencantumkan penomorannya.

Langkah-langkah dalam tahapan penelitian difokuskan pada pembuatan web service REST-API dengan terlebih dahulu telah melakukan proses mapping sesuai dengan data yang terlampir pada Feeder PDDIKTI.



Gambar 3. *Fokus Penelitian*

Aplikasi Web Service yang dibuat menggunakan tools dan teknologi Java EE (Enterprise Edition) berbasis JDK (Java Development Kit) dengan Maven untuk membuat Java Project dengan bantuan Eclipse IDE, selanjutnya di deploy ke Wildfly Server. Adapun langkah-langkah pembuatan aplikasi adalah sebagai berikut:

Langkah-langkah Pembuatan Aplikasi:

Gambar 4. *Langkah-langkah pembuatan aplikasi*

Penjelasan pembuatan aplikasi adalah sebagai berikut:

1. Tahap Pembuatan Proyek Web Menggunakan Maven Wildfly Server Archentype

Langkah pertama yang dilakukan adalah membuat Java Project menggunakan tools Maven Project dengan fitur Wildfly Server Arhentype.Tujuan tahap adalah untuk menghasilkan project template yang sesuai dengan konfigurasi server yang digunakan yaitu Server Wildfly.

2. Tahap Penulisan Script pada Java Package yang terdiri dari:

- Modul INTF
  Modul ini merupakan kelas objek yang digunakan untuk mendefinisikan hasil pemetaan data feeder. Kelas objek akan ditanslasi ke bentuk database dalam modul EJB, dalam modul ini diperlukan model interface sebagai perantara proses bisnis dalam mengolah data yang dilakukan oleh modul EJB .
- Modul EJB
  Modul ini merupakan Bussines Logic Layer, atau modul ini dibuat untukproses bisnis aplikasi seperti data persistance untuk menghubungkan data antar aplikasi dan mengatur relasi data dalam ORM (Object Relation Management). Modul EJB mengimplemantasi interface pada Modul INTF untuk manajemen data.
- Modul EAR
  Modul EAR (Enterprise Application aRchive) merupakan

standar modul arsip java yang dibuat untuk menampung paket modul lain seperti file berformat jar yang disusun menjadi satu arsip file java untuk di deploy ke server.

- Modul REST
  Modul REST merupakan layanan berdasarkan arsitektur RESTful Java dengan JAX-RS yang kerjanya mendisain antarmuka dengan mengirim perintah HTTP di JAX-RS.

3. Tahap Instalasi atau compile komponen

Tahapan ini adalah membuat directori baru bernama target yang berisi file hasil kompilasi berupa .jar, .war dan .ear. Proses ini dilakukan dengan mengeksekusi program dengan perintah mvn install. Tujuan tahap ini adalah untuk membuat file eksekusi siapdi deploy ke server java.

4. Tahap Deploy ke Server Wildfly

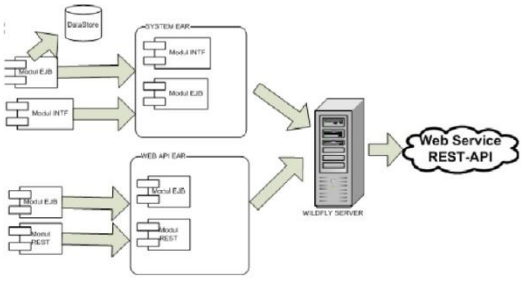
Tahapan ini merupakan proses mendeploy file .ear ke server wildfly. Proses ini dapat dilakuakan dengan beberapa cara:

a. Mengcopy file .ear ke system file dan melakukan konfigurasi pada file standalone.

b. Menggunakan antarmuka manajemen (konsol admin atau CLI)

c. Menggunakan Maven untuk mendeploy aplikasi ke server.

Gambar 5 merupakan skema diagram komponen aplikasi REST-API secara global:

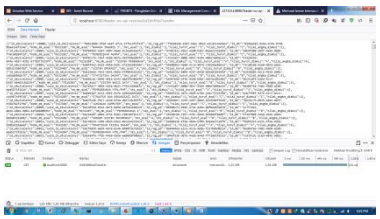


Gambar 5. *Diagram Komponen Aplikasi REST-API*

Berdasarkan gambar diagram komponen, dapat dijelaskan sesuai dengan langkah-langkah pembuatan aplikasi, dibangun komponen modul INTF, modul EJB dan modul EAR. Langkah-langkah tersebut dilakukan dua kali, yang pertama membangun modul SYSTEM EAR, berfungsi sebagai data akses dan yang kedua membangun WEB API EAR, berfungsi sebagai bisnis akses.Modul EJB pada SYSTEM EAR terkoneksi datastore yaitugudang penyimpan data untuk mengelola koleksi data yang terkoneksi menggunakan JDBC Driver dengan fungsi-fungsi SQL.Modul EJB pada WEB API EAR mengimplementasi teknologi DTO yaitu kumpulan data berdasarkan dari Model Class yang tidak memiliki validasi, tidak ada logika bisnis, tidak ada logika apapun. Ini sangat sederhana, ringan dan hanya dijadikan wadah untuk memindahkan data.Fungsinya dijadikan sebagai wadah untuk memindahkan data dari DAL (data access layer) ke BAL (bussiness access layer) atau antara lapisan dimasing-masing dalam arsitektur n-tier.

HASIL DAN PEMBAHASAN

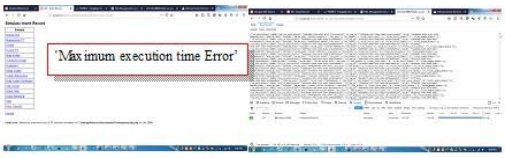
Setelah implementasi tahap kompilasi dihasilkan file eksekusi dengan ekstensi ear, selanjutnya dapat diakses melalui alamat url melalui browser, seperti pada gambar 6.



Gambar 6. *Hasil Implementasi Tahap Kompilasi*

Dari gambar 6 tampak tampilan data berformat JSON yang dipanggil menggunakan fungsi GET, sebagai bentuk umum dari hasil keluaran dari web service. Data JSON digunakan sebagai data yang akan di ‘push’ menggunakan fungsi POST untuk menambah atau mengubah database. Kelebihan web service REST-API ini, data dapat dipanggil langsung dari web browser tidak perlu membuat atau menambah tools tambahan seperti dalam bentuk SOAP atau nuSOAP.

Berikut ditampilkan perbandingan proses fungsi GET untuk mengambil data dalam bentuk JSON antara implementasi langsung fungsi XML pada SOAP Feeder PDDIKTI dengan aplikasi web service Pengembangan REST-API JavaBeans.
Sebagai sample data Feeder yang akan dipakai adalah data pada tabel nilai transfer karena proses pelaporan data feeder harus sesuai dengan periode tertentu atau semester yang berlaku, apabila tidak sesuai dengan periodenya data akan dikunci atau tidak diperkenankan menambah atau mengubah data.Akan tetapi data nilai transfer mahasiswa relatif lebih sederhana, artinya atributnya tidak banyak dan hanya memiliki relasi ke tabel data mahasiswa saja, sehingga bisa dijadikan sample percobaan.



(a) tampilan menggunakan SOAP XML (b) tampilan Pengembangan REST-API

Gambar 7. *Tampilan fungsi GET untuk mengambil data*

Berdasarkan hasil pada gambar 7(a), data tidak bisa terbaca dengan notifikasi error 'Maximum execution time', proses request yang dilakukan memakan waktu terlalu lama sehingga otomatis program dihentikan. Berbeda dengan apabila menggunakan aplikasi pengembangan REST-API, data dapat diakses atau dapat tampil seperti yang terlihat pada gambar 7(b).

Selanjutnya dilakukan percobaan 'push' ke database Feeder menggunakan fungsi POST, bertujuan untuk menambah atau mengedit, seperti yang terlihat pada gambar 8.

(a) tampilan menggunakan SOAP XML (b) tampilan Pengembangan REST-API

Gambar 8. *Tampilan Proses Push ke Feeder*

Terlihat pada gambar 9, bahwa proses POST untuk mengubah atau mengedut data berhasil dilakukan tanpa notifikasi error baik dari sisi penggunaan SOAP maupun dengan pengembangan REST-API.

Berdasarkan proses-proses GET dan POST untuk mencoba interaksi dengan database, maka dibuat tabel yang menggambarkan efektivitas proses-proses tersebut dengan bantuan tools dari web browser console Mozilla Firefox, seperti terlihat pada tabel 2.

Tabel 2. *Tabel perbandingan kecepatan dan ukuran file yang diakses*

No	Proses	Metode	Kecepatan	Ukuran File	Gambar
1	Data akses 3066 record dengan SOAP XML	GET	'maximum exceed error'	'maximum exceed error'	8(a)
2	Data akses 3066 record dengan REST-API	GET	1,42 detik	1,01 MB	7
3	Data entry 100 record Dengan SOAP XML	POST	11,81 detik	110,11 KB	9(a)
4	Data entry 100 record Dengan REST-API	POST	21,2 detik	16 B	9(b)

Berdasarkan perbandingan tabel diatas, dapat terlihat bahwa pengembangan REST-API cukup cepat tanpa notifikasi error.

PENUTUP

Berdasarkan hasil pengujian, pada aplikasi web sevice yang dikembangkan berhasil menghilangkan masalah 'maximum exeed' pada aplikasi SOAP XML, dapat melakukan fungsi GET dan POST ke aplikasi Feeder PDDIKTI tanpa ada notifikasi erorr. Perbandingan kecepatan dan bobot file yang diakses, ternyata web service pengembangan REST-API lebih cepat dan bobotnya lebih kecil. Dengan demikian dapat disimpulkan bahwa pembangunan interface dengan mengembangkan web service REST-API berupa tahapan penyisipan script pada paket-paket komponen terbukti berhasil mengintegrasikan interoperabilitas secara lebih efektif.

DAFTAR PUSTAKA

[1] . Taufiqurrochman, R. E. Indrajit, and M. Fauzi, "Penerapan Business Intelligent Dalam Pengambilan Keputusan Akademik Yang Tepat Untuk Perguruan Tinggi, Dengan Memanfaatkan Aplikasi Feeder PDDIKTI (Studi Kasus Pada Universitas Muhammadiyah

Jakarta),” in Seminar Nasional Sains dan Teknologi 2017, 2017, pp. 1–5.

[2] . R. G. Côté, “Web Service,” in Encyclopedia of Systems Biology, New York, NY: Springer New York, 2013, pp. 2351–2351.

[3] . V. H. Pranatawijaya, “Pengembangan Perangkat Lunak Generate File Untuk Migrasi Data EPSBED Ke Format Table Feeder PDDIKTI,” J. SAINTEKOM, vol. 6, no. 2, p. 1, Mar. 2017.

[4] . R. I. Perwira and B. Santosa, “Implementasi Web Service Pada Integrasi Data Akademik Dengan Replika Pangkalan Data Dikti,” Telematika, vol. 14, no. 1, pp. 1–11, 2017.

[5] . F. Kapojos, H. F. Wowor, a M. Rumagit, and a P. R. Wowor, “Implementasi Service-Oriented Architecture dengan Web Service untuk Aplikasi Informasi Akademik,” J. Fak. Tek. UNSRAT, vol. 1, no. 1, pp. 1–5, 2012.

[6] . K. Baker, R. Baker, W. Baur, J. Bulmahn, E. Greenwood, T. Hitchcock, J. Jacobs, N. Logue, F. Mentzer, E. Mona, C. Pramas, S. K. Reynolds, F. W. Schneider, M. A. Stackpole, L. Stevens, J. L. Sutter, R. Beisner, J. Fares, W. Mahy, B. Sola, F. Solhan, B. Wootten, L. Bonner, D. Team, L. Bonner, and J. Compton, “RESTful Java with JAX RS 2.0,” 2014 IEEE Int. Conf. Syst. Man, Cybern., pp. 4121233–4121235, 2014.

[7] . G. da Silva Serapião Leal, W. Guédria, and H. Panetto, “Interoperability assessment: A systematic literature review,” Computers in Industry, vol. 106. pp. 111–132, 2019.