BAB II
LANDASAN TEORI

2.1. PENGERTIAN E-GOVERNMENT

Menurut Cahyana Ahmadjayadi dalam keynote speech acara Workshop Standarisasi Menuju Interoperabilitas e-Government tahun 2006, menerangkan bahwa:

> *"e-Government merupakan kegiatan yang terkait dengan upaya seluruh lembaga pemerintah dalam bekerja bersama-sama memanfaatkan teknologi komunikasi dan informasi, sehingga dapat menyediakan jasa layanan elektronik dan informasi yang akurat kepada individu masyarakat dan dunia usaha. Inisiatif e-Government adalah suatu proses yang berlangsung terus menerus untuk memperbaiki kinerja pemerintah dan penyelenggaraan layanan yang efisien bagi publik. Perlu ditekankan bahwa, efisiensi sangat tergantung pada kurun waktu dan teknologi. e-Government yang sangat efisien saat ini belum tentu efisien beberapa tahun ke depan karena perkembangan TIK dan demand dari stakeholdernya."*

Pengembangan *e-Government* merupakan upaya untuk mengembangkan penyelenggaraan kepemerintahan melalui penggunaan media elektronik untuk meningkatkan kualitas layanan publik. Dengan adanya pengembangan *e-Government* maka perlu dilakukan penataan sistem dan proses kerja di lingkungan pemerintahan melalui pemanfaatan teknologi informasi. Pemanfaatan teknologi informasi tersebut mencakup 2 (dua) kegiatan atau aktifitas yang berkaitan langsung, (Inpres No.3, 2003) yaitu:

a. Pengolahan data, pengelolaan informasi, sistem manajemen dan proses kerja secara elektronis;
b. Pemanfaatan kemajuan teknologi informasi agar pelayanan publik dapat diakses secara mudah dan murah oleh masyarakat di seluruh wilayah negara.

Pelaksanaan dalam pengembangan *e-Government* diarahkan untuk mencapai 4 (empat) tujuan utama, (Inpres No.3, 2003) yaitu :

a. Pembentukan jaringan informasi dan transaksi pelayanan publik yang memiliki kualitas dan lingkup yang dapat memuaskan masyarakat luas serta dapat terjangkau di seluruh wilayah Indonesia pada setiap saat tidak dibatasi oleh sekat waktu dan dengan biaya yang terjangkau oleh masyarakat.
b. Pembentukan hubungan interaktif dengan dunia usaha untuk meningkatkan perkembangan perekonomian nasional dan memperkuat kemampuan menghadapi perubahan dan persaingan perdagangan internasional.
c. Pembentukan mekanisme dan saluran komunikasi dengan lembaga-lembaga negara serta penyediaan fasilitas dialog publik bagi masyarakat agar dapat berpartisipasi dalam perumusan kebijakan negara.
d. Pembentukan sistem manajemen dan proses kerja yang transparan dan efisien serta memperlancar transaksi dan layanan antar lembaga pemerintah dan pemerintah daerah otonom.

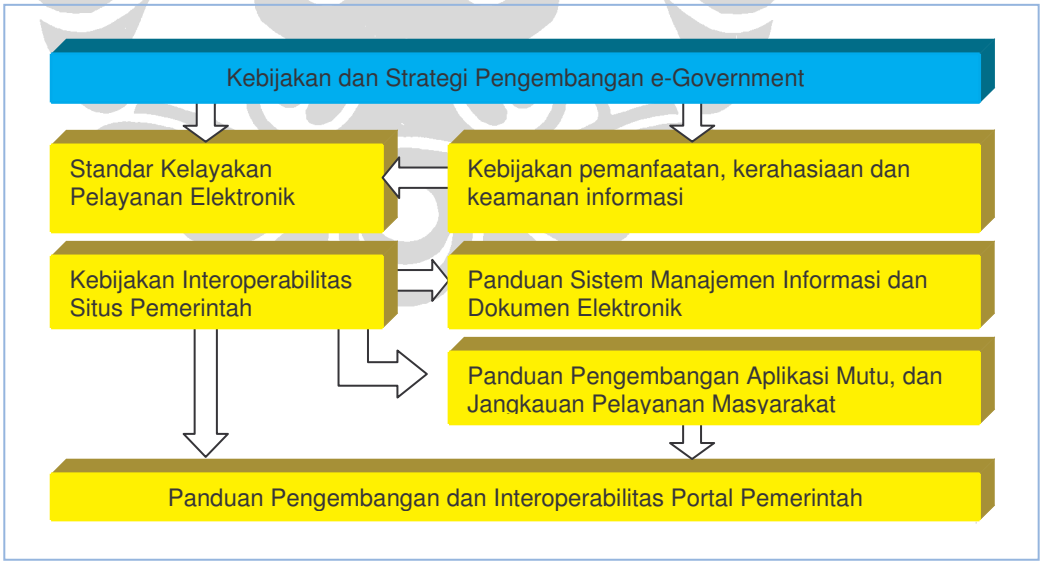
Berdasarkan sifat transaksi informasi dan pelayanan publik yang disediakan oleh pemerintah, pengembangan e-Government dilaksanakan melalui 4 (empat) tingkatan sebagai berikut (Inpres No.3, 2003):

- Tingkat 1 – Persiapan yang meliputi:
  - Pembuatan situs informasi di setiap lembaga;
  - Penyiapan SDM;
  - Penyiapan sarana akses yang mudah misalnya menyediakan sarana Warnet, SME-Center, dll;
  - Sosialisasi situs informasi baik untuk internal maupun untuk publik.
- Tingkat 2 – Pematangan yang meliputi:
  - Pembuatan situs informasi publik interaktif;
  - Pembuatan antar muka keterhubungan dengan lembaga lain;

- Tingkat 3 – Pemantapan yang meliputi:
  - Pembuatan situs transaksi pelayanan publik;
  - **Pembuatan interoperabilitas aplikasi maupun data dengan lembaga lain**.
- Tingkat 4 – Pemanfaatan yang meliputi:
  - Pembuatan aplikasi untuk pelayanan yang bersifat G2G, G2B dan G2C yang terintegrasi.

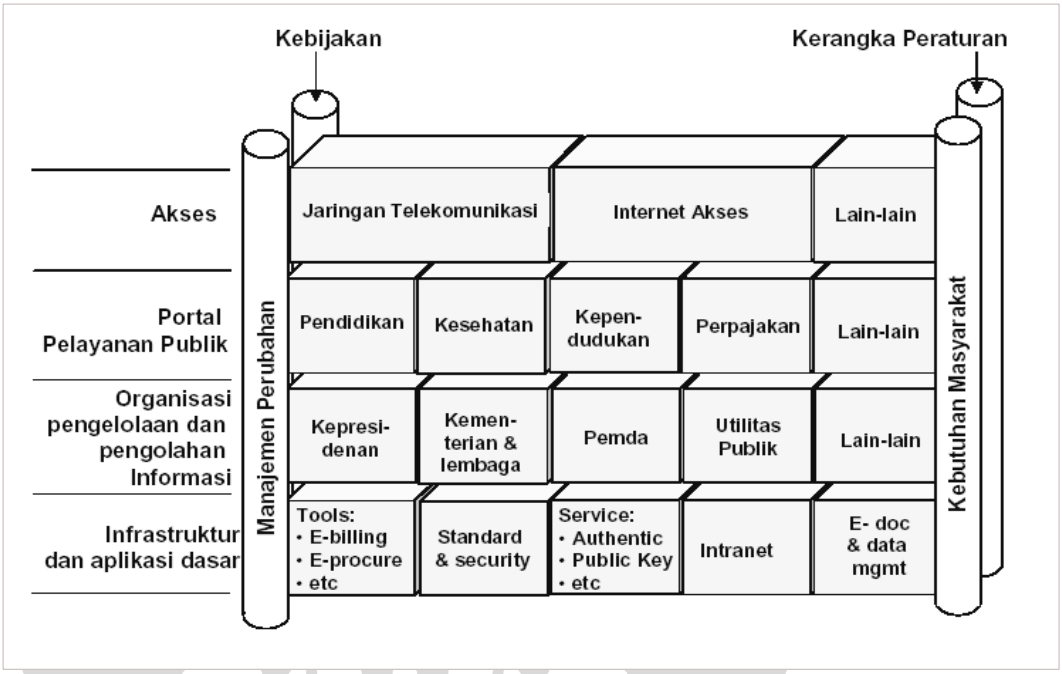
Pada tingkat ke tiga di sisi pemantapan pengembangan e-Government terdapat rencana pembuatan interoperabilitas aplikasi antar lembaga. Dengan demikian untuk menuju ke tingkat pemanfaatan, harus ada kesiapan sistem informasi layanan publik yang dapat saling berinteraksi antar lembaga.

Agar pelaksanaan kebijakan pengembangan e-Government dapat dilaksanakan secara sistematik dan terpadu serta diarahkan untuk memenuhi kebutuhan pembentukan pelayanan publik, maka pada Inpres Nomor 3 Tahun 2003 terdapat perumusan yang mengacu pada kerangka yang utuh. Kerangka tersebut mengkaitkan semua kebijakan, peraturan perundang-undangan, standarisasi, dan panduan, sehingga terbentuk ilustrasi pada gambar berikut ini.



Gambar 2.1. Pengembangan Pelayanan Publik
Melalui Jaringan Komunikasi dan Informasi
(Inpres No.3, 2003. "Telah diolah kembali")

Untuk menjamin keterpaduan serta interoperabilitas antar sistem e-Government, maka perencanaan dan pengembangan e-Government dirumuskan dalam kerangka arsitektur e-Government, seperti diilustrasikan dalam gambar berikut ini:



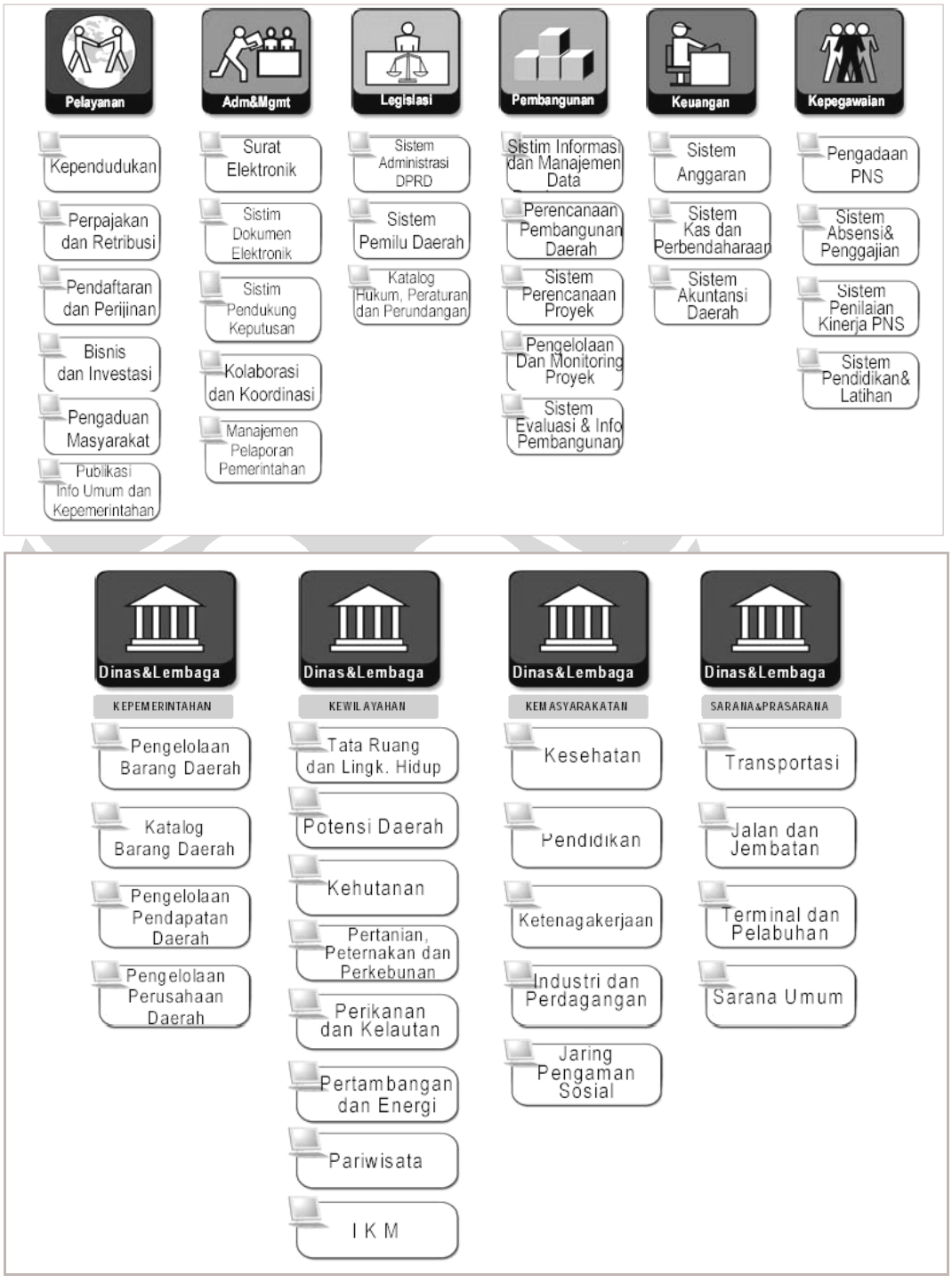
Gambar 2.2. Kerangka Arsitektur e-Government
(Inpres No.3, 2003)

2.2. KEBIJAKAN SISTEM INFORMASI LAYANAN PUBLIK

Menurut pendapat J.L. Whitten, Sistem Informasi adalah pengaturan orang, data, proses dan teknologi informasi yang saling berinteraksi untuk mengumpulkan, memproses, menyimpan dan menyediakan data sebagai sebuah informasi/keluaran yang dibutuhkan untuk mendukung kegiatan sebuah organisasi.

Demikian pula dengan sistem informasi yang dikembangkan pemerintah untuk layanan publik tidak terlepas dari pengaturan orang, data, proses dan teknologi informasi. Hal ini dibuktikan melalui adanya kebijakan pemerintah tentang Blueprint Sistem Aplikasi e-Government, dengan tujuan agar sistem informasi yang dikembangkan pemerintah dapat dipertanggung jawabkan kepada masyarakat dan sesuai dengan kebutuhan fungsional layanan publik. Fungsi-fungsi pelayanan, administrasi dan kelembagaan dikelompokan dalam blok

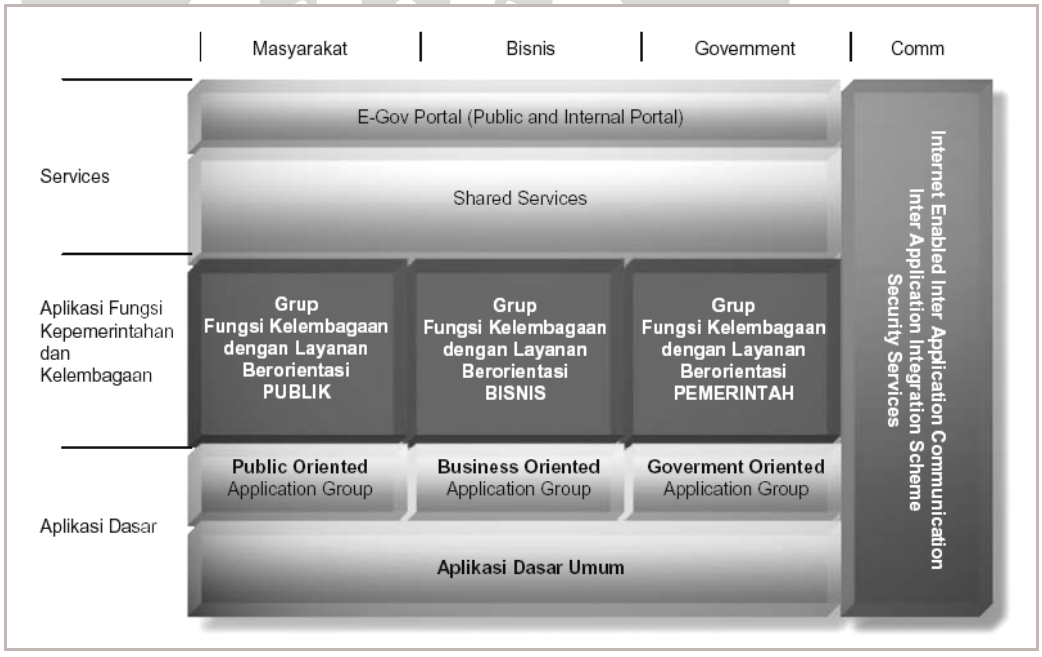
fungsi. Kelompok blok fungsi disusun dalam sebuah bagan fungsi yang selanjutnya dalam dokumen blueprint disebut sebagai Kerangka Fungsional Sistem Kepemerintahan (Blueprint e-Gov, 2004).



Gambar 2.3. Kerangka Fungsi Sistem Kepemerintahan.
(Blueprint e-Gov, 2004)

Melalui pertimbangan fungsi sistem aplikasi dan layanannya, sistem aplikasi-sistem aplikasi e-Government disusun dan dikelompokan dalam sebuah sistem kerangka arsitektur, yang selanjutnya disebut sebagai Peta Solusi Aplikasi e-Government. Dalam peta solusi e-Government, sistem aplikasi dikelompokan melalui pendekatan matrik antara orientasi fungsi layanan dan sifat fungsi sistem aplikasi pada 3 (tiga) bagian, (Blueprint e-Gov, 2004) yaitu:

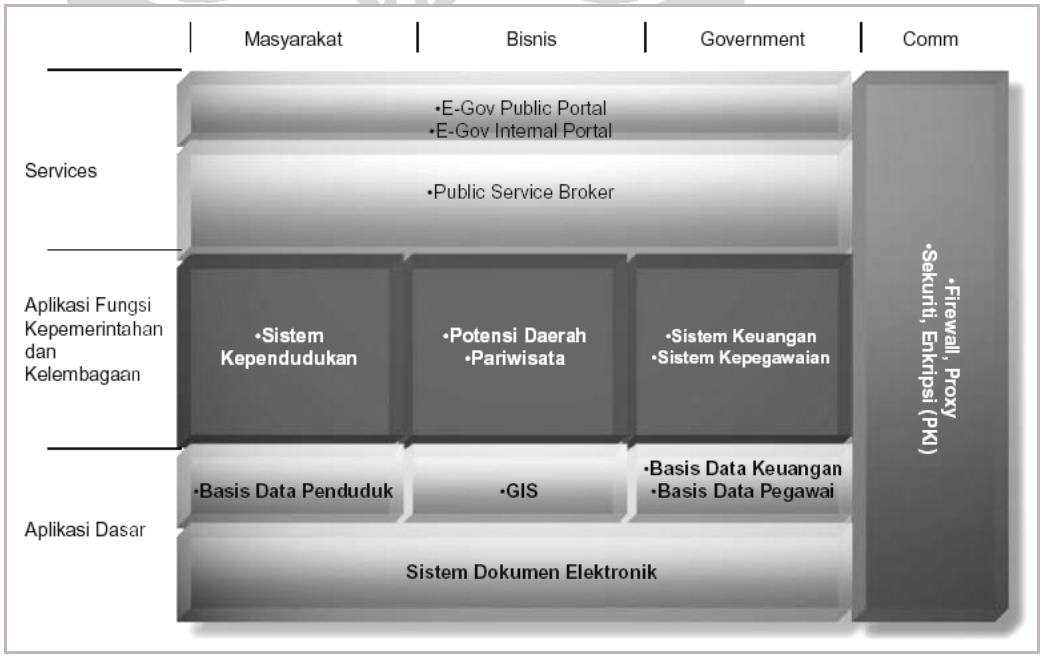
1. Kelompok sistem aplikasi yang orientasi fungsinya langsung memberikan pelayanan kepada penggunanya (aplikasi front office).
2. Kelompok sistem aplikasi yang orientasi fungsinya lebih banyak ditujukan untuk memberikan bantuan pekerjaan yang bersifat administrasi kepemerintahan, serta fungsi-fungsi kedinasan dan kelembagaan (aplikasi back office).
3. Kelompok sistem aplikasi yang fungsi layanannya bersifat mendasar dan umum, diperlukan oleh setiap pengguna, atau setiap sistem aplikasi lain yang lebih spesifik. Sifat layanan aplikasi dasar biasanya back office.



Gambar 2.4. Peta Solusi Aplikasi e-Government.
(Blueprint e-Gov, 2004)

Setiap kelompok sistem tersebut, masing-masing dibagi lagi ke dalam tiga sub kelompok berdasarkan orientasi pengguna yang dilayaninya, (Blueprint e-Gov, 2004) sebagai berikut:

1. Kelompok sistem aplikasi e-Government yang orientasi fungsinya melayani kebutuhan dan kepentingan masyarakat (G2C: Government To Citizen).
2. Kelompok sistem aplikasi e-Government yang orientasi fungsinya melayani kebutuhan dan kepentingan kalangan bisnis (G2B: Government To Business).
3. Kelompok sistem aplikasi e-Government yang orientasi fungsinya melayani kebutuhan internal lembaga kepemerintahan, atau kebutuhan dari pemerintah daerah lainnya (G2G: Government To Government).



Gambar 2.5. Contoh Peta Solusi Aplikasi e-Government.
(Blueprint e-Gov, 2004)

Gambar 2.5. pada kolom orientasi masyarakat terdapat blok fungsi sistem kependudukan yang dapat dikembangkan menjadi bagian dari layanan publik melalui Public Service Broker dan Public Portal. Dengan mengacu pada keterangan tersebut maka sistem kependudukan dapat dijadikan studi kasus pada penelitian ini.

Untuk mengembangkan sebuah sistem aplikasi e-Government, terdapat standar kebutuhan sistem aplikasi yang harus dipenuhi, yaitu (Blueprint e-Gov, 2004):

- Reliable, menjamin bahwa sistem aplikasi akan dapat berjalan dengan handal (robust) terhadap kesalahan pemasukan data, perubahan sistem operasi dan bug free.
- Interoperable, menjamin bahwa sistem aplikasi akan dapat saling berkomunikasi serta bertukar data dan informasi dengan sistem aplikasi lain untuk membentuk sinergi sistem.
- Scalable, menjamin bahwa sistem aplikasi akan dapat dengan mudah ditingkatkan kemampuannya, terutama penambahan fitur baru, penambahan user dan kemampuan pengelolaan data yang lebih besar.
- User Friendly, menjamin bahwa sistem aplikasi akan mudah dioperasikan dengan user interface (antar muka pengguna) yang lazim berlaku di pemerintahan dan sesuai dengan kebiasaan bahasa dan budaya penggunanya.
- Integrateable, menjamin bahwa sistem aplikasi mempunyai fitur untuk kemudahan integrasi dengan sistem aplikasi lain, terutama untuk melakukan transaksi pertukaran data dan informasi antar sistem aplikasi e-Government, baik dalam lingkup satu pemerintah daerah dengan pemerintah daerah lain.

2.3. PRINSIP INTEROPERABILITAS

Pada jaringan internet, pengolahan informasi dari suatu sistem dapat dilakukan dari berbagai sumber data melalui kolaborasi informasi. Meningkatnya kebutuhan akan kolaborasi informasi menyebabkan sistem informasi tersebut harus dapat berkomunikasi dan berintegrasi dengan sistem informasi lainnya. Setiap sistem informasi pada suatu organisasi akan memiliki perbedaan platform database, aplikasi, dan bahasa pemrograman. Dengan demikian proses transaksi informasi antar sistem informasi akan menjadi terhambat karena masalah ketidak sesuaian platform sistem, struktur data, sintak perintah dan konsep informasi. Hambatan tersebut dapat diatasi melalui pengembangan interoperabilitas antar

sistem informasi, karena dengan menggunakan metode interoperabilitas, pertukaran informasi secara lebih luas dapat dilakukan. Selain itu metode interoperabilitas menerapkan teknologi standar terbuka yang dapat diakses oleh beragam platform teknologi informasi, sehingga tidak memiliki ketergantungan terhadap suatu vendor.

Definisi interoperabilitas berdasarkan informasi dari Website Dublin Core Metadata Glossary adalah:

> *"The ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful manner. There are three aspects of interoperability: semantic, structural and syntactical."*

Secara teknis interoperabilitas menggambarkan kemampuan dua atau lebih sistem untuk saling tukar menukar data atau informasi dan saling dapat mempergunakan data atau informasi yang dipertukarkan tersebut (Pedoman Interoperabilitas, 2008). Dengan kata lain bahwa interoperabilitas sistem informasi merupakan kemampuan dua atau lebih suatu sistem informasi yang memiliki perbedaan jenis perangkat lunak, database, data model, maupun teknik pengontrolan untuk melakukan transaksi data atau informasi dalam rangka memaksimalkan ketersediaan akses data atau informasi.

Kehadiran konsep interoperabilitas umumnya disebabkan karena berkembangnya teknologi infomasi dengan jenis dan kemampuan yang beraneka ragam untuk memenuhi berbagai macam kebutuhan informasi. Melalui keragaman teknologi informasi ini A.P. Sheth membagi jenis keragaman teknologi informasi menjadi beberapa kategori yang berkorespondensi pada level interoperabilitas, yaitu *system, syntax, structure, dan semantic*. Bentuk dari kategorisasi keragaman teknologi informasi yang terkait dengan tingkat interoperabilitas digambarkan dalam tabel 2.1.

Tabel 2.1. Pemetaan Level Interoperabilitas Informasi

Jenis Keragaman	Level Interoperabilitas
Keragaman Informasi:	
Keragaman arti konsep	Semantic Interoperabiliy
Keragaman struktur, skema informasi	Structural Interoperability
Keragaman atribut, format	Syntactic Interoperability
Keragaman Sistem:	System Interoperability
Keragaman Sistem Informasi: - Digital Media Repository Management System - Database Management System - Recovery System	Operation Level
Keragaman Plaform: - Operating System - Hardware System	Low Level

(A.P.Sheth, 1998. "Telah diolah kembali")

Berdasarkan hasil pemetaan pada tabel 2.1. tersebut maka diperoleh dua jenis keragaman teknologi informasi yaitu (Pedoman Interoperabilitas, 2008):

1. Keragaman Informasi, terdiri dari tiga jenis yang meliputi:
    a. Keragaman Syntactic, keragaman jenis ini sudah dimulai dari model database tradisional. Adapun beberapa contoh dari keragaman syntactic adalah:
        - naming conflict, misalkan perbedaan pemberian nama akan sesuatu hal, seperti alamat dengan lokasi.
        - data representation conflict, misalkan informasi tentang tanggal dapat direpresentasikan dalam format date, numeric atau text.
        - data scalling conflict, misalkan pendefinisian penghasilan kelas bawah, menengah dan atas.

b. Keragaman Struktural atau Skema, adalah keragaman dalam katalog atau taksonomi informasi. Adapun beberapa contoh kasus pada keragaman struktural adalah:
   - superclass, sebuah konsep atau atribut seperti nama dapat memiliki arti yang berbeda karena diletakan pada struktur yang berbeda, karena 1 (satu) adalah struktur yang menunjukan nama produk (superclass adalah produk), sedangkan yang lain adalah struktur yang menunjukan nama orang (superclass adalah individu).
   - subclass, sebuah konsep yang memiliki label sama belum tentu berarti sama karena subclass yang berbeda.

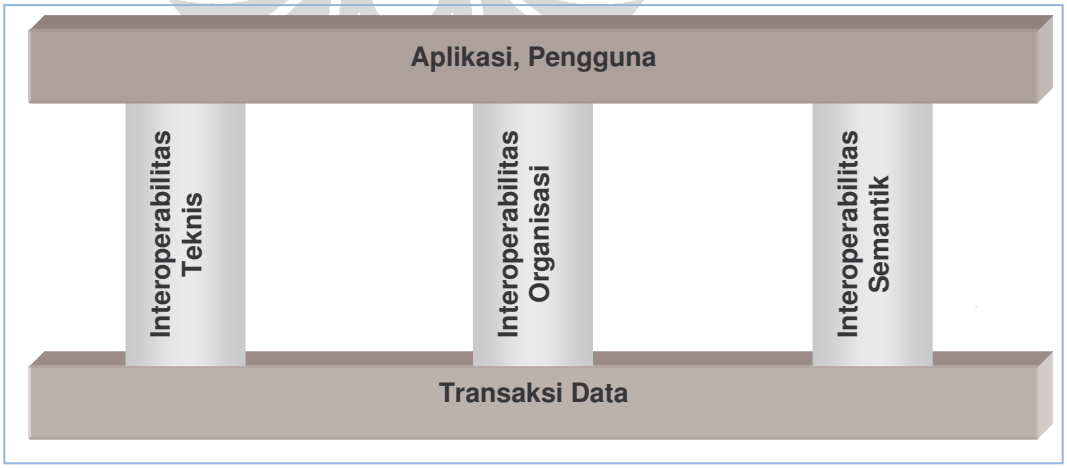
c. Keragaman Semantik, semantik adalah ilmu yang mempelajari arti, maka keragaman semantik adalah keragaman akan perbedaan arti, ini bisa bermakna:
   - sinonim, antonim, adalah persamaan dan lawan kata.
   - bagian dari, adalah menjelaskan suatu relasi.
   - menghitung tingkat kesamaan (similarity), adalah kasus untuk menghitung konsep mana yang lebih besar similar dan juga memungkinkan untuk menghitung nilai similar dalam kuantitas, sebagai contoh kalau dicari tingkat kesamaan antara pohon-kucing dengan pohon-rumput maka dengan mudah pohon-rumput lebih memiliki kesamaan dibandingkan pohon-kucing.

2. Keragaman Sistem, terdiri dari dua jenis yang meliputi:
   a. Keragaman Sistem Informasi meliputi perbedaan jenis perangkat lunak, database, data model, dan teknik pengolahan data.
   b. Keragaman Platform umumnya berupa:
      - perbedaan sistem operasi yang meliputi tipe file, model transaksi, model keamanan.
      - perbedaan perangkat keras yang meliputi keragaman processor, keragaman design, dan keragaman set instruksi.

Menurut Kerangka Acuan dan Pedoman Interoperabilitas Sistem Informasi Instansi Pemerintahan Departemen Komunikasi dan Informatika, bahwa untuk melakukan interaksi dalam interoperabilitas diperlukan 3 (tiga) pilar pendekatan secara umum, yaitu:

1. Interoperabilitas semantik, lebih memperhatikan pemahaman arti sehingga pertukaran data atau informasi dapat dipahami baik oleh manusia ataupun mesin. Sehingga dengan interoperabilitas semantik memungkinkan sistem untuk mengkombinasikan informasi yang diterima dengan berbagai sumber yang lain dan diproses untuk memberikan hasil yang memiliki arti.
2. Interoperabilitas organisasi lebih menekankan pada pendefinisian tujuan kegiatan, permodelan kegiatan dan membawa ke level administrasi untuk pertukaran informasi. Lebih lanjut interoperabilitas di tingkat organisasi memiliki tujuan untuk melakukan pendefinisian kebutuhan pelayanan masyarakat.
3. Interoperabilitas teknis adalah meliputi isu teknis pada keterhubungan sistem dan services (layanan). Ini dapat meliputi komponen seperti open interface, interconnection services, data integration and middleware, data presentation and exchange, accessibility and security services.

Tiga pilar dimensi interoperabilitas menurut kerangka acuan dan pedoman interoperabilitas tersebut diilustrasikan pada gambar berikut ini.



Gambar 2.6. Pilar Dimensi Interoperabilitas

(Kerangka Acuan dan Pedoman Interoperabilitas, 2008. "Telah diolah kembali")

Apabila jenis keragaman menurut pendapat A.P. Sheth dipetakan ke dalam Pilar Dimensi Interoperabilitas, maka akan diperoleh signifikansi antara pendapat A.P. Sheth dengan Kerangka Acuan dan Pedoman Interoperobilitas milik Depkominfo melalui ilustrasi pada tabel berikut ini.

Tabel 2.2. Pemetaan Pilar Dimensi Interoperabilitas

Jenis Keragaman	**Dimensi Interoperabilitas**
Keragaman Organisasi	Interoperabilitas Organisasi
Keragaman Informasi	Interoperabilitas Semantik
Keragaman Sistem	Interoperabilitas Teknis

Atas dasar hasil pemetaan tersebut maka penelitian ini akan membahas interoperabilitas dari sisi teknis, karena permasalahannya adalah penanganan interoperabilitas terhadap keragaman sistem informasi yang tersedia. Pada ruang lingkup interoperabilitas teknis, pelayanan berbasis internet termasuk e-Services pemerintahan memiliki berbagai bentuk interaksi. Masing-masing interaksi dalam sistem pelayanan e-Government dibedakan berdasarkan tingkat kemampuan berikut ini (Pedoman Interoperabilitas, 2008):

- **Level 1**, pelayanan online hanya memberikan informasi, masyarakat cukup membaca informasi secara online atau mengunduh.
- **Level 2**, tersedianya formulir secara online yang dapat diunduh, kemudian dikembalikan melalui pos, fax, atau email.
- **Level 3**, adanya transaksi individu antara pemakai dan pemberi layanan, memungkinkan formulir dapat diisi secara online.
- **Level 4**, multiple transaksi yang mungkin dilakukan karena pelayanan sudah terintegrasi antar berbagai lembaga pemerintah dan menunjang pengolahan data otomatis.

Secara garis besar dapat diumpamakan, bahwa level 1 dan 2 menitik beratkan pada pelayanan loket yang belum memiliki proses elektronik, sedangkan pada level 3 dan 4 sudah didukung oleh proses elektronik.

2.4. TEKNOLOGI PENDUKUNG INTEROPERABILITAS

Terkait dengan pendekatan tiga pilar dimensi interoperabilitas, maka pengembangan sistem informasi layanan publik di pemerintahan yang sudah diatur dalam kebijakan e-Government dapat diawali dengan melakukan peningkatan kemampuan sistem informasi atau sistem aplikasi untuk dapat saling berbagi akses informasi dengan sistem informasi atau sistem aplikasi lainnya. Pendekatan interoperabilitas teknis merupakan tahap yang paling mendasar untuk dilakukan, agar sistem informasi yang sudah dibuat maupun yang akan dikembangkan dapat bersinergi satu sama lain dalam memperkaya ketersediaan akses informasi sesuai kebutuhan internal maupun eksternal.

Mekanisme interoperabilitas antar sistem informasi atau sistem aplikasi merupakan hal yang menjadi perhatian utama untuk mewujudkan terlaksananya interoperabilitas sistem informasi di lingkungan pemerintah. Atas dasar kebutuhan tersebut maka diperlukan sebuah infrastruktur teknologi yang dirancang untuk dijadikan dasar proses interoperabilitas sistem informasi.

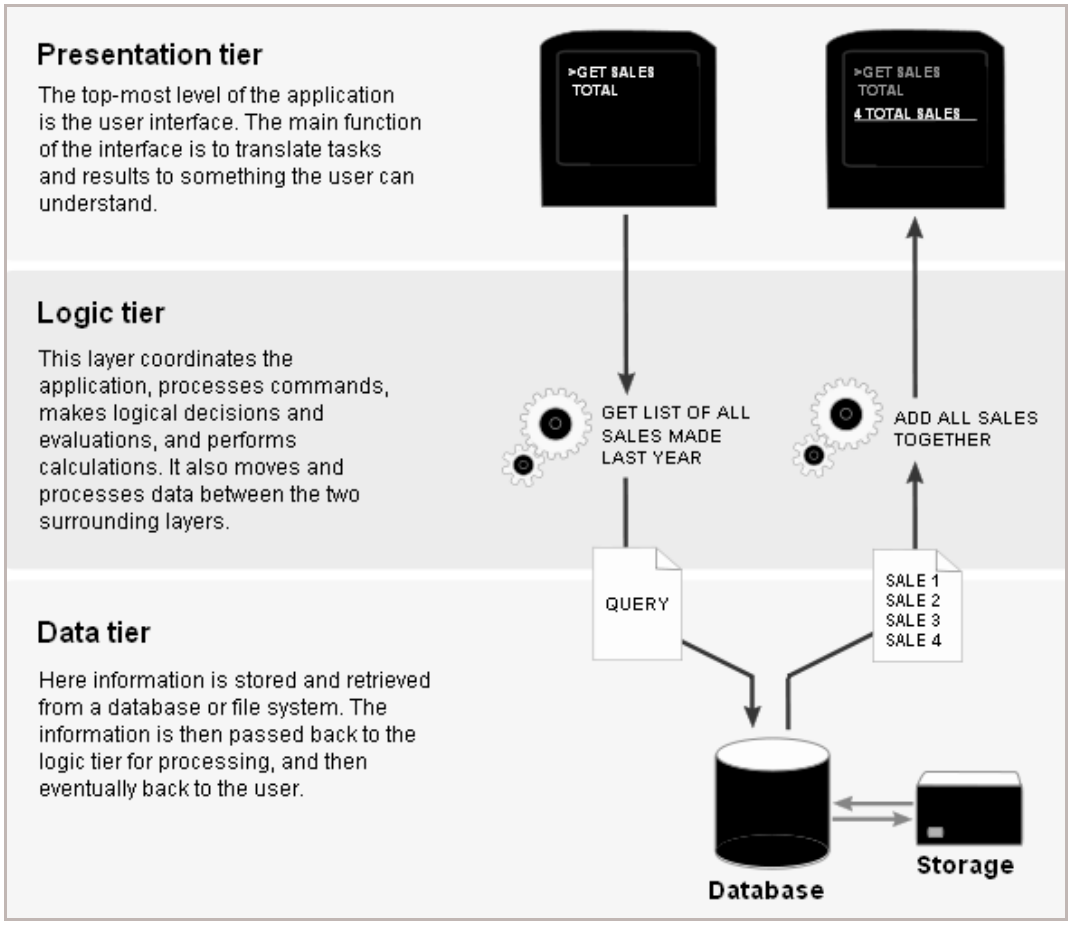
Melalui perkembangan infrastruktur teknologi informasi, pengembangan sistem informasi ke depan mengarah kepada konsep Distributed Computing. Berdasarkan definisi Wikipedia (25 Juni 2009), Distributed Computing adalah:

> *"deals with hardware and software systems containing more than one processing element or storage element, concurrent processes, or multiple programs, running under a loosely or tightly controlled regime.*"

Dalam Distributed Computing, sebuah program dipisah menjadi beberapa bagian yang berjalan secara simultan pada komunikasi jaringan di berbagai komputer. Secara teknis Distributed Computing dapat dikembangkan melalui beberapa alternatif arsitektur dasar, salah satunya adalah arsitektur 3 (Three)-tier.

> Definisi 3 (Three)-tier menurut Wikipedia (25 Juni 2009) adalah:
> *"a client-server architecture in which the user interface, functional process logic ("business rules"), computer data storage and data access are developed and maintained as independent modules, most often on separate platforms."*

Pada model Three-tier, antar muka aplikasi pengguna yang umumnya berbasis grafik berada di lokasi workstation, sedangkan proses logika fungsional berada terpisah pada workstation lain atau server aplikasi. Sementara itu database bisa saja berada pada lokasi yang sama atau berbeda dengan server aplikasi. Konsep infrastruktur Three-tier terdiri dari presentation tier, application tier (business logic), dan data tier. Masing-masing bagian dijelaskan dalam ilustrasi gambar berikut ini.



Gambar 2.7. Model Infrastruktur Three-tier.

(Wikipedia, 2009)

Keberadaan application tier sebagai logic tier atau logika fungsional pada posisi middleware bisa saja dibuat menjadi lebih dari satu, sehingga membentuk proses logika fungsional Multi-tier yang menyebabkan hadirnya konsep arsitektur baru dengan nama N-tier. Aplikasi yang berada pada posisi middleware sendiri

sering digunakan untuk koneksi dengan tier lain secara terpisah atau jarak jauh, tier tersebut bisa database atau aplikasi logika fungsional lainnya.

Beberapa teknologi Distributed Computing yang dapat digunakan untuk perngembangan N-tier diantaranya adalah CORBA (Common Object Request Broker Architecture), DCOM (Distributed Component Object Model), Java-RMI (Remote Method Invocation), dan dotNET Remoting. Teknologi tersebut menggunakan konsep Distributed Object, sehingga fungsi-fungsi logika diletakan pada interface komunikasi data agar dapat digunakan oleh sistem aplikasi. Namun pada implementasinya teknologi tersebut memiliki ketergantungan penuh terhadap platform dari vendor tertentu, maka sulit untuk diterapkan sebagai pendukung terlaksananya interoperabilitas antar sistem informasi yang memiliki keragaman sistem secara terbuka.

Dalam pengembangan aplikasi berbasis web, infrastruktur Three-tier digunakan untuk mengakses website yang berada pada server melalui internet. Bagian-bagian dari konsep Three-tier pada pengembangan website adalah:

1. Front-end Web Server yang menyediakan konten statis.
2. Aplikasi Logika Fungsional berbasis Web sebagai proses perantara (middleware) yang menyediakan konten dinamis.
3. Back-end Database Server sebagai aplikasi manajemen data.

Dengan adanya Aplikasi Logika Fungsional berbasis Web menyebabkan munculnya gagasan untuk membuat Web Services sebagai interface komunikasi antara fungsi-fungsi logika dengan sistem aplikasi. Karena Web Services dapat melakukan komunikasi secara universal pada suatu sistem informasi maka konsep distribusinya dinamakan Distributed System. Disamping itu Web Services dapat dianggap sebagai Remote Procedure Call (RPC) dalam proses interaksi antar aplikasi web melalui sebuah protokol internet yaitu Hyper Text Transport Protocol (HTTP). HTTP merupakan Transport Protocol dengan teknologi standar terbuka yang lebih populer, dan paling banyak digunakan dalam komunikasi data melalui internet. Sehingga dalam implementasnya teknologi Web Services tidak memiliki ketergantungan penuh terhadap platform dari vendor tertentu, dan memudahkan proses interoperabilitas antar sistem informasi. Untuk mendukung

implementasinya Web Services membutuhkan media yang umum digunakan pada infrastruktur Web Server berupa aplikasi preprosesor bahasa pemrograman web seperti PHP Hypertext Preprocessor (PHP), Active Server Page (ASP), atau Java Server Page (JSP) (Werner Vogels, 2003).

2.5. MODEL WEB SERVICES

Secara teknis Web Services memiliki mekanisme penunjang interoperabilitas antar sistem informasi yang berfungsi untuk melakukan interaksi antar sistem informasi baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). Bahkan Web Services memiliki layanan terbuka yang bisa diakses melalui internet oleh pihak manapun melalui teknologi yang dimiliki masing-masing pengguna untuk kepentingan integrasi data dan kolaborasi informasi.

Definisi Web Services menurut WWW Consorsium (2004) adalah:

"a software system designed to support interoperable machine-to-machine interaction over a network."

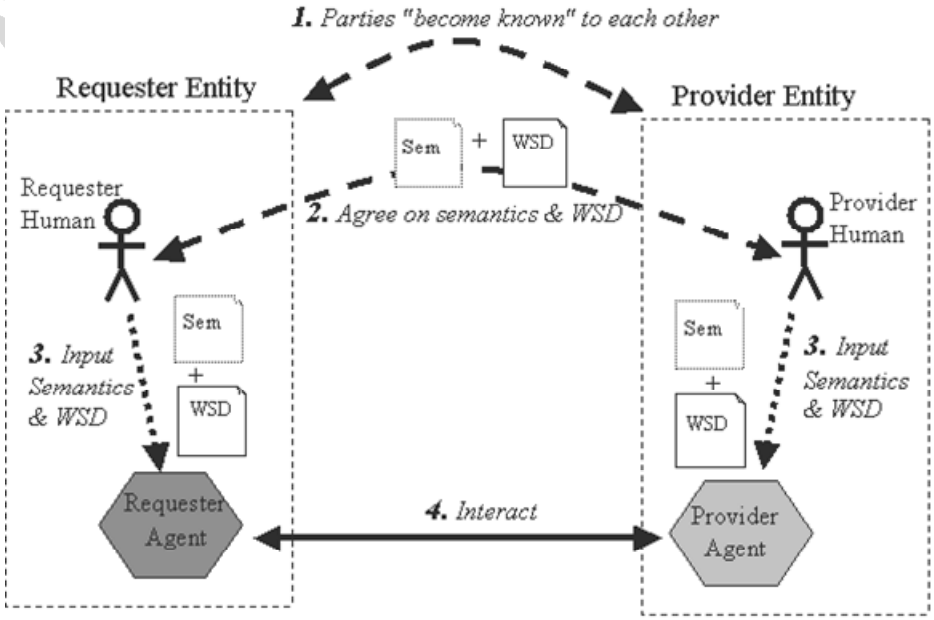
Web Services terkadang bisa juga disebut sebagai Application Programming Interface (API) berbasis web. Namun demikian, Web Services memiliki keunggulan dibandingkan API yang ada pada sistem operasi biasa, karena Web Services dapat dipanggil dari jarak jauh melalui internet. Selain itu untuk memanggil Web Services bisa menggunakan bahasa pemrograman apa saja melalui platform apa saja, tidak seperti API yang hanya bisa digunakan untuk platform tertentu saja (Lucky, 2008). Lebih tepatnya Web Services dapat dianggap sebagai Remote Procedure Call (RPC) yang mampu memproses fungsi-fungsi program yang didefinisikan pada sebuah aplikasi web dan bahkan mengekspos sebuah API atau User Interface (UI) melalui web.

Penggunaan Web Services juga menawarkan banyak kelebihan dan fleksibilitas. Beberapa diantaranya adalah sebagai berikut (Lucky, 2008):

1. Lintas Platform, memungkinkan komputer-komputer yang berbeda sistem operasi dapat saling bertukar data.
2. Language Independent, dapat diakses menggunakan bahasa pemrograman apa saja.

3. Jembatan Penghubung Dengan Database, pada umumnya sebuah aplikasi memerlukan driver database agar bisa melakukan koneksi ke sebuah database. Web services dapat dijadikan sebagai jembatan penghubung antara aplikasi dengan database tanpa memerlukan driver database dan tidak perlu tahu apa jenis DBMS yang digunakan pihak sistem informasi lain. Aplikasi tersebut cukup mengetahui fungsi apa saja yang disediakan Web Services untuk memanfaatkan fasilitasnya.
4. Mempermudah Proses Pertukaran Data, penggunaan Web Services dapat mempermudah dan mempercepat pertukaran data daripada harus menyesuaikan aplikasi, database atau platform yang digunakan.
5. Penggunaan Kembali Komponen Aplikasi, beberapa aplikasi yang berbeda bisa saja memerlukan sebuah fungsi yang sama.

Konsep layanan fungsional yang ditawarkan melalui teknologi Web Services secara tidak langsung telah memberikan gagasan untuk membentuk metode baru. Berikut ini adalah ilustrasi Arsitektur Dasar Web Services menurut W3C Working Group.



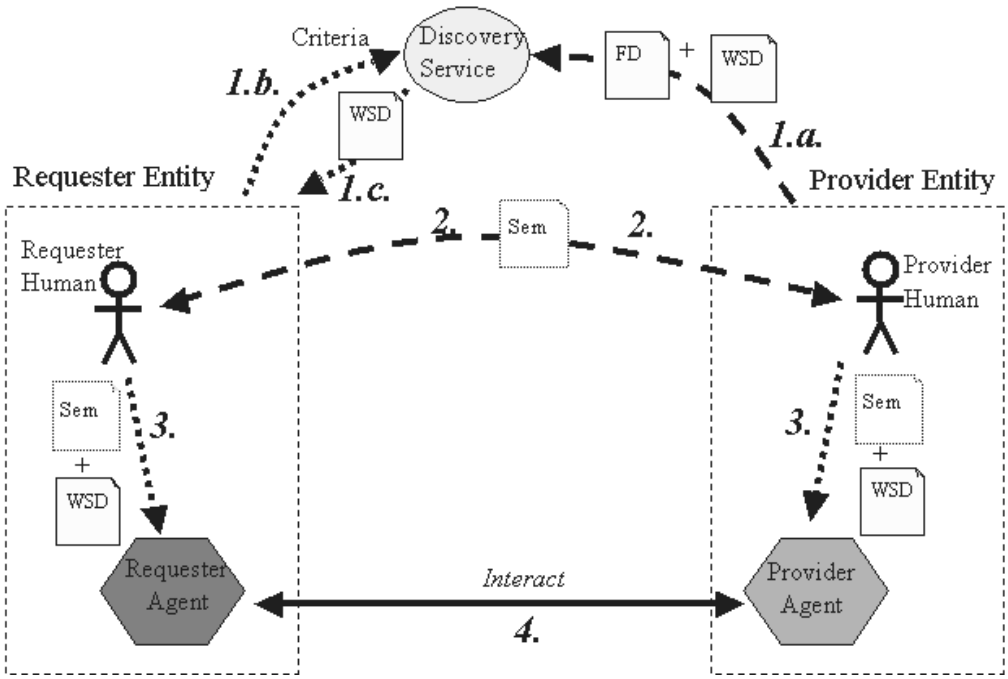
Gambar 2.8. Arsitektur Dasar Web Services

(W3C Working Group, 2004)

Pada Gambar 2.8. dijelaskan bahwa terdapat 4 (empat) langkah kegiatan dari 3 (tiga) entitas yang dapat dilakukan untuk memanfaatkan layanan fungsional yang terdapat dalam Web Services. Tiga entitas tersebut adalah pihak yang saling berhubungan dalam proses inisiasi dan eksekusi layanan fungsional, diantaranya adalah:

1. Entitas Pengguna (Requester Entity), merupakan entitas yang membutuhkan layanan fungsional terhadap penyedia layanan fungsional.
2. Entitas Penyedia (Provider Entity), adalah pihak penyedia yang menyediakan layanan fungsional untuk publik.
3. Entitas Perantara (Discovery Entity), yaitu pihak mediator yang mempertemukan kebutuhan entitas Pengguna dengan publikasi layanan yang dilakukan oleh pihak Penyedia layanan fungsional. Dalam ilustrasi tersebut belum disebutkan secara eksplisit siapa pihak mediator ini.

Untuk memperjelas keberadaan Entitas Perantara dalam metode Web Services, maka ilustrasi prosesnya dapat diperjelas melalui gambar berikut.



Gambar 2.9. Proses Discovery dalam Arsitektur Web Services

(W3C Working Group, 2004)

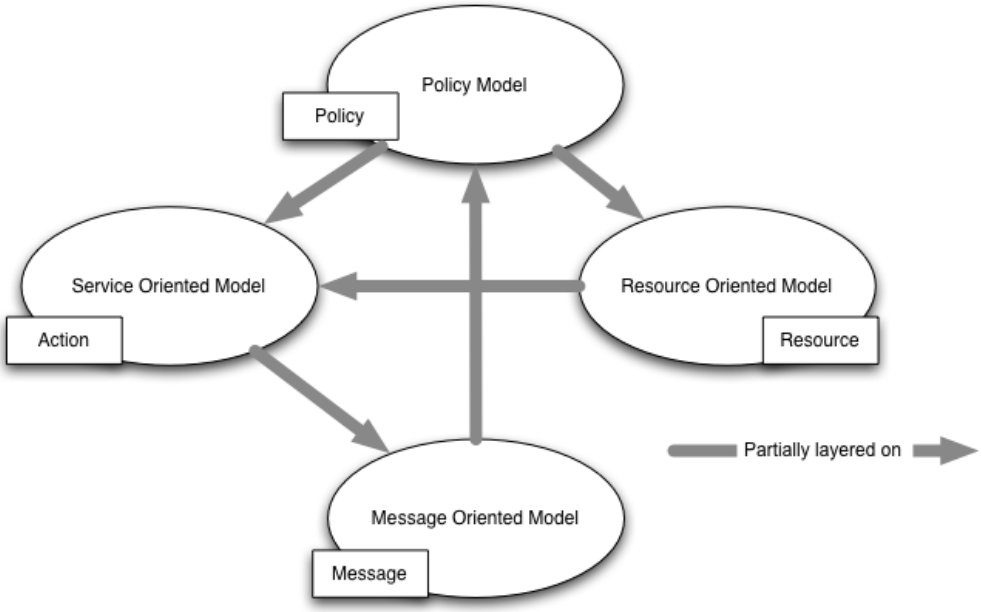
Proses interaksi antar entitas yang dijelaskan pada Gambar 2.9 tersebut adalah meliputi:

1. Provider sebagai penyedia layanan dapat mempublikasikan informasi tentang layanannya melalui Discovery Services, dengan cara mendaftarkan konten layanan yang dimiliki provider dalam format deskripsi yang disepakati bersama (1.a.).
2. Requester dapat mencari informasi layanan yang disediakan para Provider melalui publikasi informasi yang ada pada Discovery Services. Dalam daftar layanan yang disediakan Discovery Services, informasi layanan dapat dicari dan dilihat melalui deskripsi layanan dari suatu Provider (1.b.). Apabila diinginkan, maka pihak Requester dapat mengambil informasi deskripsi layanan tersebut untuk digunakan dalam mengakses layanan yang disediakan pihak Provider (1.c.).
3. Pihak Requester sebelum dapat berinteraksi dengan pihak Provider perlu melakukan validasi kesamaan semantik deskripsi layanan yang disediakan pihak Provider (2). Jika validasi kesamaan semantik berhasil dilakukan, maka semantik deskripsi layanan tersebut dapat disusun sebagai suatu format permintaan proses layanan fungsional (3). Selanjutnya interaksi dapat dilakukan melalui komunikasi data dengan cara mengirimkan permintaan Requester kepada pihak Provider, dan jika layanan fungsional berhasil dilaksanakan maka hasilnya akan dikembalikan kepada pihak Requester (4).

Terlihat jelas bahwa metode Web Services memiliki kemampuan dalam mengatur strategi pertukaran informasi yang dilakukan oleh masing-masing entitas melalui peran dan tugasnya. Inilah yang membedakan teknologi Web Services dengan teknologi lainnya dalam konsep Distributed Computing, apa yang dilakukan dalam metode Web Services lebih dekat pada kondisi kehidupan sosial dengan menganut prinsip keterbukaan dalam melakukan komunikasi dan pertukaran informasi.

Menurut W3C Working Group, berdasarkan konsep hubungan dan penyampaian informasi, Web Services dapat dikembangkan melalui 4 (empat) model arsitektur yang masing-masing berorientasi pada message, action,

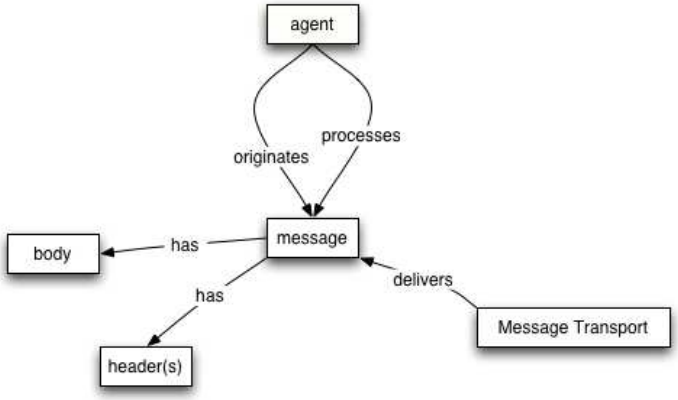
resource, dan policy. Hubungan keempat model arsitektur tersebut di gambarkan dalam sebuah Arsitektur Meta Model beriku ini.



Gambar 2.10. Arsitektur Meta Model

(W3C Working Group, 2004)

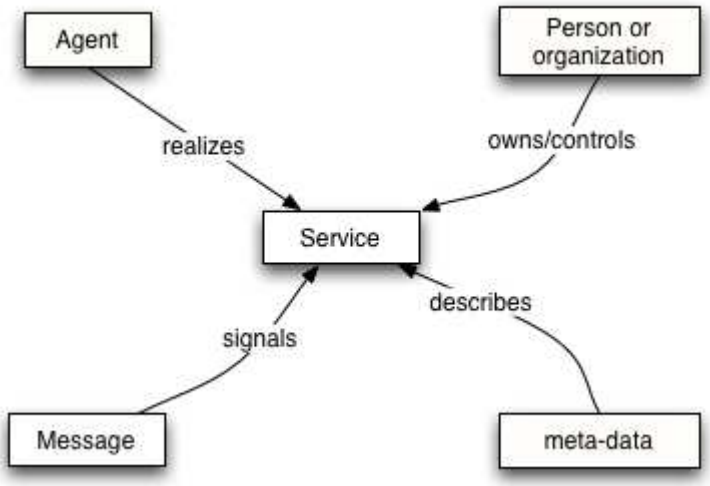
Arsitektur meta model yang berorientasi pada messages secara sederhana menekankan perhatian pada pesan, struktur pesan, transport pesan, dan sebagainya yang terkait dengan pesan. Ilustrasinya adalah seperti gambar berikut ini.



Gambar 2.11. Simplified Message Oriented Model

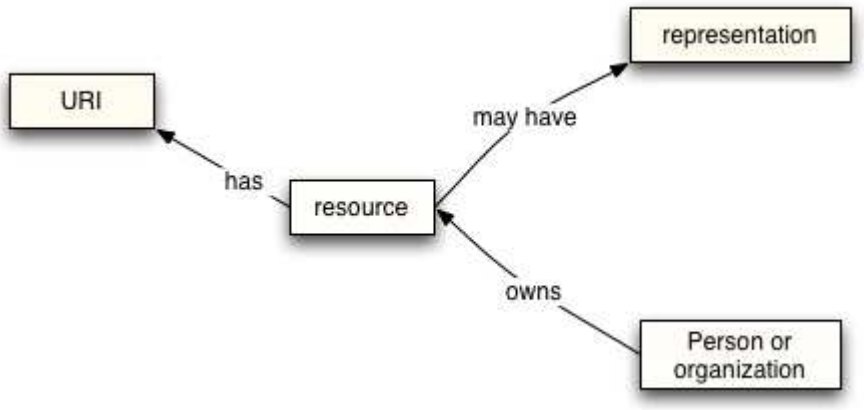
(W3C Working Group, 2004)

Arsitektur meta model yang berorientasi pada aksi layanan secara sederhana menekankan perhatian pada proses layanan, aksi layanan, dan seterusnya diilustrasikan pada gambar berikut ini.



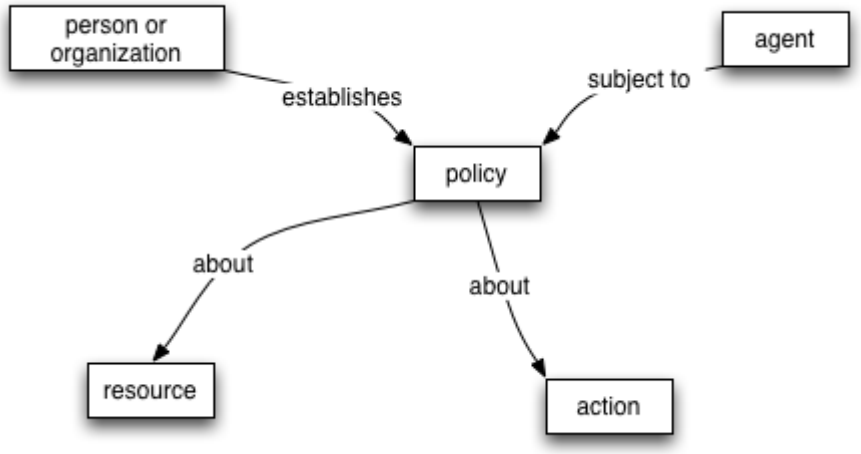
Gambar 2.12. Simplified Service Oriented Model
(W3C Working Group, 2004)

Sementara itu terdapat arsitektur yang berorientasi pada sumberdaya informasi, model ini merupakan adopsi dari Arsitektur Web. Konsep hubungan yang terjalin adalah antara pengguna dan sumberdaya, seperti yang diilustrasikan pada gambar berikut ini.



Gambar 2.13. Simplified Resource Oriented Model
(W3C Working Group, 2004)

Yang terakhir adalah arsitektur yang berorientasi pada kebijakan, model ini menekankan pada prilaku agen atau yang bertanggung jawab terhadap ketersediaan akses sumberdaya, kualitas layanan, keamanan, manajemen, dan aplikasi. Bentuk ilustrasinya adalah seperti gambar berikut ini.



Gambar 2.14. Simplified Policy Oriented Model
(W3C Working Group, 2004)

Dari keempat orientasi model arsitektur tersebut, action dan resource merupakan orientasi yang mendapat perhatian cukup luas dari beberapa peneliti dalam penerapan Web Services. Pengembangan model yang diturunkan berdasarkan orientasi pada action/proses atau Service Oriented Model (SOM) menghasilkan model arsitektur berbasis layanan dan sering disebut dengan Services Oriented Architecture (SOA). Sementara itu pengembangan model yang diturunkan berdasarkan orientasi pada resource/sumberdaya informasi atau Resource Oriented Model (ROM) menghasilkan model arsitektur berbasis sumberdaya informasi atau sering disebut dengan Resource Oriented Architecture (ROA). Secara teknis dari kedua model tersebut memiliki kemiripan, karena masih merupakan bagian dari model Web Services.

SOM sendiri diadopsi oleh konsep SOA menjadi dasar teknologi yang dapat dibuktikan langsung melalui kelengkapan struktur, format dan utilitas message melalui dokumen XML. Struktur, format, dan utilitas yang terdapat pada XML menjadi standar bagi pengembangan SOA melalui metode Simple Object Access Protocol (SOAP), karena melalui standarisasi ini Web Services dengan

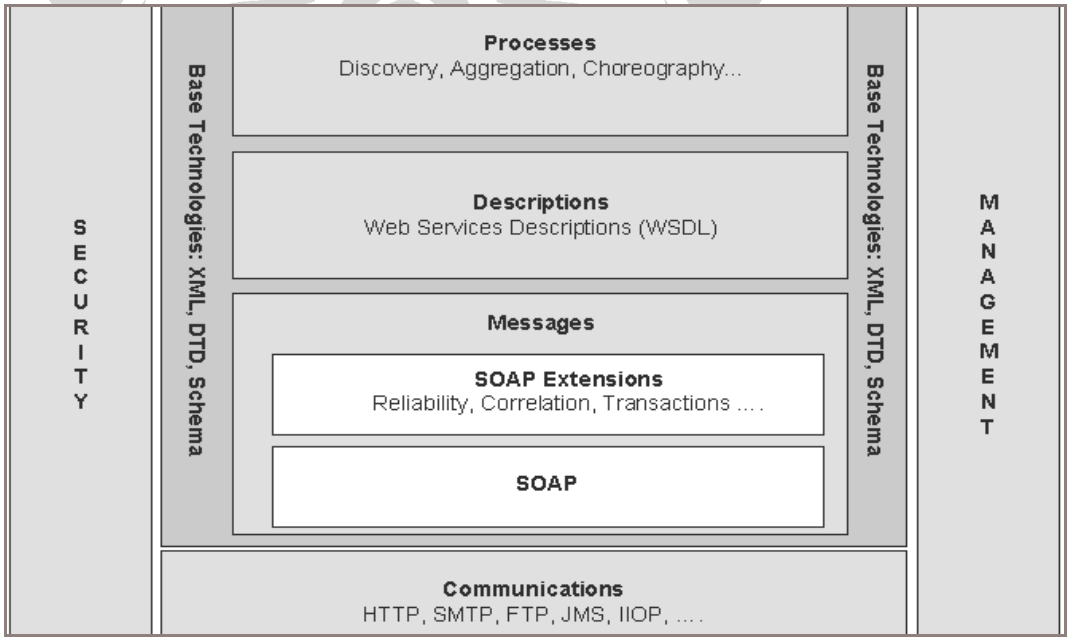
model SOA dapat dilakukan tidak hanya melalui protokol HTTP saja, tetapi dapat juga menggunakan protokol lain seperti SMTP, FTP, JMS, IIOP dan lain sebagainya. Definisi Service Oriented Architecture (SOA) menurut W3C Working Group adalah:

> ”A set of components which can be invoked, and whose interface descriptions can be published and discovered”.

Dalam perkembangannya, model Web Services memiliki 2 (dua) metode yang berorientasi pada Layanan dan Sumberdaya Informasi, yaitu:

1. Metode SOAP

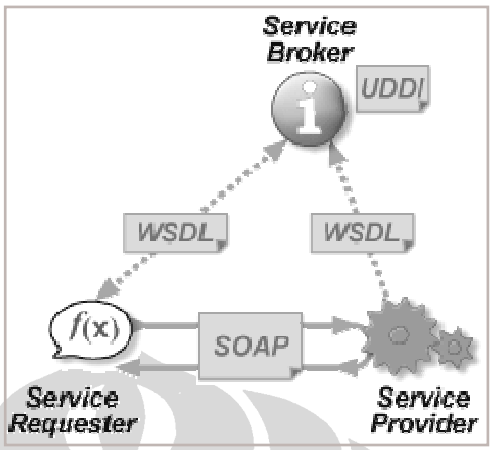
Impementasi Web Services dengan model SOA telah banyak dilakukan dan dikembangkan oleh banyak vendor seperti Microsoft, Sun dan IBM, melalui dukungan platform infrastruktur dotNet dan Java. Web Services dengan arsitektur model SOA yang implementasinya diwujudkan dalam bentuk metode Simple Object Access Protocol (SOAP), memiliki komponen-komponen dasar yang mendukung proses terlaksananya model aplikasi web berbasis layanan, yaitu berupa protokol komunikasi data, format messages (pesan/informasi), deskripsi layanan dalam bentuk semantik web, proses agregasi, keamanan sistem dan manajemen yang diilustrasikan pada gambar berikut ini.



Gambar 2.15. SOAP Web Services Stack

(W3C Working Group, 2004)

Adapun arsitektur yang digunakan dalam SOAP memiliki kemiripan dengan arsitektur Web Services yang diilustrasikan pada gambar berikut ini.



Gambar 2.16. Arsitektur SOAP

(Wikipedia, 2009)

Ilustrasi pada Gambar 2.16. menerangkan bahwa arsitektur SOAP memiliki 3 (tiga) komponen utama dalam melakukan proses layanannya yaitu meliputi:

a. Service Provider, sebuah node di network sebagai jasa layanan fungsional (services) yang dapat digunakan oleh Service Requester.
b. Service Requester, merupakan aplikasi yang menggunakan protokol SOAP Messages sehingga dapat berkomunikasi dengan Services Provider.
c. Service Broker, adalah perantara yang menyediakan layanan untuk penemuan dan penjelasan layanan publik secara terstruktur dan terintegrasi, hal ini bisa diumpamakan sebagai katalog layanan.

Selain komponen utama, untuk menggunakan arsitektur SOAP diperlukan komponen pendukung yang merupakan dasar terbentuknya mekanisme kerja Web Services. Komponen pendukung tersebut adalah (Pedoman Interoperabilitas, 2008):

a. Extensible Markup Language (XML), adalah sebuah bahasa universal yang dapat mempersatukan persepsi terhadap suatu informasi yang digunakan dalam pertukaran informasi.

b. SOAP-Envelope (SOAP-XML), merupakan sebuah paket terstruktur dalam format standar dokumen berbentuk XML yang digunakan untuk melakukan proses request dan response antara Web Services dengan aplikasi yang memanggilnya. Script SOAP-Envelope terbagi menjadi dua, yaitu SOAP Header dan SOAP Body. Pada SOAP-Header berisi informasi yang sifatnya tambahan, sedangkan SOAP-Body berisi informasi yang dipertukarkan antar Web Services. Skema dari dokumen SOAP-Envelope tersebut dapat dicontohkan pada script berikut ini.

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope>
      <SOAP-ENV:Header>

      </SOAP-ENV:Header>
      <SOAP-ENV:Body>
            <kurs>10000</kurs>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

c. Web Service Description Language (WSDL), adalah sebuah dokumen XML yang isinya menjelaskan informasi rinci sebuah Web Services berupa daftar fungsi (method), parameter yang diperlukan untuk menggunakan fungsi dan tipe data hasil penggunaan fungsi. WSDL memungkinkan client yang berbeda untuk secara otomatis mengerti bagaimana cara berinteraksi dengan Web Services. Adapun format dari WSDL yang umumnya digunakan adalah sebagai berikut:

```
<?xml version="1.0"?>
<definitions>
      <types>
      </types>
      <message>
      </message>
      <portType>

      </portType>
      <binding>

      </binding>
      <service>

      </service>
</definitions>
```

d. Universal Description, Discovery and Integration (UDDI), merupakan suatu registry service bersifat universal yang digunakan untuk mendaftarkan atau mengumpulkan (agregasi) dan menemukan penyedia Web Services. Secara umum UDDI menyediakan struktur untuk merepresentasikan layanan yang ada, hubungan antar layanan, spesifikasi metada serta akses point dari sebuah Web Services.

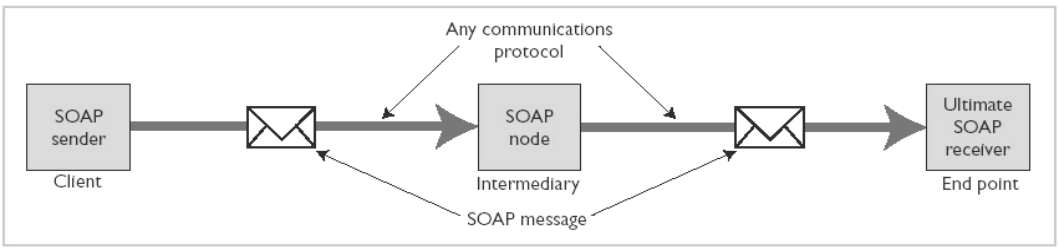
Pada prinsipnya data yang diolah oleh SOAP dianggap sebagai *message*, agar *message* ini dapat dimengerti oleh setiap entitas maka dibuatlah struktur data yang sama melalui format XML dalam bentuk dokumen Web Services Description Language (WSDL). Bentuk formal dari suatu message ini hanya bisa diolah dengan cara menggunakan utilitas teknologi SOAP-XML. Artinya dengan menggunakan protokol komunikasi apapun, transaksi dengan metode SOAP harus menggunakan utilitas teknologi SOAP-XML yang umumnya pada open source tersedia dalam bentuk pustaka fungsi (library) untuk PHP dengan nama SOAP dan NuSOAP, karena dengan utilitas ini data message yang dikemas dalam format SOAP-Envelope dapat diolah oleh sebuah interface yang digunakan untuk melayani proses komunikasi antar aplikasi. Utilitas library SOAP maupun NuSOAP terbagi menjadi dua, yaitu:

a. SOAP Server, ditempatkan pada Web Services Provider sebagai penerima permintaan dan pelaksana proses.
b. SOAP Client, digunakan pada Service Requester untuk berinteraksi dengan Web Services Provider.

Masing-masing pihak yang melakukan interaksi dengan metode SOAP, harus memiliki utilitas library SOAP yang ditempatkan pada masing-masing aplikasinya. Utilitas SOAP yang digunakan pun harus yang sama jenis format dan keluaran vendornya, karena masing-masing vendor SOAP belum tentu memiliki konsep dan format yang sama contohnya antara PHP-SOAP, PHP-NuSOAP, dan Microsoft SOAP.

Bila diperhatikan pada Gambar 2.17., berdasarkan mekanisme proses dan format dokumen yang digunakan baik WSDL maupun SOAP-Envelope, maka dapat disimpulkan bahwa pendekatan yang dilakukan metode SOAP lebih mirip

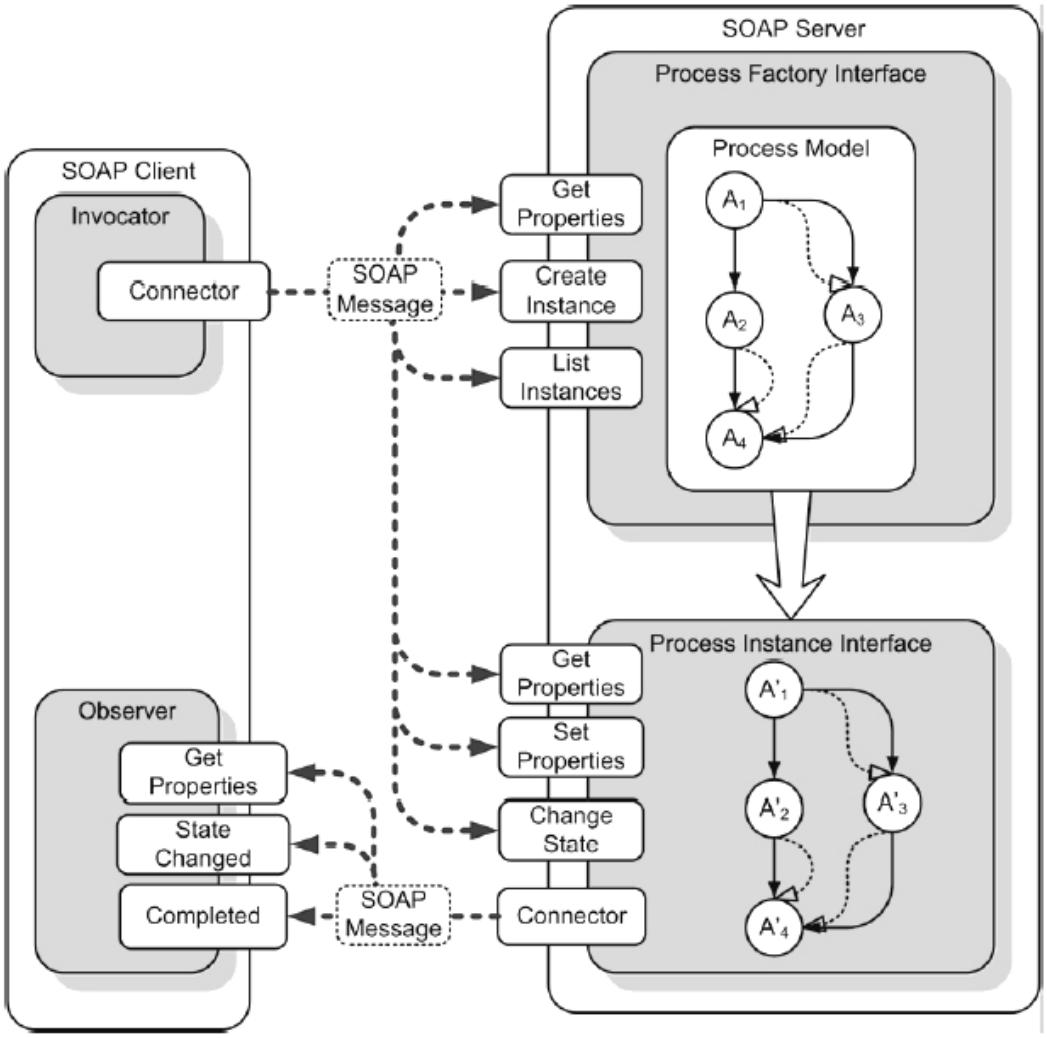
dengan konsep Distributed Object Computing yang dimiliki oleh teknologi CORBA, DCOM ataupun RMI.



Gambar 2.17. SOAP Transport Independence
(Vogels, 2003)

Hal ini disebabkan karena awalnya metode SOAP sendiri dibuat sebagai alternatif Distributed Object yang berjalan pada infrastruktur Web Server. Sejarahnya dimulai pada tahun 1998, sebagai inisiatif pengembangan teknologi Microsoft mencari cara untuk membuat DCOM bisa berjalan di atas infrastruktur Web, sehingga dirancanglah konsep SOAP versi 1.0 sebagai RPC middleware yang menggunakan protokol Web. Tak lama kemudian IBM pun bergabung dalam upaya mengembangkan SOAP versi 1.1. yang menjadi acuan W3C. Pada tahun 2003 W3C mengambil alih SOAP dan merilis SOAP versi 1.2. (Jakl, 2006).

Secara eksplisit proses interaksi antara SOAP-Client dan SOAP-Server dengan metode SOAP tersebut dapat terlihat pada gambar 2.18., dimana setiap SOAP-Client yang akan melakukan komunikasi data dengan SOAP-Server terlebih dahulu melewati fungsi invocator atau konektor untuk mengambil nilai hasil fungsi (action) pada server. Pesan (SOAP-Message) berupa nama fungsi dan parameter nilai yang diminta dikirim oleh invocator. Saat fungsi ini sampai di server, maka SOAP-Server akan membentuk jalinan dengan objek fungsi (Create Instance) yang terdapat dalam daftar fungsi (List Instances), kemudian jalinan tersebut akan mengambil properti fungsi fisik (Get Properties) yang terdapat dalam program melalui metode Get (fungsi yang bertugas untuk mengembalikan nilai suatu proses), Set (fungsi yang hanya bertugas melakukan proses tanpa mengembalikan nilai), dan Change State (fungsi even driven yang menunggu kondisi kejadian suatu proses). Hasil proses fungsi akan dikembalikan kepada SOAP-Client dalam bentuk SOAP-Message melalui fungsi event-driven State Change yang berfungsi sebagai observer/listener.



Gambar 2.18. Metode SOAP Web Services

(Michael zur Muehlen, 2004)

2. Metode REST

Disamping metode SOAP, terdapat metode lain yang berorientasi pada sumberdaya informasi (resource) dalam Web Services. Dalam desertasinya tentang Architectural Style, Roy Thomas Fielding mencoba menemukan konsep Web Services dengan metode yang diberi nama REpresentational State Transfer (REST), dengan definisi sebagai berikut:

> *"Representational State Transfer is intended to evoke an image of how a well-designed Web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing*

the next state of the application) being transferred to the user and rendered for their use."

Definisi lain tentang REST dikemukakan pula oleh W3C Working Group, yaitu:

"REST Web is the subset of the WWW (based on HTTP) in which agents provide uniform interface semantics -- *essentially create, retrieve, update and delete -- rather than arbitrary or application-specific interfaces, and manipulate resources only by the exchange of* representations. *Furthermore, the REST interactions are "stateless" in the sense that the meaning of a message does not depend on the state of the conversation."*

Metode REST didasari oleh empat prinsip utama teknologi, yaitu (Pautasso, 2008):

a. *Resource identifier through Uniform Resource Identifier (URI)*, REST Web Services mencari sekumpulan sumberdaya yang mengidentifikasi interaksi antar client.
b. *Uniform interface*, sumberdaya yang dimanipulasi CRUD (Create, Read, Update, Delete) menggunakan operasi PUT, GET, POST, dan DELETE.
c. *Self-descriptive messages*, sumberdaya informasi tidak terikat, sehingga dapat mengakses berbagai format konten (HTML, XML, PDF, JPEG, Plain text dan lainnya). Metadata pun dapat digunakan.
d. *Stateful interactions through hyperlinks*, setiap interaksi dengan suatu sumberdaya bersifat stateless, yaitu request messages tergantung jenis kontennya.

Metode REST Web Services dianggap sederhana karena menggunakan format standar yang umum seperti HTTP, HTML, XML, URI, MIME. Penerapan REST Web Services sama dengan membangun web site dinamis, dengan kehandalan yang diharapkan dalam melakukan uji coba cukup melalui aplikasi Web Browser tanpa membutuhkan software khusus. Namun jika diperlukan untuk proses pengambilan data, maka konten hasil eksekusi Web Services berupa teks dapat digunakan sebagai data yang dapat diolah melalui pembentukan struktur

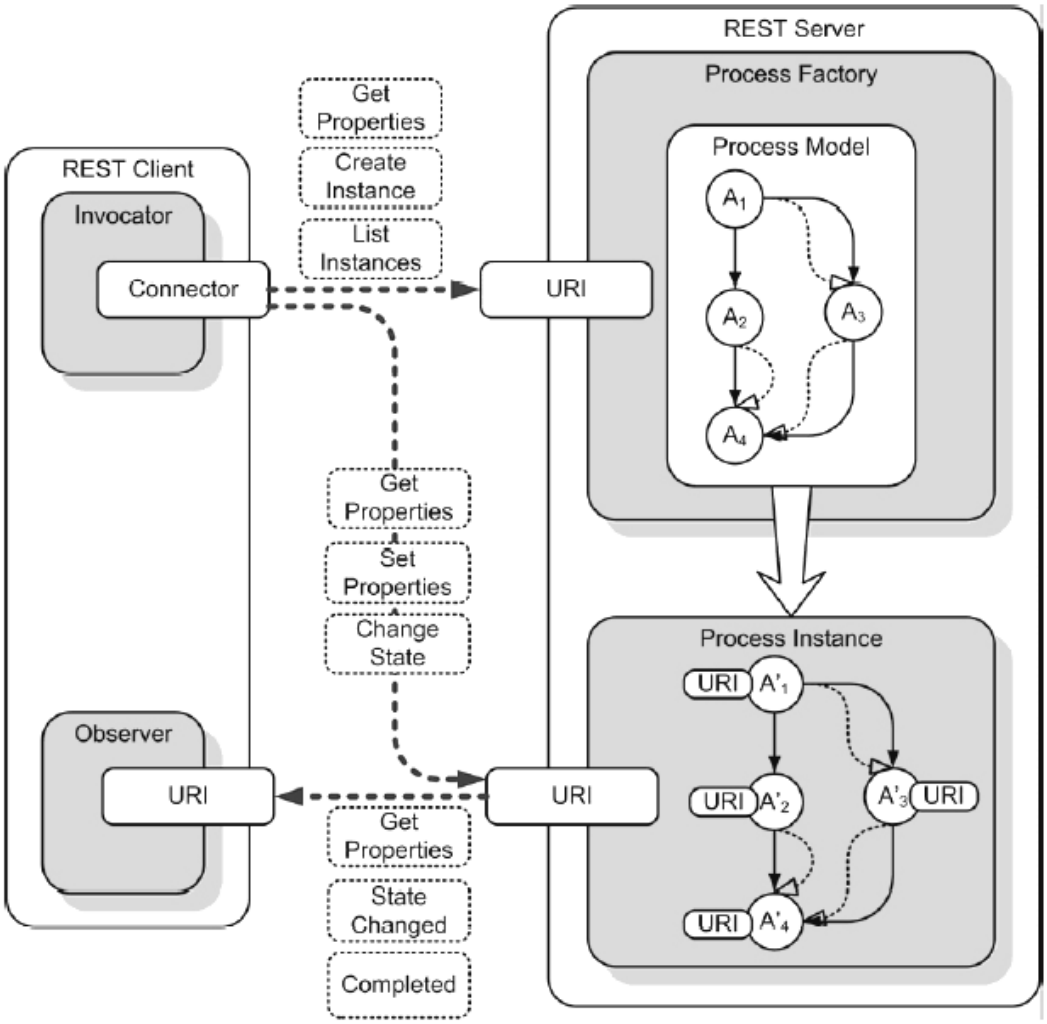
data dalam berbagai format teks, seperti XML atau HTML. Hal ini dapat dilakukan dalam kode program yang menggunakan utilitas komunikasi data melalui koneksi socket protokol HTTP. Umumnya utilitas ini tersedia dalam pustaka komunikasi pada beberapa bahasa pemrograman seperti Java, Visual Basic, Delphi, PHP, ASP, maupun JSP.

Pustaka komunikasi yang tersedia pada beberapa bahasa pemrograman tidak seluruhnya mampu untuk melewatkan parameter URI dengan metode POST, apalagi bila Web Services yang dituju berada pada protokol HTTPs. Hanya beberapa bahasa pemrograman yang telah menyediakan pustaka komunikasi data dengan kemampuan yang optimal. Sebut saja salah satunya adalah pustaka komunikasi data yang dimiliki PHP, meskipun PHP sendiri menyediakan pustaka SOAP namun dengan konsep Open Source pustaka lain yang mendukung REST juga tersedia dengan nama Client URL (CURL).

CURL bukan satu-satunya pustaka komunikasi data di dalam PHP, masih banyak lagi pustaka lain yang menyediakan kemampuan untuk komunikasi data. Namun pada umumnya CURL sudah terbukti dapat digunakan untuk memenuhi proses komunikasi data di beberapa protokol, seperti HTTP, HTTPS, FTP, GOPHER, TELNET, DICT, FILE, dan LDAP. Berdasarkan kemampuan yang dimiliki CURL, banyak pustaka lain yang menggunakan pustaka fungsi CURL sebagai dasar proses komunikasi data, salah satu pustaka fungsi yang menggunakannya adalah NuSOAP yang berjalan diatas platform PHP.

Dengan demikian penggunaan pustaka fungsi komunikasi data seperti PHP-CURL dapat mewujudkan mekanisme REST Web Services. Namun untuk menggunakan fungsi-fungsi yang dipublikasikan melalui Web Services dengan metode REST, bisa dilakukan melalui pustaka fungsi komunikasi data apa saja yang tersedia pada masing-masing bahasa pemrograman. Karena pada dasarnya metode REST tidak mempermasalahkan utilitas apa yang digunakan untuk berkomunikasi, melainkan konten apa yang dapat diproses melalui HTTP atau HTTPs. Secara eksplisit proses interaksi antara REST-Client dan REST-Server pada metode REST dapat terlihat dalam gambar 2.19. Bila dibandingkan metode SOAP, maka metode REST memiliki kesamaan yang hampir mirip. Namun karena REST tidak menggunakan encapsulated data atau model SOAP-Envelope,

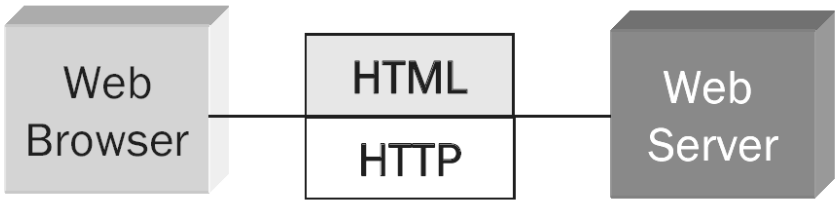
dengan demikian REST-Client hanya perlu menyertakan parameter berupa nama fungsi yang akan diproses dan nilai parameter yang diminta secara langsung dalam bentuk URI. Pengiriman URI ini dapat dilakukan dengan dua metode umum, yaitu POST dan GET. Perbedaannya kalo POST dilakukan secara implisit pada URL, tapi GET dilakukan secara eksplisit pada URL. Melalui metode ini REST-Server cukup membaca metode yang dilewatkan kedalam URI, kemudian menterjemahkan seluruh isi parameter berupa nama umum suatu fungsi yang dipetakan ke dalam fungsi sebenarnya yang diproses berdasarkan nilai parameter permintaan.



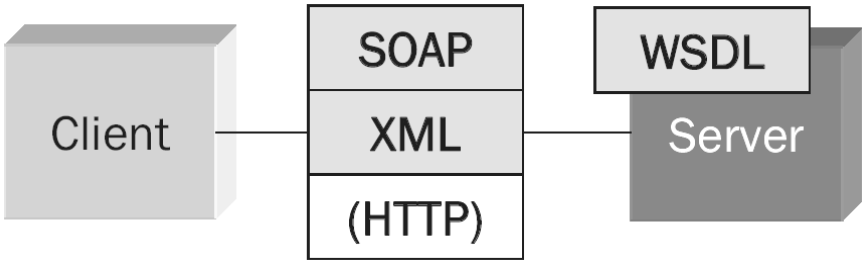
Gambar 2.19. Metode REST Web Services

(Michael zur Muehlen, 2004)

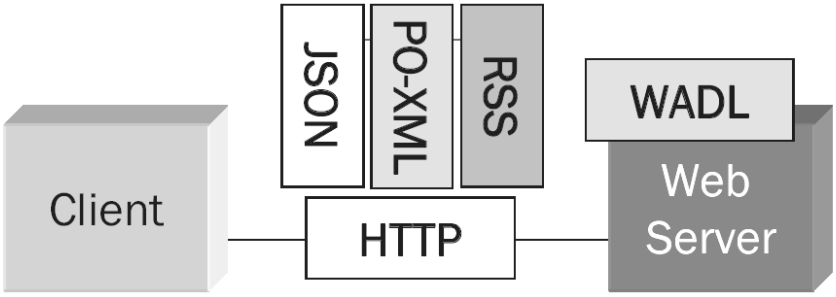
Perbandingan mendasar antara metode REST dengan SOAP Web Services dijelaskan oleh Cesare Pautasso dalam penelitian tentang membuat keputusan arsitektur yang tepat antara REST dan SOAP. Melalui metode keputusan arsitektural dari beberapa alternatif arsitektur, Cesare Pautasso membandingkan REST dan SOAP menjadi beberapa bagian, yaitu berdasarkan prinsip arsitektur, keputusan konseptual, dan keputusan teknologi. Pemaparan hasil penelitian tersebut kemudian disosialisasikan dalam Seminar SOA (SOA Symposium) tahun 2008 di Amsterdam. Ilustrasi yang disampaikan diawali dengan sejarah arsitektur Web Services seperti pada gambar berikut ini.



Gambar 2.20. Arsitektur Website Tahun 1992
(Pautasso, SOA Symposium, 2008)

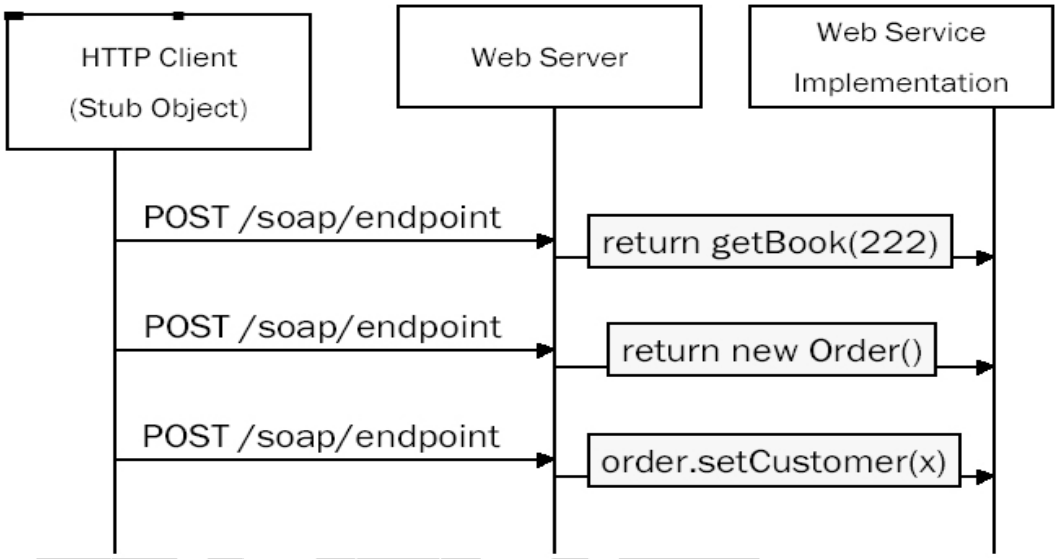


Gambar 2.21. Arsitektur SOAP - Web Services Tahun 2000
(Pautasso, SOA Symposium, 2008)

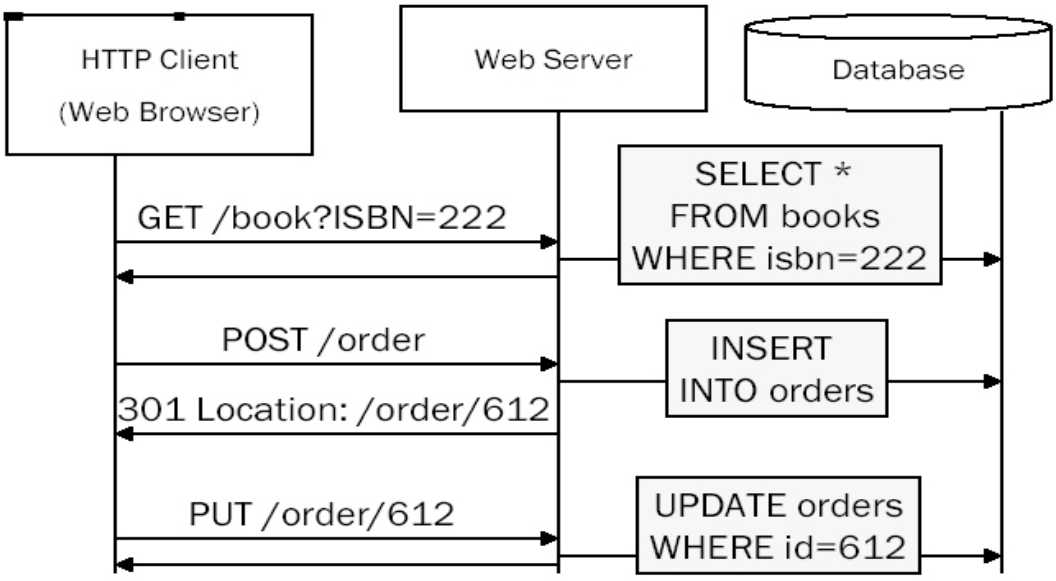


Gambar 2.22. Arsitektur REST - Web Services Tahun 2006
(Pautasso, SOA Symposium, 2008)

Berdasarkan komposisi arsitektur yang terdapat pada Gambar 2.21 dan Gambar 2.22., mekanisme proses transaksi data antara metode REST dan SOAP secara sederhana dapat dijelaskan melalui Gambar 2.23 dan Gambar 2.24 berikut.



Gambar 2.23. Mekanisme SOAP Web Services
(Pautasso, 2008)



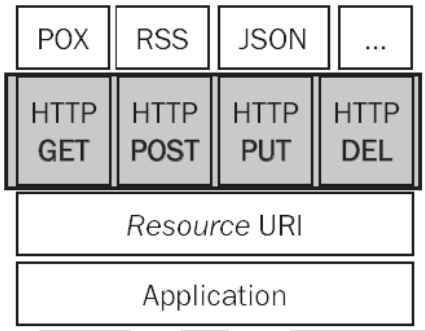
Gambar 2.24. Mekanisme REST Web Services
(Pautasso, 2008)

Perbedaan mekanisme tersebut mempengaruhi perbedaan prinsip arsitektur yang dimiliki oleh REST dan SOAP, diantaranya meliputi:

1. Protocol Layering

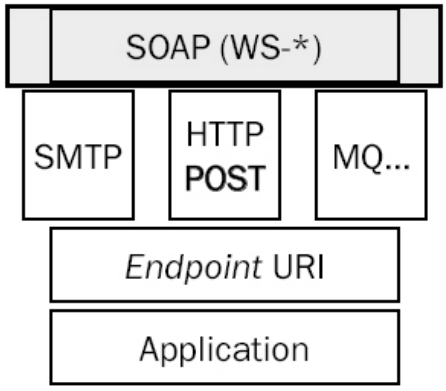
   Secara prinsip metode REST dan SOAP memiliki sudut pandang yang berbeda terhadap penggunaan protokol komunikasi data. Perbedaan tersebut adalah:

   - Metode REST Web Services menganggap penggunaan protokol HTTP sebagai protokol aplikasi (Application-level Protocol). Hal ini dapat diilustrasikan pada Gambar 2.25.



Gambar 2.25. REST: the Web is the universe of globally accessible information
(Pautasso, SOA Symposium, 2008)

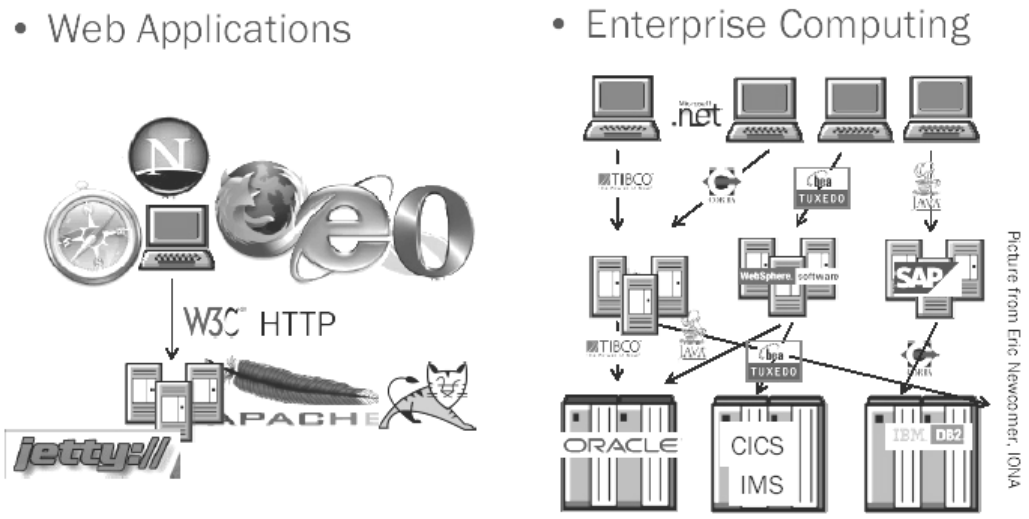
   - Metode SOAP Web Services menganggap penggunaan protokol, khususnya HTTP sebagai protokol transport (Transport-level Protocol). Hal ini dapat diilustrasikan pada Gambar 2.26.



Gambar 2.26. SOAP: the Web is the universal (tunelling) transport for messages
(Pautasso, SOA Symposium, 2008)

2. Dealing with Heterogeneity

   SOAP maupun REST Web Services memiliki kesamaan dalam hal penanganan keragaman komponen pada protokol HTTP. Namun terjadi kompetisi dukungan antar vendor aplikasi browser dan enterprise computing, sebagaimana diilustrasikan pada Gambar 2.27 berikut ini.

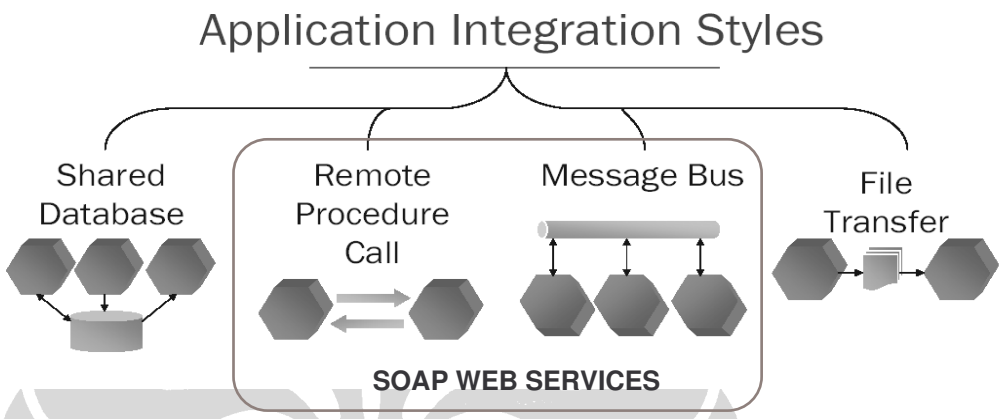


Gambar 2.27. Platform pendukung SOAP dan REST Web Services
(Pautasso, SOA Symposium, 2008)

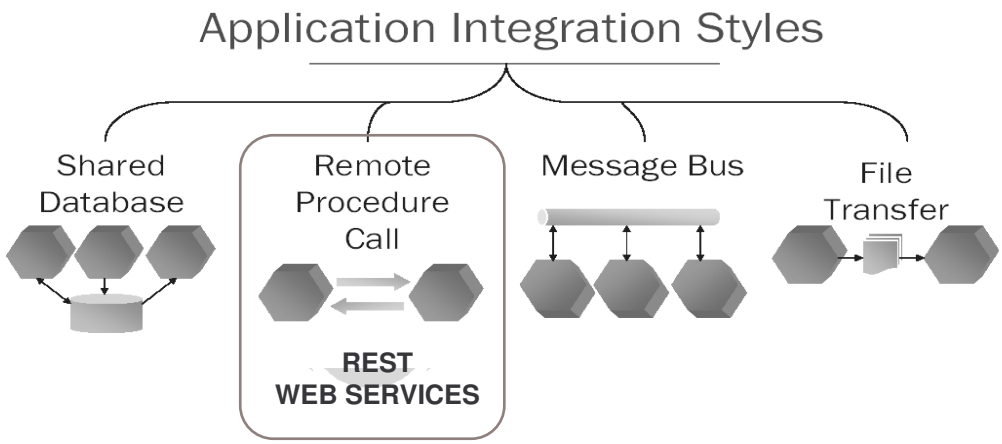
3. Loose Coupling

   Berdasarkan aspek time/availability dan location transparency, metode REST dan SOAP dianggap memiliki kecenderungan menjadi Loose Coupling (bebas ketergantungan akses). Namun dalam aspek Service Evolution, yaitu perubahan pada struktur maupun tipe data fungsi dalam Web Services yang tidak mempengaruhi deskripsinya, metode REST memiliki Loose Coupling yang cukup tinggi dibandingkan metode SOAP. Karena metode REST bebas dari bentuk format deskripsi dan hanya menggunakan format URI, sedangkan pada metode SOAP, perubahan yang terjadi dalam struktur fungsi Web Services akan mempengaruhi deskripsi Web Services yang ada dalam WSDL, sehingga WSDL pun harus dirubah untuk menyamakan struktur dan tipe datanya. Dengan adanya ketergantungan pada perubahan deskripsi ini, maka metode SOAP menjadi lebih kompleks dalam hal pengembangannya. Sebaliknya metode REST yang membebaskan pendefinisian pada URI menyebabkan pengembangannya dapat dilakukan lebih cepat dan sederhana.

Perbandingan berikutnya adalah berdasarkan keputusan konseptual yang terdapat pada metode REST dan SOAP. Pola integrasi masing-masing metode memiliki konsep arsitektur yang berbeda, REST hanya membutuhkan satu komponen, yaitu Remote Procedure Call (RPC) sedangkan SOAP membutuhkan dua komponen, yaitu RPC dan Messaging (Message Bus). Ilustrasi perbedaan tersebut tergambar dalam Gambar 2.28. dan 2.29.



Gambar 2.28. Pola Arsitektur Integrasi untuk metode SOAP Web Services
(Pautasso, SOA Symposium, 2008)



Gambar 2.29. Pola Arsitektur Integrasi untuk metode REST Web Services
(Pautasso, SOA Symposium, 2008)

Pada Pola Arsitektur Integrasi masing-masing metode menghasilkan kemampuan proses perancangan (design process) yang berbeda. Metode REST memiliki proses Resource Indentification (Exposing), URI design (Scheme), Resource Interaction Semantics (POST/GET), Resource Relationships, Data Representations/Modelling (unmandatory XML), Message Exchange Patern

(Request-response). Sedangkan metode SOAP hanya memiliki tiga proses, yaitu Data Modelling (XML Schema), Message Exchange Patterns, dan Service Operations Enumeration (WSDL).

Perbandingan terakhir yang dilakukan Cesare Pautasso adalah pada keputusan teknologi yang dimiliki metode REST dan SOAP. Masing-masing metode dibandingkan dalam beberapa alternatif arsitektur dan keputusan arsitektur teknologi seperti Transport Protocol, Payload Format, Service Identification, Service Description, Reliability, Security, Transaction, Service Composition, Service Delivery dan Implementation Techology. Hasil perbandingan tersebut menunjukan metode REST memiliki 10 (sepuluh) keputusan dari 17 (tujuh belas) atau lebih alternatif teknologi, sedangkan metode SOAP memiliki 10 (sepuluh) keputusan dari 25 (dua puluh lima) atau lebih alternatif teknologi. Dengan demikian dalam hal pemilihan teknologi metode REST lebih efisien ketimbang metode SOAP.

Dengan banyaknya keputusan dalam pilihan teknologi maka akan semakin banyak ketergantungan terhadap utilitas teknologi. Dengan demikian metode SOAP memiliki kecenderungan berpeluang Lock-In terhadap vendor teknologi. Sebaliknya melalui penggunaan teknologi Web yang bersandar pada teknologi standar terbuka, maka metode REST sangat berpeluang memenuhi kriteria untuk dijadikan model interoperabilitas meskipun hanya bisa diakses melalui protokol komunikasi HTTP saja. Namun dengan perkembangan teknologi informasi, protokol komunikasi HTTP yang telah menjadi teknologi standar terbuka akan diadopsi dan digunakan sebagai standar komunikasi yang universal. Dengan demikian metode REST berperluang untuk bebas Lock-In dari ketergantungan terhadap vendor teknologi.

Michael zur Muehlen (2005) mengungkapkan hasil survei yang dilakukan oleh Amazon.com, bahwa untuk mengembangkan Web Services ditemukan sebanyak 85% developer menggunakan metode REST, dan hanya 15% yang menggunakan metode SOAP.

Menurut hasil penelitian yang dilakukan Michael zur Muehlen (2005), perbandingan kelebihan dan kekurangan antara metode SOAP dan REST

dilakukan melalui pendekatan pada proses integrasi yang diuraikan dalam tabel berikut ini.

Tabel 2.3. Karakteristik REST dan SOAP

	REST	SOAP
Characteristics	• Operations are defined in the Messages • Unique address for every process Instance • Each object supports the defined (standard) operations • Loose coupling of components	• Operations are defined as WSDL ports • Unique address for every operation • Multiple process instances share the same operation • Tight coupling of components
Self-declared advantages	• Late binding is possible • Process instances are created explicitly • Client needs no routing information beyond the initial process factory URI • Client can have one generic listener interface for notifications	• Debugging is possible • Complex operations can be hidden behind façade • Wrapping existing APIs is straightforward increased privacy
Posible disadvantages	• Large number of objects • Managing the URI namespace can become cumbersome	• Client needs to know operations and their semantics beforehand • Client needs dedicated ports for different types of notification • Process instances are created implicitly

2.6. KONEKTOR WEB SERVICES MELALUI PUSTAKA FUNGSI PHP CLIENT URL

PHP merupakan bahasa script pemrograman berbasis Open Source untuk pengembangan aplikasi web yang dapat menyisipkan kode HTML. PHP sendiri dapat berjalan di atas platform Aplikasi Web Server apapun terutama Apache dan Micorosoft Internet Information Server (IIS). Melalui pemrograman script PHP aplikasi berbasis web khususnya untuk pengembangan Web Services akan menjadi lebih mudah dikembangkan karena konsep kesederhanaan kode perintah yang terdapat dalam PHP.

Terkait dengan masalah pengembangan Web Services, PHP memiliki dukungan pustaka fungsi yang dapat digunakan sebagai utilitas untuk berkomunikasi dan berinteraksi dengan aplikasi web lainnya melalui metode Remote Procedure Call (RPC). Konsep RPC merupakan dasar dari komunikasi antar sistem aplikasi secara point-to-point atau peer-to-peer. Metode Web Services sebenarnya mengadopsi dari konsep RPC, sehingga Web Services dapat dianggap RPC berbasis web.

Client URL (CURL) merupakan pustaka fungsi yang dibuat oleh Daniel Stenberg untuk dapat terhubung dan berkomunikasi dengan aplikasi web lain melalui berbagai server dan protokol komunikasi. Kemampuan CURL yang digunakan oleh bahasa pemrograman PHP merupakan fasilitas tambahan untuk memperkaya dan memperkuat pemrograman aplikasi web secara dinamis.

Contoh perintah sederhana dari PHP CURL yang dapat digunakan untuk berkomunikasi dengan aplikasi web lain adalah sebagai berikut:

Program 2.1. Contoh Konektor Web Services

```
<?php
  $url="http://www.contoh.com/";       // Alamat website
  $param="rupiah=250000";              // Parameter yang dilewatkan
  $ch = curl_init();
  curl_setopt($ch,CURLOPT_URL, $url);  // Inisialisasi URL
  curl_setopt($ch,CURLOPT_POST, TRUE); // Metode: POST
  curl_setopt($ch,CURLOPT_RETURNTRANSFER,TRUE); // Ada nilai balik
  curl_setopt($ch,CURLOPT_POSTFIELDS,$param); //Parameter terkirim
  $result=curl_exec($ch); // Eksekusi
  echo $result;  // Hasil keluaran dapat dicetak langsung
?>
```

Setelah melalui eksekusi terhadap alamat website (URL) dengan melewatkan parameter melalui metode POST, maka konten hasil proses yang diterima dalam variabel dapat dicetak untuk menampilkan isi datanya. Sebenarnya konsep RPC berbasis Web tidak mengembalikan data bertipe tertentu seperti pada platform Sistem Operasi, namun dengan menggunakan konsep dasar Arsitektur Web, maka semua data yang dikomunikasikan adalah bertipe teks. Dengan

demikian data yang diterima dapat langsung ditampilkan atau diolah melalui fungsi penterjemahan format data apabila data yang diterima memiliki format tertentu, misalnya yang lebih formal adalah format XML.

2.7. PEMODELAN RANCANGAN APLIKASI

Perancangan aplikasi untuk Model Interoperabilitas Sistem Informasi dibuat secara terstruktur dengan menggunakan dasar-dasar perancangan pemodelan. Karena tidak mungkin untuk memahami sistem secara terperinci terlebih lagi bila sistemnya sangat kompleks, maka dari itu pemodelan merupakan cara penyederhanaan realitas yang mendeskripsikan sebuah sistem dari perspektif tertentu. Dengan adanya pemodelan rancangan aplikasi diharapkan dapat memberikan pemahaman yang lebih baik tentang aplikasi yang akan dibuat.

Sebuah alat bantu yang digunakan untuk pemodelan rancangan aplikasi adalah berupa dokumen standar cetak biru dalam bentuk bahasa pemodelan. UML merupakan salah satu alat bantu dengan bahasa pemodelan visual yang digunakan untuk memvisualisasi, menspesifikasi, membangun dan mendokumentasikan hasil dari sebuah sistem intensif software. Meskipun UML memilki cukup banyak menyediakan diagram yang membantu mendefinisikan dan mendeskripsikan sebuah aplikasi yang sedang dibangun, namun tidak berarti bahwa semua diagram tersebut bisa menjawab persoalan yang ada. Mungkin saja diagram selain UML akan lebih cocok untuk pemodelan visual, sesuai dengan ruang lingkup pemodelan yang akan digunakan.

Visualisasi UML dikategorisasikan dalam beberapa jenis diagram yang umum digunakan seperti dalam tabel berikut ini.

Tabel 2.4. Jenis Diagram UML

Diagram	Tujuan
Activity	Perilaku prosedural dan paralel.
Class	Visualisasi Class dan relasinya.
Sequence	Interaksi antar obyek, lebih menekankan pada urutan.
Use Case	Bagaimana pemakai berinteraksi dengan sebuah sistem.