BAB II
LANDASAN TEORI

2.1. Paradigma Arsitektur Perangkat Lunak

Sebelum memahami tentang arsitetur SOA maka akan dibahas *Arsitecture* dari perangkat lunak, Arsitektur perangkat lunak adalah sekumpulan pernyataan yang menggambarkan komponen perangkat lunak dan fungsi-fungsi yang ada pada komponen tersebut. Ini menggambarkan struktur teknis, batasan-batasan, ciri-ciri, serta antarmuka pada komponen-komponen. Arsitektur merupakan rancangan fisik sistem oleh karena itu membutuhkan rencana yang sangat matang pada proses pengembangannya.

Arsitektur perangkat lunak merupakan struktur sebuah sistem yang meliputi elemen perangkat lunak seperti sifat (*property*), relasi yang terlihat. Sifat yang terlihat misalnya fungsi apa saja yang disediakan oleh elemen, bagaimana kinerjanya, bagaimana penanganan kesalahannya, sumber daya apa saja yang digunakan (Krafzig dkk, 2004).

Menurut Erl (2009), ada tiga elemen yang saling berkaitan erat ketika berbicara tentang arsitektur perangkat lunak. Pertama adalah arsitektur teknologi, yaitu desain fisik dari suatu perangkat lunak. Kedua adalah infrastruktur teknologi, yaitu lingkungan pendukung yang termasuk di dalamnya perangkat keras dan perangkat lunak. Ketiga adalah perangkat lunak itu sendiri.

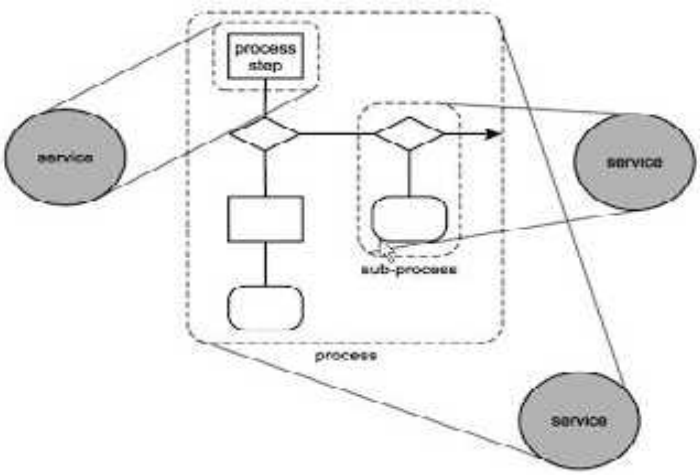
2.2. *Service Oriented Achitecture* (SOA)

Untuk memahami pengertian dari SOA, sebelumnya akan dibahas tentang *Service. Service* dalam lingkup SOA adalah sekumpulan dari fingsi-fingsi prosedur atau proses yang akan memberikan respon jika diminta oleh *client.* SOA adalah bentuk teknologi atsitektur yang mengikuti prinsip-prinsip *Service-Orientation* (berorientasi pada layanan), dimana konsep ini melakukan pendekatan dengan membagi proses bisnis yang besar menjadi sekumpulan *service* yang sederhana yang bertujuan untuk menylesaikan masalah tertentu. SOA tidak terikat pada teknologi tertentu, akan tetapi lebih kearah pendekatan untuk membangun perangkat lunak yang modular(Erl,2005).

2.2.1. Konsep *Service Oriented*

Setelah dijelaskan diatas, *Service-Oreiented* merupakan sebuah pendekatan yang dalam penyelesainnya membagikan proses bisnis yang besar menjadi bagian-bagian (*service*) kecil yang lebih spesifik. Istilah seperti ini sudah cukup banyak digunakan untuk beberapa masalah. Contoh dari permasalahannya adalah peminjaman buku dipustaka. Misalkan pengunjung pustaka akan mendaftar sebagai anggota perpustakaan, kemudian mencari buku yang akan dibaca, selanjutnya memberikan beberapa buku yang akan dipinjam kepada petugas, untuk jangka waktu anggota akan melakukan pengembalian. Jika dilakukan dengan pendekatan *Service-Oriented* maka akan akan dibagi-bagi menjadi pendaftaran anggota, pengecekan buku, peminjaman, dan pengembalian buku, dan pengecekan batas tanggal pengembalian.

Service sendiri dapat dipandang sebagai engkapsulasi logika dari suatu kumpulan aktivitas yang disusun kembali menjadi step-step dari proses bisnis. Hal seperti ini tentunya disesuaikan dengan kebutuhan. Bila diumpamakan proses peminjaman diperpustakaan diatas maka pada tahap pengembalian dapat dikomposisi lagi menjadi pengecekan anggota, batas pengembalian dan pengecekan denda, dan status buku. Dari semua kemungkinan itu harus dapat diselesaikan dengan memungkinkan seluruh *service* bisa berpatisipasi delam sebiat orkestrasi. Oleh sebab itu ada beberapa permasalahan yang dimiliki oleh *service* yaitu bagaimana *service* berhubungan, didesain dan bagaimana *service* pesan antar *service* didefinisikan(Erl,2008).



Gambar 2.1. Enkapsulasi *bussiness process* dengan *service*(Erl,2008)

2.3. Prinsip Berorientasi Layanan

Ada beberapa prinsip umum yang dipaparkan pada berorientasi layanan sebagai berikut: (Erl,2005)

1. *Reusability*, logika dibagi menjadi beberapa *service* yang mudah digunakan kembali (*reuse*)
2. *Contract. Service* harus mempunyai kontak atau kesepakatan bagaimana cara berhubungan dengan yang lain.
3. *loose coupling*. setiap *service* berdiri sendiri secara independen tidak bergantung dengan yang lain, sehingga harus dikonsepkan secara matang agar bisa digunakan kembali dan mempunyai mekanisme yang baik dengan yang lain.
4. *Service abstraction*. Layanan membungkus logika (*enkapsulasi*). Bagian yang tampak dari luar hanyalah kontrak layanan. Logika serta implementasi di balik itu tidak kelihatan dan tidak penting bagi peminta layanan.
5. *Composability*. Beberapa *service* dapat disusun untuk membentuk *service* baru. Sehingga memungkinkan logika dapat mewakili atau mempromosikan *service* pada level *astration*
6. *Aautonomy*. Layanan bersifat mandiri. Pengembangan layanan dapat dilakukan terpisah. Layanan memiliki kendali penuh terhadap logika yang dimilikinya.
7. *Statelessness*. Layanan tidak didesain untuk menyimpan informasi keadaan tertentu (*state*).
8. *Discoverability*. Layanan harus dapat ditemukan. Deskripsi layanan harus dapat dicari dan dimengerti oleh peminta layanan.

2.4. Komponen Arsitektur Berorientasi Layanan

Seperti yang dipaparkan oleh (Erl,2008) SOA terdiri dari sekumpulan *service*, namun dari sekumpulan itu tidak cukup untuk membentuk sebuat arsitektur. Sebuah *arsitectur* mempunyai empat komponen yaitu :

1. Pesan (*messages*) yaitu data yang dibutuhkan untuk menyelesaikan suatu sebagian atau unit kerja yang diberi dan diambil antara satu *service* dengan yang lain.

2. Operasi (*operation*s) yaitu fungsi-fungsi yang dimiliki oleh *service* untuk memproses pesan sehingga mendapatkan apa yang dibutuhkan, fungsi-fungsi ini yang nanti akan menuntaskan sebuah unit kerja.
3. Layanan (*service*) yaitu sekumpulan operasi yang dapat melakukan tugas tertentu. Dimana *service* akan mempresentasikan sekumpulan dari operasi yang berhubungan.
4. Proses (*processes*) yaitu aturan bisnis (*role*) yang menentukan operasi layanan apa yang digunakan untuk menyelesaikan suatu tujuan tertentu.

2.5. Layanan Web (*Web Service*)

Menurut (Erl,2005) *web service* merupakan sistem pertukaran informasi berbasis XML yang menggunakan internet untuk interaksi antar aplikasi. Teknologi ini merupakan standar yang diadopsi oleh banayak vendor perangkat lunak, karena standar yang terbuka. Dengan standar ini yang membuat aplikasi *web service* dapat diimplementasikan oleh vendor yang berbeda sehingga dapat berkomunikasi satu dangan yang lainya. Dengan ektensi ini membuat *web service* berkembang dengan cepat dan luas, karena fungsi-fungsi *web service* yang bisa digunakan dimana saja.

Web service mengacu pada sebuah aplikasi perangkat lunak yang dikembangkan dengan konsep khusus yaitu dengan standar pemograman komunikasi berbasis internet protocol. Tujuannya adalah untuk memungkinkan aplikasi atau komponen untuk memanggil fungsi dan pertukaran data. Ada tiga standar yang membentuk teknologi *web service* yaitu (Huang,2004)

1. *Simple Object Access Protocol* (SOAP), digunakan untuk memanggil fungsi dan prosedur pada *remote* sistem dan mempertukarkan data antar proses
2. *Web Service Description Language* (WSDL), untuk menggambarkan layanan secara spesifik pada komponen aplikasi dan bentuk *interface* layanan secara detail.
3. *Universal Description, Discovery, And Integration* (UDDI), untuk memunculkan bentuk pencarian *web service* yang sudah siap untuk digunakan dan dipublikasi sehingga *service* bisa digunakan dan dicari pada saat dibutuhkan.

Apa yang membuat *web service* unik dibandingkan dengan teknologi perangkat lunak yang lain dan sangat menarik untuk perusahaan, dikarenakan *web service* dapat mengurangi kompleksitas (pengurangan biaya). untuk aplikasi yang heterogen dan terintegrasi terlepas dari lokasi penggunaan, bahasa pemograman dan platform yang dipakai. Jadi perusahaan bisa berinofasi membuat *service* yang sebelumnya telah ada (*Lloosely-Coupled*), terdistribusi, dan mempunyai layanan yang terintegrasi. Ada beberapa standar yang dimiliki oleh *web service* sehingga menjadi unik yaitu:

1. Teknologi *web service* dapat diteima secara luas dan standar public-domain internet seperti HTTP dan XML.
2. Untuk pertama kalinya semua vendor teknologi menyepakati secara umum format *massage* yaitu SOAP untuk berkomunikasi antar aplikasi.

2.5.1. *Simple Object Access Protocol (*SOAP*)*

(Mabrouk,2008) menerangkan bahwa SOAP merupakan protokol yang berbasiskan XML. Protokol ini memberikan kemudahan kepada aplikasi untuk bertukar informasi menggunakan protokol *transportasi* seperti HTTP. Beberapa keuntungan yang membuat SOAP sangat direkomendasikan dan telah menjadi protokol komunikasi yang paling banyak digunakan. Keuntungan menggunakan *web service* dengan SOAP adalah sebagai berikut:

1. SOAP tidak tergantung pada *platform* manapun (*platform independent*).
2. SOAP dapat digunakan dalam jaringan internet.
3. Pesan SOAP dapat dibaca manusia, terstruktur, serta berbasis teks karena berupa XML

SOAP dapat dikatakan sebagai gabungan antara HTTP dengan XML karena SOAP umumnya menggunakan protokol HTTP sebagai sarana trans*port*asi data dan yang akan dipertukarkan ditulis dengan format XML. Karena SOAP menggunakan HTTP dan XML, maka SOAP memungkinkan pihak-pihak yang berbeda dapat saling mempertukarkan datanya (Priyambodo,2005).

(Erl,2005) menjelaskan bahwa tujuan utama SOAP adalah untuk memberi definisi format pesan yang standar. Struktur pesan ini cukup sederhana, tetapi

dapat diperluas dan disesuaikan dengan kebutuhan. Setiap pesan SOAP dipaket ke dalam suatu wadah yang disebut ***Envelope.***

Header **berisi** ***Header block***. ***Header block*** melengkapi pesan dengan beragam informasi yang mungkin dibutuhkan oleh *web service* . Penggunaan ***Header block*** memberikan kemungkinan pesan dapat menampung informasi tambahan yang banyak dan bervariasi. Beberapa contoh fitur yang dapat ditambahkan dengan penggunaan ***Header block*** yaitu:

1. Instruksi pemrosesan yang akan dieksekusi oleh entitas terentu.
2. Informasi *routing*
3. Mekanisme keamanan (*security*)
4. Pemrosesan transaksi (*transaction*)

Isi pesan yang sesungguhnya ditampung dalam ***body,*** yang biasanya berupa data dalam format XML. Isi *body* ini sering disebut sebagai ***message payload***. Bagian *body* ini juga dapat berisi elemen ***fault,*** yang digunakan untuk menyampaikan informasi ketika terjadi suatu kesalahan (*exception*).

Pada suatu pesan SOAP, elemen paling luar adalah <*Envelope*>. Dalam elemen tersebut terdapat elemen opsional <*Header* > dan elemen wajib <*Body*>. Ada dua atribut penting yang menentukan bentuk pesan SOAP serta tipe data yang digunakan untuk merepresentasikan data, yaitu atribut *style* yang digunakan oleh elemen **SOAP:*binding*** serta atribut *use* yang ada di elemen **SOAP:*body***. Kedua elemen ini berada pada elemen **SOAP:*binding*** suatu dokumen WSDL.
Atribut *style* pada elemen soap:*binding* dapat bernilai dokumen atau *rpc*. Dokumen mendukung keseluruhan dokumen XML dimasukkan ke dalam ***body*** SOAP, sementara ***rpc*** lebih mencerminkan komunikasi rpc tradisional yang mendukung tipe data parameter. Atribut *use* pada elemen **soap:*body*** dapat bernilai literal atau *encoded*. SOAP memiliki tipe datanya sendiri untuk merepresentasikan isi *body*. Kemudian, dikeluarkan dukungan untuk tipe data XSD. Atribut ini menunjukkan tipe data mana yang akan digunakan. Penggunaan literal menunjukkan bahwa tipe data XSD akan digunakan. Kombinasi *style* dan *use* yang mungkin adalah sebagai berikut:

1. ***Style*** **RPC** dan ***use encoded*** (disingkat ***rpc/encoded***)
2. ***Style*** **RPC** dan ***use literal*** (disingkat ***rpc/literal***)

3. ***Style* dokumen** dan ***use encoded*** (disingkat **dokumen/*encoded***)

2.5.2. *Web Service Description Language* (WSDL).

Mabrouk (2008) memaparkan bahwa WSDL merupakan dokumen yang ditulis menggunakan XML untuk menggambarkan *web service*. WSDL menunjukkan lokasi dan operasi apa yang diberikan oleh layanan sehingga dapat digunakan oleh peminta layanan (*service consumer*). Suatu WSDL menggambarkan empat hal berikut:

1. Layanan yang disediakan oleh *web service*, seperti daftar operasi yang dapat digunakan serta atribut pesan.
2. Tipe data pesan, baik pesan masukan (*input*) maupun pesan keluaran (*output*).
3. Informasi *binding* untuk protokol trans*port*asi, seperti HTTP dan JMS.
4. Alamat layanan yang digunakan untuk memanggilnya.

WSDL merupakan sebuah bahasa berbasis XML yang digunakan untuk mendefinisikan *web service* dan menggambarkan cara untuk mengakses *web service* tersebut (Shohoud dalam Priyambodo,2005). Deskripsi WSDL mendefinisikan sebuah *service* sebagai kumpulan dari *port*, dimana tiap-tiap *port* didefinisikan secara abstrak sebagai *portType* yang mendukung sekumpulan operasi. Tiap-tiap operasi memproses sekumpulan pesan tertentu (Priyambodo, 2005). Berdasarkan tulisan yang berjudul “Introduction to WSDL”, WSDL telah menjadi rekomendasi W3C sejak 26 Juni 2007.

Erl (2005) menjelaskan bahwa WSDL menggambarkan titik hubungan sebuah penyedia layanan (*service* provider), yang juga dikenal dengan sebutan *service endpoint* atau *endpoint* saja. WSDL menyediakan definisi formal dari antarmuka endpoint, sehingga apapun yang akan berkomunikasi dengan layanan mengetahui dengan tepat bagaimana ia harus menyusun pesannya. WSDL juga memberikan lokasi fisik (alamat) sebuah layanan. Suatu WSDL dapat dibagi kedalam dua kategori yaitu deskripsi abstrak dan deskripsi konkit, seperti diperlihatkan pada gambar berikut.

Deskripsi abstrak memberikan informasi tentang antarmuka *web service* tanpa tergantung teknologi apapun yang digunakan. Dengan memisahkan informasi ini, integritas deskripsi layanan (*service* description) dapat dijaga

terhadap semua perubahan yang mungkin terjadi pada *platform* di mana layanan dijalankan. Ada tiga bagian utama dalam deskripsi abstrak, yaitu ***portType***, ***operation*** dan ***message***. Bagian *port*Type menjelaskan pesan (*message*) apa saja yang dapat diproses oleh layanan ke dalam sekumpulan fungsi yang disebut sebagai operasi (*operation*). Setiap operasi menunjukkan aksi tertentu yang dilakukan oleh layanan. Operasi terdiri dari sekumpulan pesan masukan (*input message*) dan pesan keluaran (*output message*). Pada WSDL versi 2.0, ***portType*** berganti nama menjadi ***interface*.**

Agar suatu *web service* dapat menjalankan prosesnya, definisi antarmuka abstrak perlu dihubungkan dengan suatu implementasi teknologi yang nyata, yakni protokol trans*port*asi. Hubungan ini didefinisikan dalam deskripsi konkrit suatu WSDL, yang terdiri dari ***binding, port*** dan ***service*.** *Binding* menggambarkan teknologi transportasi yang akan digunakan layanan untuk berkomunikasi. SOAP adalah bentuk yang paling umum, namun teknologi lain juga dapat digunakan. *Binding* dapat diterapkan pada keseluruhan antarmuka atau pada operasi tertentu. *Port* menggambarkan lokasi fisik di mana layanan dapat diakses dengan protokol tertentu. *Service* digunakan untuk menunjukkan sekumpulan *endpoint* yang saling berhubungan. Pada WSDL versi 2.0, ***port*** berganti nama menjadi ***endpoint.***

Pada suatu dokumen WSDL, elemen paling luar adalah <*definitions*>. Dalam elemen tersebut terdapat lima element utama yaitu:

1. Elemen <**type**> untuk mendefinisikan tipe data yang digunakan dalam pesan.
2. Elemen <***message***> untuk mendefinisikan format pesan, baik itu pesan masukan (*input*) atau pesan keluaran (*output*) suatu operasi.
3. Elemen <***portType***> untuk mendefinisikan sekumpulan operasi (***operation***). Tiap-tiap elemen <***operation***> mendefinisikan sebuah operasi serta pesan masukan yang dibutuhkan dan pesan keluaran yang diberikan oleh operasi tersebut.
4. Elemen <***binding***> untuk memetakan operasi-operasi dan pesan yang telah didefinisikan pada *portType* ke suatu protokol tertentu.

5. Elemen <***service*** > untuk mendefinisikan sekumpulan ***port*** yang saling berhubungan. Elemen <***port***> memetakan ***binding*** kelokasi sebuah ***web service*** .

2.5.3. *Universal Description, Discovery, And Integration* (UDDI)

UDDI adalah sebuah *service registry* bagi pengalokasian *web service*. UDDI mengkombinasikan SOAP dan WSDL untuk pembentukan sebuah *registry* API bagi pendaftaran dan pengenalan *service*. *Service registry* ini menyediakan sebuah area umum dimana sebuah organisasi dapat mengiklankan keberadaan mereka dan *service* yang mereka berikan (*web service*).

2.6. *Service Lifecycle*

Siklus hidup dari *service* merupakan pase mengembangkan yang memungkinkan kemampuan *service* dibentuk lebih akurat untuk dijadikan aplikasi. Ada tiga tahap yang harus diperhatikan oleh pengembang untuk siklus *web service* ini(dirvasula,2006).

1. *Requirements and Analysis.*
2. *Design and Development*
3. *IT Operations*

Gambar 2.2 Tiga tahapan *Service lifecycle* (dirvasula,2006)

2.7. Implementasi SOA

Untuk kehandalan SOA dan *web service* Liu(2004) melakukan penelitian dengan meimplementasikan pada lembaga terbesar di China dalam melakukan proses seleksi 40 ribu proposal penelitian dari 1.400 universitas dan *institute riset*

di China. Kebutuhan yang diperlukan oleh lembaga adalah proses pengumpulan data proposal dan penyeleksiannya yang dilakukan delm rentang waktu yang singkat dibandingkan lembaga-lembaga yang lain didunia. Adapun pemasalahannya adalah :

1. Puluhan ribu dokumen yang dikumpulkan menggunakan format dan *platform* yang berbeda.
2. Pemprosesan dokumen dilakukan oleh lembaga keuangan China, temasuk merubah format dan penyeleksiannya dalam waktu yang relative sangat singkat.

Solusi yang diusulkan adalah sebagai berikut:

1. Konsep *reuse* dan *Lloosely-Coupled* SOA diterapkan dengan membagi *server* lembaga keuangan ke berbagai *server* yang ada ditiap universitas. Setiap dokumen uang masuk dalam berbagai format diolah dan dipetakan dengan standar XML.
2. Setelah dokumen-dokumen tersebut diolah diberbagai *server* universitas maka dokumen yang sudah mengikat standar tersebut baru akan dikirim oleh *server* universitas ke lembaga keuangan.

Terdapat banyak bukti bahwa SOA mampu berhasil dalam menyelesaikan permasalahan diberbagai bidang seperti diatas implementasi SOA dibidang keuangan dan dibidang bisnis seperti yang implementasi perabot elektrik rumah tangga dengan SOA(Nakamura,2004) dimana produk rumah tangga cerdas juga bisa mengadopsi SOA sebagai perinsip pengembagan. Dimana ketika *service* sentric arcitectur (SCA) bisa membuat komputasi semua diarahkan ke home *server* dan membuat ketergantungan sangat tinggi kepada *home server*. Dan apabila *home server crash* maka akan berpengaruh dengan yang lainya. Oleh sebab itu SOA dengan prinsip *Lloosely-Coupled*-nya bisa mengatasi dengan membuat *service* yang berdiri sendiri dan harus bisa beradaptasi dengan yang lain.

2.8. *Unified Modelling Language* (UML)

Menurut Rumbaugh (1999), UML merupakan sustu bahasa (notasi) yang menggambarkan aspek-aspek pengembangan selama proses OOAD. UML

digunakan untuk menganalisa, menspesifikasi, dan men*design* suatu sistem perangkat lunak. UML terdiri dari 14 tipe diagram, beberapa diantaranya :

1) ***Use Case Diagram***

   Use case adalah langkah-langkah terdiri dari tindakan yang akan timbul ketika seorang menggunakan sistem untuk menyelesaikan suatu proses. *Use case diagram* menggambarkan *use case* tersebut beserta relasi antara aktor dengan sistem. *Use case diagram* adalah menganalisa perilaku dari sistem atau pun sebsistem dari sudut pandang pengguna yang menggambarkan interaksi antara actor dan penguna.

2) ***Activity Diagram***

   Sebuah diagram yang menunjukan aktivitas-aktivitas dan tindakan-tindakan untuk menggambarkan proses-proses kerja ke dalam suatu gambar .

3) ***Sequence Diagram***

   Sebuah interaksi yang menujukan pesan antara objek-objek dan keles-kelas yang saling berinteraksi. *Sequence diagram* dapat digambarkan pada detail level yang berbeda dan juga untuk mecapai tujuan yang berbeda pada tahap-tahap siklus hidup pengembangan perangkat lunak

4) ***Class Diagram***

   Sebuah struktur UML yang menunjukkan kelas-kelas dengan atribut-atribut dan operasi-operasi serta kelas tersebut dengan kelas lainnya

2.9. Profil Organisasi

2.9.1. Dewan Dakwah Indonesia (DDI)

Pada 26 Februari 1967, atas undangan pengurus masjid Al-Munawarah, Kampung Bali, Tanah Abang, Jakarta Pusat, para alim ulama dan zu'ama berkumpul untuk bermusyawarah, membahas, meneliti, dan menilai beberapa masalah, terutama yang rapat hubungannya dengan usaha pembangunan umat, juga tentang usaha mempertahankan aqidah didalam kesimpangsiuran kekuatan-kekuatan yang ada dalam masyarakat. Dari musyawarah itu terbentuklah dewan dakwah indonesia (DDI) dan seiring perkembangannya dibentuk juga beberapa

pengurus cabang di beberapa wilayah di Indonesia salah satunya di kota Pekanbaru.

Dewan Dakwah yang dikukuhkan keberadaannya melalui Akte Notaris Syahrim Abdul Manan No.4, tertanggal 09 Mei 1967, melandaskan kebijaksanaannya kepada empat hal:

1. Yayasan Dewan Dakwah Islamiyah Indonesia berdasarkan taqwa dan keridhaan Allah.
2. Dalam mencapai maksud dan tujuannya, Dewan Dakwah mengadakan kerja sama yang erat dengan badan-badan dakwah yang telah ada di seluruh Indonesia.
3. Dalam hal yang bersifat kontroversial (saling bertentangan) dan dalam usaha melicinkan jalan dakwah, Dewan Dakwah bersikap menghindari dan atau mengurangi pertikaian faham antara pendukung dakwah, istimewa dalam melaksanakan tugas dakwah.
4. Dimana perlu dan dalam keadaan mengizinkan, Dewan Dakwah dapat tampil mengisi kekosongan, antara lain menciptakan suatu usaha berbentuk atau bersifat dakwah, usaha mana sebelumnya belum pernah diadakan, seperti mengadakan pilot projek dalam bidang dakwah.

Musyawarah Alim Ulama juga merumuskan program kerja sebagai penjabaran dari landasan kebijaksanaan di atas. Program kerja tiga pasal itu ialah sebagai berikut:

1. Mengadakan pelatihan-pelatihan atau membantu mengadakan pelatihan bagi muballighin dan calon-calon muballighin.
2. Mengadakan *research* (penelitian) atau membantu mengadakan penelitian, yang hasilnya dapat segera diinanfaatkan bagi perlengkapan usaha para muballighin pada umumnya.
3. Menyebarkan aneka macam penerbitan, antara lain buku-buku, brosur, dan atau siaran lain yang terutama ditujukan untuk memperlengkapi para muballighin dengan ilmu pengetahuan, baik ilmu agama maupun ilmu pengetahuan umum lainnya, guna meningkatkan mutu dan hasil dakwah.

Usaha ini diharapkan dapat mengisi kekosongan-kekosongan di berbagai bidang.

2.9.2. Ikatan Masjid Indonesia (IKMI)

Ikatan Mesjid Indonesia merupakan organisasi masyarakat yang bergerak dalam bidang pelayanan muballigh yang bertugas untuk menyampaikan Khutbah jumat diberbagai mesjid di kota Pekanbaru, pengajian dan sebagai penceramah dalam berbagai acara yang diselenggarakan oleh pengurus. IKMI merupakan cabang dari Dewan dakwah Indonesia kota pekanbaru. Dimana visi dan misi dari DDI di kelola dan dijalankan oleh IKMI. Jadi dewan dakwah menjadi penyusun kerangka kerja dan sebagaii pengawas.

IKMI dan DDI beralamat di Jalan Todak/ jl.Udang Putih No.1 Kelurahan Tanggerang Barat, Kecamatan Marpoyan Damai PEKANBARU.

2.10. *Layering* Pada SOA

Perangkat lunak yang tidak menggunakan SOA secara umum dapat dibagi menjadi dua layer utama, yaitu *Application layer* dimana aplikasi dijalankan dan *Business Process Layer* yang mendeskripsikan bagaimana proses bisnis dalam perusahaan berjalan. Proses bisnis organisasi akan didefinisikan dalam aplikasi bersamaan dengan kode program yang bersifat teknis.

Dalam implementasi SOA , konsep *Service-Oriented* yang dipegangnya diimplementasikan dalam sebuah *layer* diantara *Business Process Layer* dan *Aplication Layer* yang mana keduanya merupakan bagian dari *Enterprise Logic*. *Layer* tersebut dinamakan *Service Interface Layer*.

Fungsi dari *layer* ini adalah untuk mengenkapsulasi lojik yang ada di *Application Logic*, sekaligus *Business Process* yang ada di *Business Process* yang ada di *Business logic*. Ada beberapa jenis layer yaitu (Erl,2005):

1. ***Application Service Layer***

Application Service Layer, menyediakan sekumpulan *service* yang spesifik untuk mengekapsulasi teknologi tertentu yang terdapat di dalam *Application Logic*. *Service* yang disediakan dalam layer ini akan melakukan abstraksi terhadap semua lojik yang tidak terkait dengan proses bisnis, namun dibutuhkan untuuk menjalankan fungsi-fungsi yang ada diproses bisnis tersebut.

2. ***Business Service Layer***

Business Service Layer merepresentasikan *business logic* dari aplikasi. Layer ini bisa diibaratkan sebagai coktroller dari *application service layer*. Pada *Business Service Layer* inilah fungsi-fungsi bisnis, yang berupa aktivitas-aktivitas yang dilakukan untuk menjalankan dua macam *service* yang mungkin disediakan oleh *Business Service Layer* , yaitu :

a. *Task-centric*, yaitu sebuah *service* yang mengenkapsulasi logik tertentu dalam proses bisnis yang melibatkan satu atau dua entitas bisnis. *Service* jenis ini memiliki potensi reuse yang terbatas, namun proses analisis untuk menghasilkannya akan lebih mudah.

b. *Entity-centric*, yaitu *service* yang mengenkapsulasi lojik yang sifatnya spesifik terhadap entitas bisnis tertentu. *Service* jenis ini sangat baik dalam *reusability*, namun proses analisis yang dibutuhkan untuk menghasilkan lebih sulit.

3. ***Orchestration Service Layer***

Orchestration service layer merupakan sebuah layer yang menyediakan abstraksi dengan level tertinggi dari aplikasi. Pada layer ini semua proses bisnis yang ada di dalam sistem yang ada di dalam sistem didefinisikan dan dijalankan dengan menggunakan fungsi-fungsi yang terdapat pada *Business Service Layer* .

2.11. Kuisioner

langkah-langkah pengolahan data penelitian menurut Martadiputra adalah Menyusun instrumen penelitian berdasarkan dimensi dan indikator yang dirujuk, Uji validitas dan reliabilitas teoritis instrumen penelitian, Uji validitas dan reliabilitas empirik instrumen penelitian hasil uji coba:

1. Untuk setiap jawaban responden, ubah data kualitatif hasi langket menjadi data kuantitatif, misalnya :SS=5, S=4, N=3 ,TS=2, dan STS=1 untuk pernyataan positif dan SS=1, S=2, N=3, TS=4, dan STS=5 untuk pernyataan negatif.
2. Hitung skor total untuk setiap variabel penelitian.

3. Ubah skor tota luntuk setiap variabel penelitian kedalam bentuk nilai (persen) dengan rumus:

$$\text{Nilai} = \frac{\text{Skor total}}{\text{Banyak kuisioner X banyak soal}} \times 100$$

2.12. Penelitian Sebelumnya.

Pada penelitian ini diambil beberapa penelitian terkait dengan penelitain yang telah dilakukan sebelumnya dengan metode SOA untuk dijadikan rujukan utama. Diantara penelitian itu adalah:

1. Penelitian yang dilakukan oleh Kapojos,F Dkk(2012) yang mengangkat judul tentang Implementasi *Service Oriented Architecture* Dengan *Web Service* Untuk Aplikasi Informasi Akademik. Dimana tujuan dari penelitian ini adalah membuat suatu *Interface web service* yang menerapkan SOA untuk diIntgrasikan dengan Aplikasi Informasi Akademik Fakultas Teknik UNSIRAT. Dari penelitian ini kesimpulan yang bisa diambil bahwa penggunaan SOA dan *web service* dalam perancangan sistem informasi suatu organisasi sangat tepat untuk menghasilkan suatu aplikasi yang memiliki tinggkat penggunaan kembali yang sangat tinggi, selain itu dapat digunakan antar *flatform* dan sistem operasi.

2. Penelitian yang dilakukan oleh Toninetti(2009) merupakan mahasiswa jurusan *Database*, BINUS University Jakarta, yang mengangkat judul Analisis Dan Perancangan Modul Aplikasi Penerimaan Mahasiswa Baru Berbasiskan *Service Oriented Architecture* (SOA). Dimana pada penelitian ini penulis mencoba sebuah solusi dari kombinasi tahapan baru untuk mengatasi masalah proses pengembangan sistem informasi yang lama dengan aplikasi baru yang sudah dirancang dengan SOA. Selain itu penelitian ini menghasilkan kombinasi kombinasi dengan menghadirkan tiga modul aplikasi penerimaan mahasiswa baru yang lebih mempercepat kegiatan pengaturan penerimaan mahasiswa baru.

3. Kemudian penelitian yang dilakukan oleh Marini(2012), mahasiswa komputerisasi akuntansi STMIK ATMA LUHUR pangkal pinang Bangka Belitung yang meneliti tentang Penerapan Model *Service Oriented*

Architectur(SOA) pengintegrasian Sistem Informasi STMIK ATMA LUHUR. Dimana Penelitian ini akan membuat kajian tentang penerapan model SOA untuk model pengintegrasian sistem informasi perguruan tinggi selanjutnya membuat sebuah prototipe perangkat lunak yang akan diimplementasikan pada perguruan tinggi dengan menggunakan teknologi *web services* untuk pengintegrasian sistem informasi.. tujuan dalam pelaksanaan penelitian ini yaitu mengidentifikasi kebutuhan layanan informasi yang diharapkan oleh *user* pada perguruan tinggi, mengidentifikasi proses bisnis dan pemanfaatan teknologi informasi dan komunikasi kini, membuat sebuah model proses pengintegrasian sistem informasi dengan menggunakan SOA, mengimplementasikan hasil perancangan dengan sebuah modul perangkat lunak untuk mengintegrasikan sistem informasi pada perguruaan tinggi.