PEMBUATAN PROTOTIPE APLIKASI WEB SERVICES BERBASIS XML MENGGUNAKAN TEKNOLOGI J2EE DENGAN STUDI KASUS RESERVASI HOTEL

Isye Arieshanti, Joko Lianto B, Waskitho Wibisono
Jurusan Teknik Informatika,
Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember
Kampus ITS, Jl. Raya ITS, Sukolilo – Surabaya 60111, Telp. + 62 31 5939214, Fax. + 62 31 5913804
Email : isye@its-sby.edu

ABSTRAK

Dalam era globalisasi, para pelaku bisnis secara intensif melakukan usaha-usaha untuk memasuki pasar global. Suatu perusahaan semakin membutuhkan transaksi bisnis yang bersifat fleksibel, yang bisa dilakukan dengan siapa saja, kapan saja dan dimana saja. Tentunya sistem informasi yang dimiliki perusahaan tersebut harus bisa berkomunikasi dengan sistem yang dimiliki oleh patner bisnis, tanpa harus terlalu banyak perjanjian dan persetujuan. Hal ini berarti diperlukan standard infrastruktur sederhana untuk pertukaran data bisnis.Kebutuhan ini dapat dipenuhi oleh teknologi web service sebagai teknologi yang menyediakan infrastruktur sederhana bagi pelaku bisnis untuk berkomunikasi melalui pertukaran pesan XML.

Pada Penelitian ini dikembangkan sebuah prototipe aplikasi web service dengan studi kasus reservasi hotel melalui perantara broker. Studi kasus ini dipilih karena dapat merepresentasikan sistem yang terdistribusi. Dimana broker berperan sebagai penghubung antara customer dan beberapa sistem yang terdistribusi.Pada pembuatan aplikasi ini dipilih teknologi J2EE karena framework J2EE yang telah ada mendukung penerapan web service. Dan selain itu, J2EE bersifat netral terhadap berbagai macam platform (tidak tergantung pada platform tertentu). Dan pada akhirnya prototipe ini diharapkan menjadi solusi alternatif dalam masalah komunikasi transaksi bisnis antara perusahaan dengan perusahaan lain.

Kata kunci : *web service*

1. PENDAHULUAN

Dengan semakin besarnya kebutuhan pelaku bisnis untuk melakukan transaksi secara fleksibel dengan siapa saja, kapan saja dan dimana saja, dibutuhkan suatu kolaborasi yang memungkinkan pelaku bisnis berhubungan dengan partner bisnis(business to business/B2B) maupun dengan customernya(business to consumer/B2C) secara mudah, aman dan efisien. Kolaborasi ini membutuhkan integrasi antara beberapa sistem yang berbeda dan terdistribusi.

Integrasi sistem terdistribusi ini bukan merupakan hal yang mudah mengingat aplikasi suatu sistem bisa berbeda dengan aplikasi pada sistem yang lain. Misalnya system perusahaan A adalah aplikasi yang berplatform .NET, sedangkan system perusahaan B adalah aplikasi yang berplatform Java. Adanya perbedaan ini menuntut integrasi system yang bisa memberikan solusi untuk permasalahan :

- Otomatisasi pertukaran data transaksional secara real time
- Perbedaan format file, protocol dan standard security
- Standard umum yang memungkinkan sebuah perusahaan dan partnernya bias saling mengirim transaksi menggunakan kombinasi aplikasi, format file, jalur komunikasi, protocol komunikasi, protocol B2B dan standard XML.
- Scalable, secara horizontal dan vertical.

Permasalahan-permasalahan ini membutuhkan pengembangan teknologi-teknologi baru yang diharapkan mampu memberikan solusi. Salah satunya adalah teknologi *web services*.

Untuk mendeskripsikan solusi web services untuk permasalahan sistim terdistribusi, pada pembahasan ini akan dibuat prototype aplikasi web services menggunakan teknologi J2EE dengan studi kasus reservasi hotel broker. Dengan penerapan web service, prototipe ini diharapkan menjadi solusi alternatif bagi perusahaan-perusahaan untuk berkomunikasi dalam melakukan transaksi bisnis. Dan dengan penggunaan teknologi J2EE yang sangat mendukung web service, diharapkan pula adanya pemanfaatan web service secara lebih fleksibel. Adapun database yang digunakan pada pembuatan prototype ini adalah database yang didukung driver JDBC.

2. WEB SERVICES

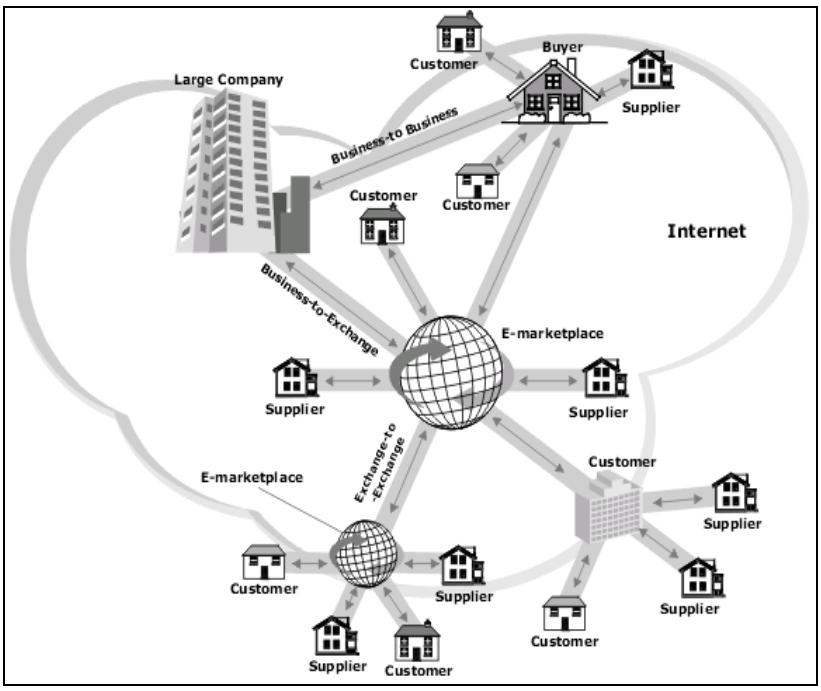
Web services adalah fungsi-fungsi bisnis (*services*) yang disediakan oleh entitas-entitas bisnis melalui koneksi internet sebagai sarana bagi entitas

bisnis lain untuk menggunakan *service* melalui pertukaran *message* yang berbasis XML. Sebuah entitas bisnis mengirim *request* kepada penyedia layanan dari URL yang diberikan dengan menggunakan protokol SOAP melalui HTTP .

Teknologi web services sendiri merupakan gabungan dari teknologi XML(eXtensible Markup Language), SOAP (Simple Object Access Protocol), WSDL(Web Services Description Language) dan UDDI (Universal Discovery, Description and Inventory).

*Web service*s memiliki karakteristik sebagai berikut :

- Berbasis XML.
- Berbasis pesan (*message*).
- Tidak tergantung pada bahasa pemrograman tertentu.
- Diakses melalui internet.
- *Loosely Coupled.*
  Dalam komunikasi antara pengguna *service* dan penyedia *service* hanya memerlukan kesamaan sintak dan *semantic* pesan XML. Sedangkan model *object*, bahasa pemrograman dan pemrograman API tidak harus sama. Sehingga *web service* bisa dikatakan sebagai *loosely coupled*.
- Berdasarkan pada standard industri.



Gambar 1. Sistem terdistribusi terintegrasi melalui web service

Karakteristik web services ini memungkinkan proses-proses bisnis seperti ordering, invoicing dan payment dilakukan melalui internet.

Untuk mendeskripsikan permasalahan sistim terdistribusi yang dipecahkan dengan solusi web services, maka pada pembangunan prototype ini diasumsikan sistim beberapa hotel sebagai sistim yang terdistribusi. Dan masing-masing sistim menyediakan web service untuk berbagi fungsi-fungsi yang mewakili proses bisnis reservasi dari masing-masing sistim. Fungsi-fungsi proses bisnis yang didefinisikan antara lain :

- Fungsi informasi kamar/ruang kosong yang bisa dipesan
- Fungsi reservasi
- Fungsi konfirmasi pembayaran
- Fungsi perubahan reservasi
- Fungsi pembatalan reservasi

Disini sistim hotel merupakan web services provider. Sedangkan pengguna web service (web services requester) adalah aplikasi broker yang akan melakukan pemanggilan fungsi-fungsi pada beberapa sistim hotel sekaligus dan memanfaatkan data tersebut untuk customer yang membutuhkan.

Seperti yang telah disebutkan pada pembahasan sebelumnya, bahwa teknologi web services merupakan gabungan dari teknologi XML, SOAP, WSDL dab UDDI. Pada bagian selanjutnya akan dijelaskan tentang masing-masing teknologi yang mendukung teknologi web services tersebut.

2.1. XML

XML adalah suatu standard industri untuk menampilkan data yang tidak tergantung pada sistem. Seperti HTML(HyperText Markup Language), XML melingkupi data dengan tag. Tetapi ada perbedaan antara tag HTML dengan tag XML. Di XML tag

merupakan arti dari data yang dilingkupi tag itu sendiri, sedangkan di HTML tag mendefinisikan cara menampilkan/display data yang dilingkupi oleh tag tersebut [7]. Contoh XML berikut menunjukkan daftar kamar pada suatu hotel dengan jenis dan harga sewa.

```
<priceList>
<room>
<type>Standard</ type >
<price>100</price>
</room>
<room>
<type>Deluxe</ type >
<price>200</price>
</room>
</priceList>
```

Tag<room> dan </room> memberitahu parser bahwa informasi diantara kedua tag tersebut adalah tentang kamar/room. Dua tag lain didalam tag <room> menentukan bahwa informasi yang berada dalam dua tag <room> adalah informasi tentang type kamar dan harga sewa kamar. Karena tag-tag XML mengindikasikan isi dan struktur data yang dilingkupinya, maka XML memungkinkan proses archiving dan searching.

Perbedaan lain antara HTML dan XML adalah tag XML dapat dikembangkan (extensible) sehingga developer dapat mendefinisikan sendiri tag XML untuk mendeskripsikan content yang dibutuhkan. Hal ini tentunya tidak ditemui pada HTML, dimana penggunaan tag harus berdasarkan pada spesifikasi HTML yang telah didefinisikan sebelumnya.

Dengan extensibility yang disediakan oleh XML, developer dapat membuat tag yang dibutuhkan untuk sebagian type dokumen. Pendefinisian tag menggunakan bahasa schema XML Sebuah schema mendeskripsikan struktur sekumpulan dokumen XML dan dapat digunakan untuk membatasi content dari dokumen XML. Bahasa schema yang paling luas digunakan adalah Document Type Definition (DTD). Contoh DTD berikut mendefinisikan tag-tag yang digunakan dalam document XML daftar harga kamar. DTD ini juga mendefinisikan struktur hirarki dari sebuah dokumen XML, termasuk urutan tag yang harus ditampilkan.

```
<!ELEMENT priceList (room)+>
<!ELEMENT room (type, price) >
<!ELEMENT type (#PCDATA) >
<!ELEMENT price (#PCDATA) >
```

Baris pertama dari contoh DTD diatas merupakan elemen level tertinggi pricelist, yang berarti tag-tag yang lain akan berada diantara tag <priceList> dan </priceList>. Element DTD baris pertama ini juga mendefinisikan bahwa element pricelist harus berisi satu atau lebih dari satu elemen room(ditandai oleh tanda '+').

Baris kedua mendefinisikan bahwa masing-masing elemen room harus berisi elemen type dan elemen price. Baris ketiga dan keempat, menentukan bahwa data diantara tag <type> dan </type> dan diantara tag <price> dan </price> adalah data character yang harus diparsing.

2.2. SOAP

SOAP adalah protokol ringan untuk pertukaran data pada sistem yang terdesentralisai dan terdistribusi. SOAP merupakan protokol berbasis XML yang terdiri dari 3 bagian, yaitu :

- Pembungkus(*envelope*) yang mendefinisikan sebuah kerangka kerja tentang apa yang ada didalam pesan(message) dan bagaimana pesan tersebut diproses.
- Himpunan aturan *encoding* untuk menjelaskan tipe data yang didefinisikan oleh aplikasi.
- Rumusan untuk menampilkan *request* dan *response* dari *procedure remote.*

Bagian pertama dari SOAP adalah envelope. Sebuah message SOAP harus terdiri dari sebuah envelope dan body. Sedangkan keberadaan header bersifat *optional*. Dalam dokumen XML, elemen envelope ditulis sebagai "Envelope". Elemen ini dapat berisi deklarasi *namespace* beserta atributnya. Elemen ini juga dapat berisi sub elemen. Anak elemen dari envelope adalah "Header". Elemen ini berisi sekumpulan header *entries* dan berfungsi untuk menjelaskan bagaimana memproses entry dan siapa yang bisa memprosesnya. Selain itu envelope berisi elemen "Body". Elemen ini dapat berisi body entries yang berfungsi menyediakan mekanisme pertukaran informasi serta untuk proses *marshalling* RPC dan *error report*.

Bagian kedua dari SOAP adalah aturan encoding yang mengatur tentang pendefinisian tipe data. Misalnya : sebuah array merupakan kumpulan nilai berdasarkan urutan tertentu, dsb. Selain array juga didefinisikan tipe-tipe data yang lain(struct, compound type dll).

Bagian terakhir dari SOAP adalah rumusan tentang *request* dan *response* RPC. Salah satu tujuan dari SOAP adalah enkapsulasi pertukaran pemanggilan RPC dengan memanfaatkan perluasan dan kefleksibelan dari XML. *request* dan *response* dari method RPC dibawa oleh elemen body dengan ketentuan sebagai berikut:

- *Request/response* dari method dimodelkan sebagai struct.
- *Request/response* dari method ditinjau sebagai struct tunggal yang berisi pengakses untuk masing-masing parameter in atau out

Penamaan parameter in atau out disesuaikan dengan tipe parameternya

2.3. WSDL

WSDL adalah sebuah format XML untuk menggambarkan *service-service* jaringan sebagai sekumpulan endpoint yang beroperasi pada pesan baik yang berorientasi *procedure* maupun dokumen. Operasi-operasi dan pesan-pesan dideskripsikan secara abstrak dan kemudian diikatkan pada protokol

jaringan nyata dan format pesan untuk mendefinisikan sebuah *endpoint*.

Didalam WSDL, definisi abstrak dari *endpoint-endpoint* dipisahkan dari pengembangan jaringan secara nyata serta terpisah pula dari pengikatan format data. Hal ini memungkinkan pemanfaatan ulang definisi abstrak *message* dan tipe-tipe *port*. *Message* merupakan pendeskripsi data yang dipertukarkan. Sedangkan tipe-tipe *port* merupakan kumpulan abstrak dari operasi-operasi. Sebuah *port* didefinisikan oleh sekumpulan alamat jaringan dengan pengikat yang dapat digunakan kembali. Sedangkan sekumpulan *port* mendefinisikan sebuah *service*.

Untuk mendefinisikan sebuah *service*, WSDL menggunakan elemen-elemen sebagai berikut:

- *Types*
- *Message*
- *Operation*
- *Port Type*
- *Binding*
- *Port*
- *Service*

Pada pembangunan prototype aplikasi web services reservasi hotel ini didefinisikan WSDLnya seperti pada gambar 2.

2.4. UDDI(UNIVERSAL, DESCRIPTION, DISCOVERY, AND INTEGRATION)

UDDI (*Universal, Description, Discovery, and Integration*) adalah sebuah protokol yang berfungsi untuk mendeskripsikan keberadaan method-method yang bisa diakses dalam *web services*. Standard ini memungkinkan pendaftaran entitas-entitas bisnis pada sebuah direktori di *internet*. Dalam direktori itu dideskripsikan jenis-jenis pelayanan(*service*) yang disediakan oleh perusahaan tersebut sehingga perusahaan yang lain dapat menemukan fungsi informasi atau fungsi yang diperlukan untuk *request* transaksi dengan perusahaan yang dikehendaki.

Setelah menemukan service yang dibutuhkan, maka selanjutnya dibuat koneksi dengan penyedia method. Koneksi ini ditangani oleh protokol SOAP. Dengan SOAP maka object dapat mengakses method object atau fungsi yang terletak pada *remote server*. Setelah method yang diminta dieksekusi oleh *remote server* maka, data hasil eksekusi dikirimkan kepada peminta melalui SOAP juga.

Untuk mendeskripsikan *web services* digunakan standard WSDL(*Web Service Description Language*). WSDL adalah service IDL (*Interface Definition Language*) berbasis XML yang berfungsi untuk mendefinisikan *interface service* serta karakteristik implementasinya. WSDL direferensi oleh entri-entri UDDI serta mendeskripsikan *messages* SOAP **[4]**.

Seperti yang telah disebutkan sebelumnya, teknologi J2EE sangat mendukung teknologi web service. Pada pembuatan prototype ini dimanfaatkan teknologi J2EE karena aplikasi yang dibuat nantinya tidak tergantung pada platform sistim operasi tertentu. Pada bagian berikut akan dideskripsikan dukungan J2EE dalam aplikasi web services.

2.5. J2EE (JAVA 2ND ENTERPRISE EDITION)

Dalam rangka komputasi sisi *server* dengan Java, Sun memproduksi *platform* yang disebut *Java 2 Platform, Enterprise Edition*. Misi dari J2EE adalah menyediakan suatu perangkat yang tidak tergantung pada *platform*, *portable*, *multi user*, aman, dan berkelas standar *enterprise* (bisnis) yang digunakan dalam pendistribusian sisi *server* yang ditulis dalam bahasa pemrograman Java. J2EE menyederhanakan banyak kekompleksan yang ada dalam pembuatan suatu aplikasi sisi *server* berbasis komponen [6].

Untuk *deployment web services* , J2EE mendukung teknologi XML dalam Java API for XML. Java APIs for XML ini terbagi menjadi 2 kategori. Kategori pertama bersifat *document-oriented*, berfungsi untuk memproses dokumen XML. Kategori kedua bersifat *procedure-oriented* yang berfungsi menangani *procedure-procedure*. Yang termasuk pada kategori pertama adalah JAXP (*Java API for XML Processing*). Dan yang termasuk pada kategori kedua adalah JAXM (*Java API for XML Messaging*), JAXR (*Java API for XML Registries*), dan JAX-RPC (*Java API for XML-based RPC*) [2].

Java APIs for XML tersebut sebenarnya merupakan Interface Logic dalam suatu aplikasi web service. Interface Logic disini didefinisikan sebagai pengirim, penerima dan pemroses pesan XML. Logic berikutnya adalah *Bussiness Logic yaitu logic* yang berfungsi mengimplementasikan service. Yang termasuk kategori Bussiness Logic adalah : EJB dan Java Object. Logic terakhir adalah *Integration Logic* yaitu logic yang berfungsi sebagai penghubung program dengan basis data atau dengan program lain. Contoh *Integration Logic* adalah JDBC dan JMS.

Pada bagian selanjutnya akan dideskripikan Java API for XML sebagai teknologi J2EE yang mendukung teknologi web service.

2.6. JAVA API FOR XML PROCESSING (JAXP)

JAXP adalah *library* untuk memproses dokumen-dokumen XML dengan standard parser seperti SAX (*Simple API for XML Parsing*) dan DOM (*Document Object Model*). JAXP juga mendukung XSLT (*XML Stylesheet Language Transformations*). SAX adalah *parser* yang membaca dokumen XML dari awal sampai akhir secara berurutan dan akan memproses setiap sintax yang dikenali. DOM adalah satu set *interface* untuk membangun sebuah representasi *object*. DOM dapat melakukan manipulasi *object* seperti *insert* dan *remove* sebagaimana memanipulasi struktur data tree yang lain. Sehingga dengan karakteristik seperti itu, DOM mampu mengakses bagian-bagian tertentu dari dokumen XML. XSLT adalah aturan yang digunakan

untuk merubah dokumen XML menjadi dokumen XML yang lain atau menjadi format data yang lain. XSLT juga menentukan bagaimana dokumen XML akan ditampilkan [2].

```
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://localhost:8080/hotelws/wsdl/ReservasiHotel"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:ns2="http://localhost:8080/hotelws/types/ReservasiHotel" name="ReservasiHotel" targetNamespace="http://localhost:8080/hotelws/wsdl/ReservasiHotel">
  - <types>
    - <schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://localhost:8080/hotelws/types/ReservasiHotel"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soap-enc="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://localhost:8080/hotelws/types/ReservasiHotel">
        <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      + <complexType name="ObjStsRuang">
      + <complexType name="ArrayOfObjKonfStsRuang">
      + <complexType name="ObjKonfStsRuang">
      + <complexType name="ObjReservasi">
      + <complexType name="ObjCustomer">
      + <complexType name="ObjKonfReservasi">
      + <complexType name="ObjBayar">
      + <complexType name="ObjKonfBayar">
      + <complexType name="ObjProfilHtl">
      + <complexType name="ArrayOfObjRuangHtl">
      + <complexType name="ObjRuangHtl">
      </schema>
    </types>
  + <message name="ReservasiIF_Info">
  + <message name="ReservasiIF_InfoResponse">
  + <message name="ReservasiIF_Reservasi">
  + <message name="ReservasiIF_ReservasiResponse">
  + <message name="ReservasiIF_UbahReservasi">
  + <message name="ReservasiIF_UbahReservasiResponse">
  + <message name="ReservasiIF_BatalReservasi">
  + <message name="ReservasiIF_BatalReservasiResponse">
  + <message name="ReservasiIF_KonfirmasiByr">
  + <message name="ReservasiIF_KonfirmasiByrResponse">
    <message name="ReservasiIF_ProfilHotel" />
  + <message name="ReservasiIF_ProfilHotelResponse">
    <message name="ReservasiIF_RuangHotel" />
  + <message name="ReservasiIF_RuangHotelResponse">
  + <portType name="ReservasiIF">
  + <binding name="ReservasiIFBinding" type="tns:ReservasiIF">
  + <service name="ReservasiHotel">
  </definitions>
```

Gambar 2. Web Services Definition Language (WSDL)

2.7. JAVA API FOR XML MESSAGING (JAXM)

Untuk pengiriman dokumen XML melalui *internet*, J2EE menyediakan *library* JAXM. *Library* ini berdasarkan pada SOAP 1.1 dan SOAP *attachment*. Dan menyediakan sarana untuk *packaging*, *routing*, dan transport *message* [2].

2.8. JAVA API FOR XML REGISTRIES (JAXR)

Sedangkan untuk pengaksesan registri bisnis melalui internet, disediakan *library* JAXR . Fasilitas ini memudahkan *enterprise* untuk mem*publish* informasi bisnisnya pada tempat registri publik sehingga banyak pihak dapat melakukan *searching* dan mengakses *service-service* yang disediakan [2].

2.9. JAVA API FOR XML-BASED RPC (JAX-RPC)

Selain JAXM dan JAXR, J2EE menyediakan library JAX-RPC yang dapat digunakan oleh aplikasi untuk membuat RPC melalui SOAP. Selain itu JAX-RPC juga dapat digunakan untuk mengirim *message request* dan *respon*. JAX-RPC juga dapat digunakan untuk mendefinisikan skema XMLnya sendiri dan menggunakan skema itu untuk mengirim dokumen XML dan *fragment-fragment* XML. Dengan fasilitas-fasilitas itu, JAX-RPC menjadikan implementasi *web service* lebih mudah dilakukan [2].

3. IMPLEMENTASI SISTEM

Pada tahap ini dilakukan implementasi sistem hotel sebagai web service provider dan sistem broker sebagai web services requester. Arsitektur sistem bisa dilihat pada gambar 3.

Sistem hotel ini mendefinisikan service yang menyediakan fungsi-fungsi yang bisa dipanggil oleh sistem lain. Fungsi-fungsi yang disediakan oleh service ini antara lain :

- Fungsi untuk menerima permintaan informasi serta memberikan informasi status kamar kosong.
- Fungsi untuk melakukan proses reservasi.
- Fungsi untuk melakukan proses konfirmasi pembayaran.
- Fungsi untuk melakukan proses perubahan reservasi.
- Fungsi untuk melakukan proses pembatalan reservasi.
- Fungsi untuk memberikan informasi profil hotel.

Implementasi selanjutnya adalah aplikasi yang menangani proses bisnis dari masing-masing fungsi yang didefinisikan pada web service. Aplikasi ini juga melakukan koneksi dengan database untuk mengakses data yang dibutuhkan. Tahap selanjutnya adalah implementasi database. Pada tahap ini dilakukan generate script DDL yang dihasilkan pada tahap perancangan. Script DDL digenerate pada RDBMS Oracle 8i. Implementasi basis data ini dilakukan baik untuk sistem broker maupun untuk sistem hotel.

Tahap terakhir adalah implementasi aplikasi broker sebagai pengguna layanan web service aplikasi hotel. Pada aplikasi ini dibuat antar muka sebagai penghubung antara user dengan beberapa sistem hotel melalui pemanggilan fungsi pada web service yang disediakan oleh aplikasi hotel. Pembuatan antarmuka berdasarkan kebutuhan untuk masing-masing user. User pada sistem broker terdiri dari admin broker dan customer. Masing-masing user mendapatkan fasilitas untuk mengakses halaman tertentu sesuai dengan haknya masing-masing. Untuk user customer, fasilitas yang didapatkan antara lain: List Reservasi yang dimiliki,pemesanan, perubahan , pembatalan dan pembayaran reservasi. User Admin Broker mendapatkan fasilitas untuk melakukan view customer, view reservasi, view data pembayaran, serta dapat melakukan manajeman data-data reservasi dan user yang termasuk dalam sistem broker. Activity diagram yang terjadi dalam sistem adalah seperti pada gambar 4.

Penjelasan activitynya: customer mencari data kamar/ruang kosong dengan kriteria tertentu. Kemudian system broker memanggil web service yang disediakan hotel dan memanggil fungsi yang seuai dengan transaksi customer. Misalkan customer membutuhkan data tentang kamar/ruang kosong, maka system broker memanggil fungsi ReservasiIF_Info. Informasi tentang fungs-fungsi yang bisa dipanggil bisa dilihat di WSDL yang telah didefinisikan sebelumnya. Web service yang berada di sistem hotel ,emerima request ini dan menjalankan fungsi yang dipanggil. Kemudian data yang dihasilkan oleh eksekusi fungsi dikirimkan kembali ke system broker. Selanjutnya sistem broker menerima dan menampilkan data yang diterima dari web service, dan customer bisa view data dari system broker. Untuk class diagram bisa dilihat pada gambar 5.

Pada class diagram ini hanya didefinisikan class-class yang melayani service dan class-class yang memanggil service. Sedangkan class-class yang menangani proses bisnis dan koneksi database, tidak dijelaskan disini. Class ReservasiIF adalah class interface pada System Hotel yang mendefinisikan fungsi-fungsi yang disediakan pada service tersebut. Sedangkan class ReservasiImpl adalah class yang mengimplementasikan fungsi-fungsi pada class Reservasi IF. Kedua class ini terletak di sistem Hotel. Sedangkan class Konf_Reservasi, reservasi, Ubah_Reservasi dan Batal_reservasi adalah class-class pada

System Broker yang berfungsi untuk menangani proses pemanggilan fungsi pada web service Hotel dan selanjutnya menangani data hasil pemanggilan fungsi pada web service tersebut.
Class Konf_Reservasi adalah class yang memanggil fungsi informasi status ruang/kamar kosong. Class Reservasi adalah class yang menangani pemanggilan fungsi Reservasi untuk transaksi reservasi. Class Ubah_Reservasi adalah class yang menangani fungsi untuk transaksi perubahan reservasi. Class Batal_Reservasi adalah class yang menangani fungsi untuk transaksi pembatalan reservasi.

4. UJI COBA DAN ANALISA HASIL

Uji coba dilakukan pada jaringan intranet dengan menggunakan 3 komputer sebagai server aplikasi hotel, 1 komputer sebagai server broker dan 1 komputer sebagai server database. Masing-masing komputer server aplikasi hotel dan server aplikasi broker memiliki spesifikasi hardware : prosesor Pentium III 800 MHz dengan memori fisik sebesar 384 Mb**.** Sistem operasi yang ter-*install* adalah Windows 2000. Sedangkan komputer yang digunakan sebagai *server* database memiliki spesifikasi hardware : prosesor AMD Athlon 800 MHz dengan memori fisik sebesar 256 Mb**.** Sistem operasi yang ter-*install* pada komputer ini adalah Linux Mandrake 7.2.

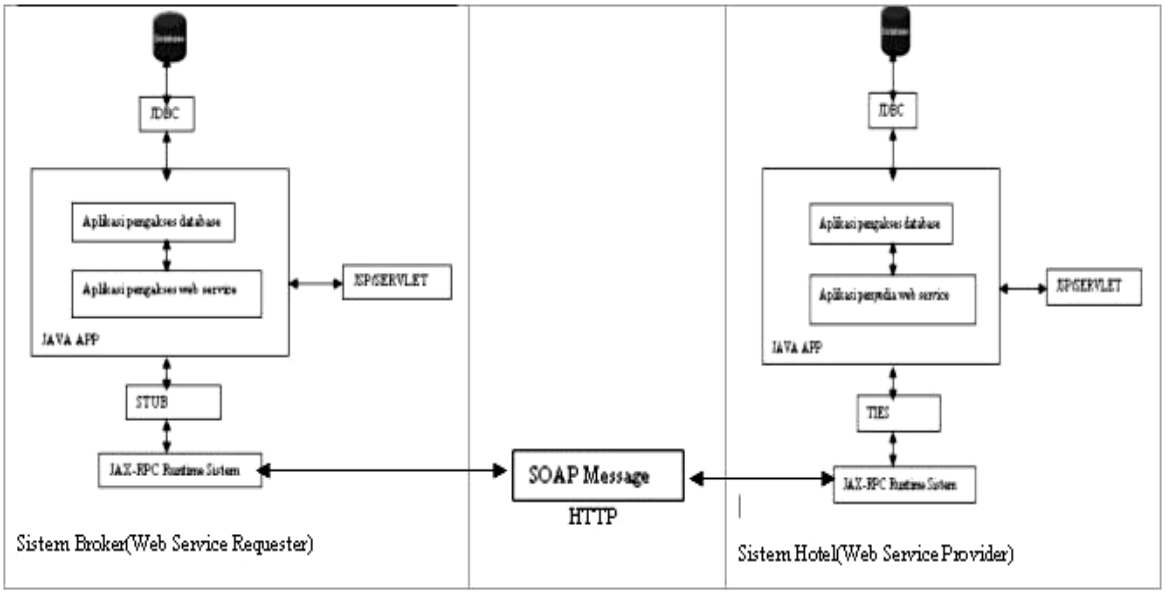
Untuk melaksanakan ujicoba fungsional dibuat beberapa skenario yang berfungsi menguji fungsi-fungsi proses bisnis yang dibuat. Skenario tersebut antara lain:

- Customer melalui aplikasi broker melakukan proses pencarian hotel yang bisa dipesan sesuai dengan kriteria yang dibutuhkan. Pada proses ini aplikasi broker menghubungi web service pada beberapa aplikasi hotel sesuai dengan kebutuhan customer. Data yang akan dikirimkan oleh web service hotel harus merupakan data terbaru tentang kamar yang masih bisa dipesan.
- Dari data yang diperoleh dari beberapa web service, customer memutuskan untuk melakukan transaksi pemesanan hotel yang sesuai. Transaksi ini harus bisa mengurangi jumlah kamar yang bisa dipesan secara otomatis .
- Jika customer benar-benar akan memanfaatkan layanan jasa hotel yang sudah dipesan, maka customer harus melakukan pembayaran secara manual melalui ATM kepada rekening hotel yang dituju. Kemudian customer melalui aplikasi broker melakukan proses konfirmasi pembayaran yang telah dilakukan dengan mengirimkan nomor bukti transfer.
- Jika customer ingin merubah transaksi reservasi, customer melalui aplikasi broker memanggil fungsi perubahan transaksi pemesanan pada web

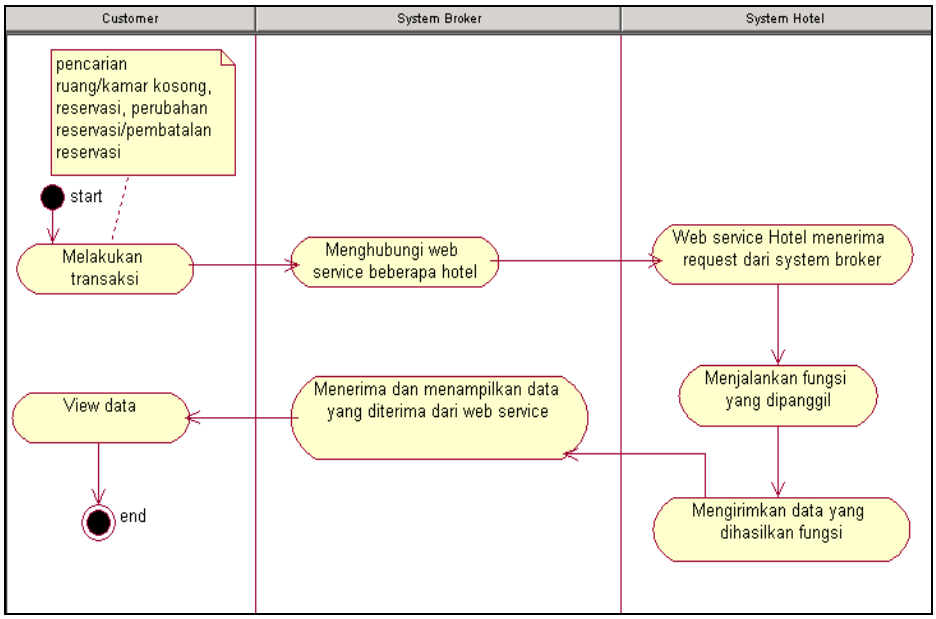
service hotel yang bersangkutan. Transaksi ini harus bisa mengubah data jumlah kamar yang bisa dipesan secara otomatis .

- Jika customer ingin membatalkan transaksi reservasi, customer melalui aplikasi broker memanggil fungsi pembatalan transaksi pemesanan pada web service hotel yang bersangkutan. Transaksi ini harus bisa menambah jumlah kamar yang bisa dipesan secara otomatis .

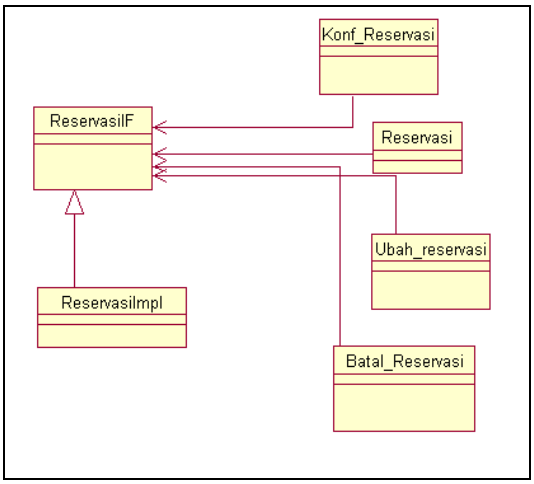
Dari setiap skenario, dilakukan pengecekan pada sisi *web service provider* (sistem hotel) dan *web service requester* (*sistem broker*). Pengecekan yang dilakukan antara lain:



Gambar 3. Arsitektur Sistem



Gambar 4. Activity Diagram



Gambar 5. Class Diagram Object Web Service

- Memastikan bahwa komunikasi antara sistem *broker* (sebagai *web service requester*) dengan sistem hotel (sebagai *web service provider*) sudah berjalan dengan baik, dilakukan pemanggilan *remote method* dengan parameter yang sesuai, dan dicek apakah data respon yang diterima sudah sesuai dengan data yang dikirimkan oleh sistem hotel (sebagai *web service provider*).
- Apakah hasil *query* dan perubahan basis data di sistem hotel (*web service client*) sudah sesuai dengan data yang dikirimkan oleh sistem *broker* melalui *web service*

Pada saat ujicoba, sistem broker berhasil memanggil method/fungsi-fungsi dari beberapa sistem hotel sekaligus. Data yang diterima oleh fungsi-fungsi pada aplikasi hotel juga sesuai dengan data yang dikirimkan oleh aplikasi broker. Kesesuaian data ini disimpulkan berdasarkan perubahan data transaksi reservasi di sistem hotel yang sesuai dengan data yang dikirim oleh sistem broker. Sedangkan data yang diterima kembali oleh sistem broker juga sama dengan data terbaru yang dikirim oleh sistem hotel secara realtime.

Dari hasil ujicoba untuk masing-masing skenario, dapat disimpulkan bahwa komunikasi antara sistem broker dengan beberapa sistem hotel yang terdistribusi berjalan dengan baik. Dan proses bisnis reservasi hotel juga berhasil ditangani secara baik oleh sistem hotel dan sistem broker. Hal ini berdasarkan data yang dihasilkan telah sesuai dengan skenario yang telah ditentukan sebelumnya.

5. KESIMPULAN

Adapun kesimpulan yang dapat diambil adalah sebagai berikut:

1. Penerapan aplikasi web service memudahkan komunikasi antara beberapa sistem yang terdistribusi.
2. Dengan adanya penerapan web service pada sistem terdistribusi, suatu sistem dapat berinteraksi dengan beberapa sistem lain secara bersamaan untuk mendapatkan data terbaru secara *realtime* .
3. Dengan penerapan webservice, transaksi antar sistem yang terdistribusi dapat dilakukan secara otomatis,tanpa campur tangan administrator.
4. Penggunaan teknologi J2EE dalam aplikasi web service lebih fleksibel karena tidak tergantung pada platform sistim operasi tertentu.

6. DAFTAR PUSTAKA

1. Craig Larson, “Applying UML and Patterns, An Introduction to Object Oriented Analysis and Design”, 1999.
2. Maydene Fisher, “Introduction to Web Services”, http://java.sun.com, 2002.
3. Ida Bagus Komang Yoga Adhyaksa, “Perancangan dan Pembuatan Perangkat Lunak Sistem Front Office Hotel Online Pada Jaringan Intranet”, Penelitian Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember, 2000.
4. James Kao, “Developer’s Guide to Building XML-Based Web Services”, The Middleware Company, 2001.
5. Navaro Ann, Chuck White, Linda Burman, “Mastering XML”, SYBEX Inc., 2000.
6. Roman Ed, “Mastering Enterprise JavaBeans and the Java™ 2 Platform”, Enterprise Edition, John Wiley & Sons, Inc., 1999.
7. Shang Shin, “Sun™ ONE & Web Services”, Sun Microsystem.
8. Sun Microsystems, “The Java ™ Web Services Tutorial”, http://java.sun.com/ webservices/
9. 9W3C, “Extensible Markup Language(XML) 1.0”, (Second Edition) Specification, http://www .w3c.org/TR/2000/REC-xml-20001006