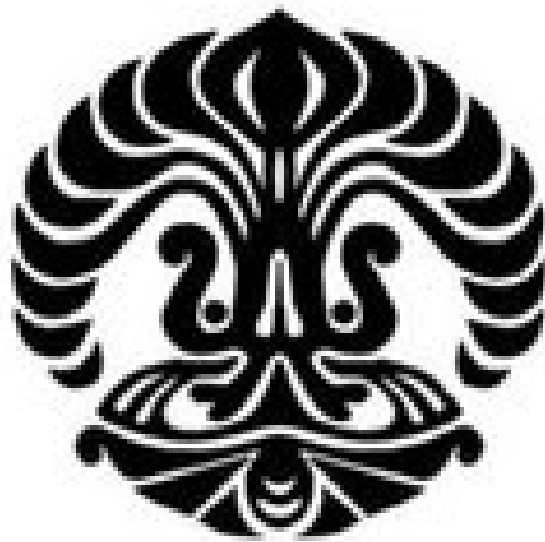


TUGAS MATA KULIAH PENGANTAR SISTEM OPERASI DAN SISTEM KOMPUTER

BAB 13:

INTERNET DAN LAYANAN APLIKASI TERDISTRIBUSI



Oleh:

**Herry Z. Anshary (7205000881)
Safitri Juanita (7205001012)**

**Magister Teknologi Informasi
Fakultas Ilmu Komputer
Universitas Indonesia
2005**

DAFTAR ISI
Internet dan Distribusi Layanan Aplikasi

	Hal
Apa yang dimaksud dengan Internet?	4
Sejarah Internet	4
Internet pada saat ini	4
Standar Web Protokol dan service	11
Apa yang dimaksud dengan Komputasi Terdistribusi?	13
Apa perbedaan antara Distributed Computing dan Cluster Computing	16
Lalu bagaimana perbedaan antara Distributed Computing versus Grid Computing?	18
Client-Server Architecture	19
Peer-to-peer Architecture	20
N-tier Client-Server Architecture	21
Karakteristik Aplikasi Komputasi Terdistribusi	22
Dalam penyediaan akses yang terdistribusi, bukankah system operasi harus bisa melayani pengguna dalam mengakses sumber-sumber tersebut baik dari dalam maupun dari luar. Bagaimana membedakan bahwa itu adalah sumber luar maupun dalam, serta bagaimana mengakses sumber yang diluar?	23
Protocol Stacks	24
Sedangkan apa yang dimaksud dengan Assessing Remote	25

Resources?	
Sebelum menjelaskan tentang Interprocess Communication yang pernah disinggung diatas, bagaimana bentuk-bentuk aplikasi komputasi terdistribusi yang sedang berkembang saat ini?	26
Lalu, apa yang dimaksud Interprocess Communications?	27
Pipes	27
Fifos	28
Shared Memory	28
Mapped Memory	29
Message Queues	30
Sockets	30
Remote Procedure Calls	30
Apa yang dimaksud dengan SOAP, CORBA, COM, DCOM, COM+, ActiveX tersebut?	32
SOAP	32
CORBA	36
COM	40
ActiveX	41
DCOM	41
COM+	42
Ketika beberapa sumber terdistribusi antar jaringan, bahwa sumber yang satu (pengguna) dengan penyedia (provider) harus saling bertemu. Mekanisme apa yang diperlukan?	42
Lightweight Directory Access Protokol (LDAP)	43
Bagaimana Cara Kerja X.500 tersebut, lalu apa perbedaan	44

dengan LDAP?

Membicarakan Directory Service, MS Active Directory (MSAD) 46

merupakan arsitektur yang tidak bisa diabaikan, bisa tolong
dijelaskan?

Referensi 48

Apa yang dimaksud dengan Internet?

Pengertian Internet Secara harafiah, internet (kependekan daripada perkataan 'inter-network') ialah rangkaian komputer yang berhubung menerusi beberapa rangkaian. Manakala Internet (huruf 'I' besar) ialah sistem komputer umum, yang berhubung secara global dan menggunakan TCP/IP sebagai protokol pertukaran paket (*packet switching communication protocol*). Rangkaian internet yang terbesar dinamakan **Internet**. Cara menghubungkan rangkaian dengan kaedah ini dinamakan internetworking.

Sejarah Internet

Rangkaian pusat yang membentuk Internet diawali pada tahun 1969 sebagai ARPANET, yang dibangun oleh ARPA (United States Department of Defense Advanced Research Projects Agency). Beberapa penyelidikan awal yang disumbang oleh ARPANET termasuk kaedah rangkaian tanpa-pusat (decentralised network), teori queueing, dan kaedah pertukaran paket (packet switching).

Pada 01 Januari 1983, ARPANET menukar protokol rangkaian pusatnya, dari NCP ke TCP/IP. Ini merupakan awal dari Internet yang kita kenal hari ini.

Pada sekitar 1990-an, Internet telah berkembang dan menyambungkan kebanyakan pengguna jaringan-jaringan komputer yang ada.

Internet pada saat ini

Internet dijaga oleh perjanjian bi- atau multilateral dan spesifikasi teknikal (protokol yang menerangkan tentang perpindahan data antara rangkaian).

Protokol-protokol ini dibentuk berdasarkan perbincangan Pasukan Juruter Internet (Internet Engineering Task Force), yang terbuka kepada umum. Badan ini mengeluarkan dokumen yang dikenali sebagai **RFC** (Request for Comments). Sebagian dari RFC dijadikan Standar Internet (Internet Standard), oleh Badan Arsitektur Internet (Internet Architecture Board - IAB). Protokol-protokol internet yang sering digunakan adalah seperti:

1. **IP** atau *Internet Protocol* adalah protokol di internet yang mengurus masalah pengalamatan dan mengatur pengiriman paket data sehingga ia sampai ke alamat yang benar.
2. **TCP** (Transmission Control Protocol) adalah suatu protokol yang berorientasi sambungan dan reliable, yang didokumentasikan di RFC 793.
3. Protokol Datagram Pengguna (*User Datagram Protocol*) (**UDP**) adalah protokol lapisan pengangkutan berorientasi pesan minimal yang didokumenkan dalam IETF Permohonan untuk komen (*Request for Comments -RFC*) 768
4. **DNS** (Domain Name System, bahasa Indonesia: Sistem Penamaan Domain) adalah sebuah sistem yang menyimpan informasi tentang nama host maupun nama domain dalam bentuk basis data tersebar (*distributed database*) di dalam jaringan komputer, misalkan: Internet. DNS menyediakan alamat IP untuk setiap nama host dan mendata setiap server transmisi surat (mail exchange server) yang menerima surat elektronik (*email*) untuk setiap domain.

5. **Point-to-Point Protocol (PPP)** , sebuah protokol untuk transfer data komputer

6. **SLIP** (Serial Line Internet Protocol) Protokol dengan menggunakan sambungan serial MAC Ethernet, FDDI, ISDN, ATM,

7. **Internet Control Message Protocol (ICMP)** adalah salah satu protokol inti dari keluarga protokol internet. ICMP utamanya digunakan oleh sistem operasi komputer jaringan untuk mengirim pesan kesalahan yang menyatakan, sebagai contoh, bahwa komputer tujuan tidak bisa dijangkau.

ICMP berbeda tujuan dengan TCP dan UDP dalam hal ICMP tidak digunakan secara langsung oleh aplikasi jaringan milik pengguna. salah satu pengecualian adalah aplikasi ping yang mengirim pesan ICMP *Echo Request* (dan menerima *Echo Reply*) untuk menentukan apakah komputer tujuan dapat dijangkau dan berapa lama paket yang dikirimkan dibalas oleh komputer tujuan.

8. **POP3** (*Post Office Protocol version 3*) adalah protokol yang digunakan untuk mengambil surat elektronik (email) dari server email. Protokol ini sangat erat hubungannya dengan protokol SMTP dimana protokol SMTP berguna untuk mengirim surat elektronik dari komputer pengirim ke komputer server dari penerima. Kemudian penerima mengambil surat elektronik yang dikirim dari server dengan menggunakan protokol ini.

Protokol ini dibuat karena desain dari sistem surat elektronik yang mengharuskan adanya server surat elektronik yang menampung surat elektronik untuk sementara sampai surat elektronik tersebut diambil oleh

penerima yang berhak. Kehadiran server surat elektronik ini disebabkan kenyataan hanya sebagian kecil dari komputer penerima surat elektronik yang terus-menerus melakukan koneksi ke jaringan internet.

9. **IMAP** atau Internet Message Access Protocol adalah protokol standar untuk mengakses atau mengambil e-mail dari server. IMAP memungkinkan pengguna memilih pesan e-mail yang akan diambil, membuat folder di server, mencari pesan e-mail tertentu, bahkan menghapus pesan e-mail yang ada. Kemampuan ini jauh lebih baik daripada POP (Post Office Protocol) yang hanya memperbolehkan kita mengambil/download semua pesan yang ada tanpa kecuali.

10. **SMTP** (*Simple Mail Transfer Protocol*) merupakan salah satu protokol yang umum digunakan untuk pengiriman surat elektronik di Internet. Protokol ini dipergunakan untuk mengirimkan data dari komputer pengirim surat elektronik ke server surat elektronik penerima.

11. **HTTP** (HyperText Transfer Protocol) adalah protokol yang dipergunakan untuk mentransfer dokumen dalam World Wide Web (WWW). Protokol ini adalah protokol ringan, tidak berstatus dan generik yang dapat dipergunakan berbagai macam tipe dokumen.

Pengembangan HTTP dikoordinasi oleh Konsorsium World Wide Web (W3C) dan grup bekerja Internet Engineering Task Force (IETF), bekerja dalam publikasi satu seri RFC, yang paling terkenal RFC 2616, yang menjelaskan HTTP/1,1, versi HTTP yang digunakan umum sekarang ini.

HTTP adalah sebuah protokol meminta/menjawab antara client dan server. Sebuah client HTTP seperti web browser, biasanya memulai permintaan dengan membuat hubungan TCP/IP ke port tertentu di tuan rumah yang jauh (biasanya port 80). Sebuah server HTTP yang mendengarkan di port tersebut menunggu client mengirim kode permintaan (request), seperti "GET / HTTP/1.1" (yang akan meminta halaman yang sudah ditentukan), diikuti dengan pesan MIME yang memiliki beberapa informasi kode kepala yang menjelaskan aspek dari permintaan tersebut, diikuti dengan badan dari data tertentu. Beberapa kepala (header) juga bebas ditulis atau tidak, sementara lainnya (seperti tuan rumah) diperlukan oleh protokol HTTP/1,1. Begitu menerima kode permintaan (dan pesan, bila ada), server mengirim kembali kode jawaban, seperti "200 OK", dan sebuah pesan yang diminta, atau sebuah pesan error atau pesan lainnya.

12. **HTTPS** adalah versi aman dari HTTP, protokol komunikasi dari World Wide Web. Ditemukan oleh Netscape Communications Corporation untuk menyediakan autentikasi dan komunikasi tersandi dan penggunaan dalam komersi elektrik.

Selain menggunakan komunikasi plain text, HTTPS menyandikan data sesi menggunakan protokol SSL (Secure Socket layer) atau protokol TLS (Transport Layer Security). Kedua protokol tersebut memberikan perlindungan yang memadai dari serangan eavesdroppers, dan man in the middle attacks. Pada umumnya port HTTPS adalah 443.

Tingkat keamanan tergantung pada ketepatan dalam mengimplementasikan pada browser web dan perangkat lunak server dan didukung oleh algoritma penyandian yang aktual.

Oleh karena itu, pada halaman web digunakan HTTPS, dan URL yang digunakan dimulai dengan 'https://' bukan dengan 'http://'

Kesalahpahaman yang sering terjadi pada pengguna kartu kredit di web ialah dengan menganggap HTTPS “sepenuhnya” melindungi transaksi mereka. Sedangkan pada kenyataannya, HTTPS hanya melakukan enkripsi informasi dari kartu mereka antara browser mereka dengan web server yang menerima informasi. Pada web server, informasi kartu mereka secara tipikal tersimpan di database server (terkadang tidak langsung dikirimkan ke pemroses kartu kredit), dan server database inilah yang paling sering menjadi sasaran penyerangan oleh pihak-pihak yang tidak berkepentingan

13. **SSH** adalah paket program yang digunakan sebagai pengganti yang aman untuk rlogin, rsh dan rcp. Ia menggunakan public-key cryptography untuk mengenkripsi komunikasi antara dua host, demikian pula untuk autentikasi pemakai. Ia dapat digunakan untuk login secara aman ke remote host atau menyalin data antar host, sementara mencegah *man-in-the-middle attacks* (pembajakan sesi) dan *DNS spoofing*. Ia akan melakukan kompresi data pada koneksi anda, dan komunikasi X11 yang aman antar hos

14. Remote Login (Telnet)

Telnet memungkinkan pengguna komputer dapat melakukan login ke dalam suatu komputer di dalam jaringan. Ketika kita melakukan telnet, secara tidak langsung kita telah menjadi pengguna yang sah dari komputer tersebut.

15. **FTP** (singkatan dari *File Transfer Protocol*) adalah protokol Internet yang merupakan standar untuk pentransferan berkas (*file*) komputer antara mesin-mesin yang menjalankan sistem yang sangat berbeda.

Karena berfungsi sebagai file sharing maka kita dapat *download* dan *upload* file yang kita inginkan.

Seperti halnya *browsing*, FTP juga memiliki alamat. Alamat yang digunakan untuk browsing dapat diawali dengan HTTP misalnya <http://id.wikipedia.org> sedangkan FTP diawali dengan ftp misalnya <ftp://globalscape.com>

16. **LDAP** adalah merupakan media penyimpanan terpusat dan akses informasi jarak jauh (LDAP akan dibahas secara detail pada bab selanjutnya).

17. **SSL** (Secure Sockets Layer) adalah metode enkripsi yang dikembangkan oleh Netscape untuk memberikan keamanan di Internet. Ia mendukung beberapa protokol enkripsi dan memberikan autentikasi client dan server. SSL beroperasi pada layer transpor, menciptakan saluran enkripsi yang aman untuk data, dan dapat mengenkripsi banyak tipe data. Hal ini dapat dilihat ketika mengunjungi site yang aman untuk melihat dokumen online aman dengan Communicator, dan berfungsi sebagai dasar komunikasi yang aman dengan Communicator, juga dengan enkripsi data Netscape Communication lainnya

Beberapa layanan populer di internet yang menggunakan protokol di atas, ialah email/surat_elektronik, Usenet, Newsgroup, perkongsian file (File Sharing), WWW (World Wide Web), Gopher, akses sesi (Session Access), WAIS, finger, IRC, MUD, dan MUSH. Di antara semua ini, email/surat_elektronik dan World Wide Web lebih kerap digunakan, dan lebih banyak servis yang dibangun berdasarkannya, seperti milis (Mailing List) dan Weblog. Internet memungkinkan adanya servis terkini (Real-time service), seperti web radio, dan webcast, yang dapat diakses di seluruh dunia.

Standar Web Protokol dan service

Semua sumber daya web di identifikasi dengan Uniform Resource Locator (URL) seperti [Http://averia.mgt.unm.edu/default.htm](http://averia.mgt.unm.edu/default.htm). URL memiliki 4 komponen seperti :

1. Protokol –spesifikasi bagian atas sumberdaya akses protokol (http:// adalah nilai default)
2. Host – nomor IP atau nama yg terdaftar dari internet host komputer atau perangkat
3. Port
4. Sumber Daya – nama path yang lengkap dari sumber daya dari host

Tabel Web Protokol

Kategori	Contoh Protokol
Format dan linked dokumen	Hypertext Markup Language (HTML) dan Extensible Markup Language (XML)
File dan transfer dokumen	File transfer protocol (FTP) dan Hypertext Transfer Protocol (HTTP)
Remote login dan eksekusi proses	Telnet, tn3270, dan Remote Procedure Call (RPC)
Mail dan pesan	Simple Mail Transfer Protocol (SMTP), post office protocol (POP), dan internet message acces protokol (IMAP)
Eksekusi program	Java, javaScript, dan Visual basic Script (VBScript)

Apa yang dimaksud dengan Komputasi Terdistribusi?

Dalam memahami komputasi terdistribusi (distributed computing) dapat dilakukan dengan berbagai macam cara. Salah satu defenisi sederhana dari komputasi terdistribusi adalah proses berjalannya sebuah aktifitas komputasi yang dilakukan oleh lebih dari satu komputer yang berbeda. Pemahaman lain tentang komputasi terdistribusi adalah sebuah distribusi sebagian dari sebuah sistem informasi melalui banyak sistem komputer di banyak lokasi (Burd: 2003)

Sekarang ini banyak sekali produk-produk yang pasarkan juga telah banyak proyek yang mengembangkan apa yang disebut Komputasi Terdistribusi ini, yaitu pengembangan sebuah arsitektur system yang menciptakan adanya distribusi proses data antar jaringan system yang tersambung. Sebagai contoh proyek adalah proyek Seti@Home, sebuah proyek yang berupaya mengungkap adanya kehidupan yang cerdas (ET) di luar angkasa. Proyek lain yang dikembangkan saat ini dan dikembangkan dengan metode komputasi terdistribusi (Brain H.Rudal, 2005), diantaranya;

- a. Einstein@home, sebuah proyek yang mengungkap tentang teori Einstein tentang grafitasi serta hokum relativitas yang sampai saat ini belum bisa dibuktikan;
- b. Brain Fingerprinting, sebuah proyek amerika yang akan membuat system yang dapat mengungkap informasi yang ada pada otak manusia;
- c. Climate model forecast future, ini merupakan proyek yang akan memproyeksikan kondisi cuaca dunia akibat efek rumah kaca. Dengan

system ini kondisi cuaca bumi hingga setengah abad kemudian dapat segera diprediksi.

Komputasi terdistribusi ini banyak diminati disebabkan metode ini lebih efisien, yaitu memanfaatkan kondisi idle/diamnya sebuah proses kerja CPU, media penyimpanan yang terdapat pada ratusan maupun ribuan sistem komputer yang terkoneksi yang bekerja bersama-sama menyelesaikan satu pekerjaan. Walaupun masih banyak kelemahan, seperti terbatasnya aplikasi untuk kompilasi, maupun 'bottleneck' pada jalur bandwidth, ditambah lagi masalah keamanan serta standarisasi juga merupakan masalah lain yang tidak dapat diabaikan.

Beberapa perusahaan telah bertahun-tahun membuat dan memasarkan sistem komputasi terdistribusi, dan juga telah mengembangkan berbagai macam inisiatif dan arsitektur sehingga memungkinkan adanya pendistribusian proses data dan obyek lintas jaringan system yang terkoneksi. Hal yang menarik dari sebuah komputasi terdistribusi telah ditemukan dan dirasakan pada perkembangannya yang terakhir. Dan ini akan menjadi fokus pembahasan di mana terdapat mengkondisikan 'idle'nya sebuah CPU, media penyimpanan dalam ratusan bahkan ribuan system yang terkoneksi untuk bersama-sama menyelesaikan satu permasalahan tertentu.

Perkembangan model pemrosesan tertentu sangat lah terbatas, sehingga kompilasi aplikasi, kondisi 'bottlenecks' pada bandwidth, ditambah lagi tingkat keamanan, standarisasi menjadi sebuah tantangan sendiri. Pada tahun terakhir beberapa perusahaan seperti Napster, Intel, Microsoft, Sun dan Compaq akhirnya dapat berupaya menyelesaikan permasalahan di atas, yaitu

bagaimana menjadikan komputasi terdistribusi ini menjadi sebuah konsep yang cerdas dalam perkembangan system informasi modern.

Dalam perkembangannya, untuk memahami konsep komputasi terdistribusi dapat dilakukan perbandingan dengan berbagai macam konsep lain yang masing-masing memiliki karakteristik yang berbeda, diantaranya adalah cluster computing dan grid computing.

Apa perbedaan antara Distributed Computing dan Cluster Computing

Berbeda dengan komputerisasi dengan mekanisme kluster dimana kondisi komputerisasi terdistribusi secara khusus tidak lagi sama dengan menjalankan sekumpulan aktifitas, dimana komputer berkluster biasanya lebih berjalan sangat bersamaan. Perbedaan ini menjadikan komputerisasi terdistribusi lebih menarik sebab, jika di konfigurasi secara tepat, hal tersebut dapat memanfaatkan sumber-sumber komputasi yang diinginkan atau tidak sama sekali. Selain itu juga dapat menciptakan sumber-sumber komputerisasi atau tidak mungkin sama sekali. Sebagai contoh proyek Seti@home menggunakan 'idle time' pada ratusan komputer seluruh dunia, dan ini dapat digunakan untuk menganalisa penerimaan sinyal yang sebelumnya tidak mungkin dapat dilakukan. Beberapa kondisi memungkinkan untuk mengelola data dimana memerlukan kekuatan superkomputer yang sangat mahal harganya.

Selain itu Komputerisasi yang terdistribusi sangat menarik sebab operasi yang interaktif yang membiarkan lebih banyak komputer dalam kondisi idle yang lebih banyak. Proses ini dimana menjalankan aspek yang terdistribusi

(misalnya pada saat menjalankan mesin juga melaksanakan pekerjaan lain) ini biasanya didesain untuk memprioritaskan pekerjaan ringan, penggunaan juga memperhitungkan tenaga yang terbuang percuma.

Akan tetapi, memiliki proses prioritas rendah secara konsisten menghalangi sistem operasi memberdayakan pengelolaan rutin dengan menempatkan prosesor pada kondisi power rendah, ini mengakibatkan meningkatnya konsumsi listrik. Pada beberapa CPU (terutama yang terbaru dan memiliki kecepatan tinggi), perbedaannya dapat mencapai 10 watt.

Komputerisasi yang terdistribusi acapkali terlibat pada persaingan dengan sistem distribusi yang lain. Kompetisi ini mungkin saja sangat bergengsi, atau juga berarti untuk menarik perhatian pengguna untuk dapat ikut menyumbang 'processing power' pada proyek tertentu. Sebagai contoh, terdapat istilah "stat race" yaitu penilaian terhadap proyek yang dikelola untuk menghasilkan berbagai pekerjaan yang terdistribusi selama sehari atau seminggu yang telah lewat. Hal ini terbukti sangat penting karena secara nyata bahwa proyek komputerisasi yang terdistribusi tersebut dapat menghadirkan sebuah analisa kinerja mereka, dan diperbaharui paling tidak setiap hari, walau jika tidak dituntut harus real-time.

Komputerisasi yang terdistribusi juga merupakan cakupan penelitian yang aktif dan kaya akan literatur. Konferensi tentang komputersisasi terdistribusi yang terkenal adalah The Internasional Conference on Dependable Systems and Networks (Konferensi internasional pada sistem dan jaringan handal) dan the ACM Symposium on principles of distributed computing (ACM - Symposium pada prinsip-prinsip komputerisasi terdistribusi). Jurnal-jurnal termasuk the journal of parallel and distributed computing (jurnal tentang

komputerisasi yang paralel dan terdistribusi) IEEE transactions on parallel and distributed systems (IEEE- transaksi pada sistem yang paralel dan terdistribusi) dan lain-lain.

Lalu bagaimana perbedaan antara Distributed Computing versus Grid Computing?

Pada dasarnya kedua trend ini komputasi terdistribusi dan komputasi terkoneksi (grid computing) berjalan bersamaan, tergantung bagaimana cara melihat dan memahaminya. Pada kondisi pasar yang ada saat ini, kedua terminology tersebut saling tumpang tindih, dimana komputasi terdistribusi merupakan bagian dari komputasi terkoneksi. Terminology komputasi terkoneksi diambil berdasarkan sebuah scenario ideal dimana mata rantai pekerjaan CPU dan penyimpanan jutaan system antar fungsi jaringan di dunia dapat diakses oleh siapapun yang membutuhkannya secara mudah, fleksible dan siap saji. Hal ini tak ubahnya perusahaan listrik dalam membagikan koneksi listrik kepada pengunanya.

Sun Microsystem mendefisikan komputasi terkoneksi sebagai sebuah infrastruktur perangkat keras dan lunak yang dapat menyediakan akses kemampuan komputasi secara konsisten, canggih dan murah. Komputasi terkoneksi juga dapat meliputi PC desktop, tapi dalam hal ini lebih berkonsentrasi pada workstation yang lebih kuat, atau server, atau mungkin mainframe dan supercomputer yang bekerja seharian guna menyelesaikan satu masalah. Sehingga komputasi terkoneksi ini cenderung dipahami sebagai system yang berdedikasi (dedicated), dan penggunaan utamanya untuk pekerjaan yang berbeda.

Dalam skala yang lebih luas, komputasi terdistribusi dengan berbagai macam bentuknya dapat diartikan sebagai sumber dari sekumpulan ratusan, ribuan PC end-user, dimana masing-masing memiliki kekurangan memory dan kekuatan proses pengolahan, dimana tujuannya bukan semata-mata mendistribusi tugas komputasi, tetapi adalah pelayanan terhadap pengguna(user). Namun kedua jenis konsep ini jangan dilakukan secara massif, perlu dilakukan pembatasan CPU baik pada level group, divisi, departemen dalam sebuah perusahaan.

Dalam implementasinya di lapangan, terdapat jenjang dan bentuk dari arsitektur komputasi terdistribusi, Client-Server Architecture, Peer-to-peer Architecture dan N-Layer Client-Server Architecture.

Client-Server Architecture

Menurut *Gallaugher & Ramanathan (1996)* :“client/server adalah client mengirim permintaan ke server, server menterjemahkan pesan, kemudian berusaha memenuhi permintaan”. Sedangkan menurut *Blaha & Premerlani (1998)* : “client/server adalah suatu arsitektur dimana sumber daya server menyediakan komputasi untuk banyak komponen client. Client dapat mengakses satu server atau multiple server. Client dan server bisa berjalan pada mesin yg sama atau berbeda, ditulis dalam berbagai bahasa dan menggunakan sistem operasi yang berbeda.

Secara umum Client/Server adalah arsitektur jaringan aplikasi yang memisahkan klien dari server (umumnya GUI). Setiap satuan perangkat lunak klien berhubungan dengan perangkat lunak server. Client/server adalah

arsitektur berskala dimana setiap komputer atau proses pada jaringan berperan sebagai klien atau server. Perangkat lunak server pada umumnya walaupun tidak selalu akan menjalankan komputer dengan kekuatan penuh yang secara khusus dan eksklusif menjalankan aplikasi bisnis. Sedangkan Perangkat lunak Client pada umumnya berjalan pada PC/workstation biasa. Client akan mendapatkan seluruh informasinya dan mengirimkannya kepada perangkat lunak server untuk sebuah keperluan, sebagai contoh konfigurasi file, kuota penyimpanan, program aplikasi bisnis atau untuk membebaskan intensitas pekerjaan komputasi dan mengkondisikan komputer Client bebas dan siap menjalankan pekerjaan lainnya.

Client yang paling populer dan digunakan secara luas saat ini adalah web browser dimana berkomunikasi dengan web server melalui internet untuk mendapatkan dan menampilkan isi halaman web.

Bentuk lain dari arsitektur client/server dikenal dengan istilah 'thin client', dimana terdapat sejumlah Client yang sedikit. Thin client menggunakan sedikit sumber dari PC Host yang ada. Tugas dari thin client pada umumnya hanya menampilkan secara grafik tentang informasi umum dari perangkat lunak server. Hal ini memungkinkan perusahaan untuk lebih mudah mengelola bisnis logiknya untuk semua aplikasi-aplikasi pada pusat lokasi.

Peer-to-peer Architecture

Bentuk lain dari arsitektur jaringan dikenal dengan arsitektur peer-to-peer, sebab setiap node atau instance dari sebuah program adalah merupakan klien dan server dan memiliki tugas dan tanggung jawab sendiri. Keduanya

baik client/server maupun arsitektur peer-to-peer banyak digunakan. Masing-masing memiliki keunggulan maupun kelemahan.

N-tier Client-Server Architecture

Aplikasi server selalu menyimpan data pada mesin ke-tiga, dikenal dengan server database. Ini juga disebut arsitektur 3-lapis dimana arsitektur Client/Server yang utama adalah dua-lapis (two-layers). Umumnya arsitektur n-tier atau arsitektur multi-tier dapat menerapkan beberapa kegiatan yang berbeda, termasuk hubungan transitif antara aplikasi server yang menerapkan fungsi berbeda dari bisnis logis, setiap fungsi bisa saja memanfaatkan atau tidak terhadap sistem database berbeda.

Pada proses komputerisasi, three-tier adalah arsitektur client-server dimana user interface, logika proses fungsional (aturan bisnis) dan penyimpanan data serta akses data dikembangkan danelihara sebagai modul yang berdiri sendiri, sering kali dalam platform yang berbeda. Istilah three-tier atau three-layer, sebagaimana konsep arsitektur multi-tier, terkesan dikembangkan oleh aplikasi lunak rasional atau microsoft.

Model three-tier cenderung menjadi arsitektur aplikasi lunak dan inti dari desain aplikasi lunak. Terpisah dari kelebihan umum dari aplikasi lunak modular yang didefinisikan sebagai interface, arsitektur three-tier memiliki kecenderungan bentuk three-tier ini ditingkatkan atau diganti secara mandiri sebagai tuntutan dari perubahan teknologi. Sebagai contoh, peningkatan sistem operasi desktop dari microsoft windows ke unix merupakan dampak dari kode interface pengguna.

Seiring dengan hal tersebut, interface pengguna menjalankan PC desktop atau workstation dan penggunaan GUI, logika proses fungsionalnya terdiri dari satu atau lebih modul yang berbeda yang berjalan diatas server workstation atau aplikasi server, dan sebuah RDMS pada server database atau mainframe memiliki penyimpanan data yang logic. Lapisan antara dapat juga berbentuk memulti lapiskan dirinya sendiri (dalam kasus ini keseluruhan arsitektur dikenal dengan arsitektur n-tier).

Dalam eksplorasi aplikasi lunak, arsitektur multi-tier digunakan untuk menjabarkan sesuatu dimana derajat pemisahan yang dimiliki oleh satu atau lebih agent aplikasi lunak antara komponen diskrit dalam upaya untuk memfasilitasi pemrosesan di beberapa hal. Sebagai contoh hal ini digunakan pada perangkat tengah untuk pelayanan permintaan data yang lebih efisien antara pengguna dengan database. Ini juga dapat disandarkan sebagai arsitektur n-tier. Istilah yang paling banyak digunakan adalah arsitektur three-tier.

Karakteristik Aplikasi Komputasi Terdistribusi

Pada dasarnya tidak semua aplikasi dapat digunakan pada komputasi terdistribusi ini. Jika aplikasi ini semakin real-time, maka semakin tidak cocok untuk digunakan. Meskipun setiap proses pekerjaan dapat diselesaikan dalam satu jam tidak akan memberikan manfaat jika 2 kondisi terjadi, yaitu lintas komunikasi antar system terdistribusi ini serta perubahan proses pelayanan pada klien mengalami 'bottleneck'.

Untuk itu, semua aplikasi dengan tugas tertentu dan memerlukan akses terhadap banyak data maka memerlukan system yang lebih besar lagi dibandingkan dengan sebuah PC. Jika data tersebut sebesar terabytes, sebuah supercomputer akan mungkin melakukan pekerjaan lintas system secara baik tanpa harus merusak atau membebani jaringan yang ada. Server ataupun system kluster yang lainnya akan lebih layak untuk aplikasi data yang lebih sedikit. Sedangkan untuk aplikasi terdistribusi dengan menggunakan banyak PC, kebutuhan data harus sesuai dengan kapasitas dari memory pada PC, dengan tetap menyediakan ruang kosong sebagai cadangan.

Berkenaan dengan hal tersebut, United device merekomendasikan bahwa sebuah aplikasi harus memiliki kemampuan untuk mengexploitasi apa yang disebut 'coarse-grained pallelism' yang maksudnya adalah bahwa harus memiliki kemampuan untuk membagi aplikasi menjadi beberapa pekerjaan yang terpisah dan berdiri sendiri dan dapat dilakukan secara bersamaan. Hal yang paling penting untuk diperhatikan adalah kebutuhan adanya komunikasi antar pekerjaan tersebut, hal ini dikenal dengan interprocess communications.

Dalam penyediaan akses yang terdistribusi, bukankah system operasi harus bisa melayani pengguna dalam mengakses sumber-sumber tersebut baik dari dalam maupun dari luar. Bagaimana membedakan bahwa itu adalah sumber luar maupun dalam, serta bagaimana mengakses sumber yang diluar?

Untuk menjawab masalah tersebut, komponen system operasi memiliki fungsi, diantaranya disebut Protocol Stacks dan Remote Resource Access.

Protocol Stacks

Protocol stack adalah perangkat lunak yang khusus digunakan pada sebuah kumpulan protocol jaringan komputer (a computer protocol suite). Istilah ini kadang-kadang sering tertukar, jelasnya istilah suite adalah defenisinya sedangkan protocol adalah perangkat lunak yang melaksanakannya.

Protocol sendiri di dalam suite sering dianggap memiliki satu tujuan. Persepsi ini akan memudahkan dalam melakukan evaluasi dan desain, sebab setiap modul protocol biasanya akan melakukan komunikasi antar 'dua' dari suatu yang diayangkan sebagai lapisan dari irisan protocols. Protocol terendah dikenal dengan 'low level' yang secara fisik berinteraksi dengan perangkat keras. Semakin tinggi lapisannya semakin banyak fitur yang dimiliki. Pengguna aplikasi acapkali berhubungan dengan lapisan paling atas.

OSI Reference Model memiliki 7 (tujuh) layer yang mendefinisikan fungsi-fungsi dari data communications protocols. Masing-masing layer dari OSI model merepresentasikan fungsi yang akan dilakukan ketika data di transfer antara cooperating applications across dan intervening network.

protocol TCP/IP dapat diilustrasikan seperti pile dari susunan balok-balok bangunan antar satu dan lainnya. Karenanya, struktur ini juga dikenal sebagai stack atau protocol stack.

TCP/IP terdiri dari 4 lapisan (layer), berupa sekumpulan protokol yang bertingkat (protokol stack). Lapisan lapisan tersebut adalah:

- Lapisan Hubungan Antarmuka Jaringan (Network Interface Layer), Bertanggung jawab untuk mengirim dan menerima data dari media fisik.
- Lapisan_Internet, Bertanggung jawab dalam proses pengiriman ke alamat yang tepat (IP, ARP, dan ICMP).
- Lapisan Transportasi, Bertanggung jawab dalam mengadakan komunikasi antar host.
- Lapisan Aplikasi, Tempat aplikasi-aplikasi yang menggunakan TCP/IP stack berada.

Dalam prakteknya, protocol stacks selalu dibagi menjadi 3 bagian besar, media, transport dan aplikasi. Sistem operasi tertentu akan memiliki minimal 2 sistem interface yang baik , yaitu satu antara lapisan media dan lapisan transport, sedang yang satunya antara lapisan transport dan lapisan aplikasi.

Sedangkan apa yang dimaksud dengan Assessing Remote Resources?

Dalam melakukan koneksi dengan sumber yang jauh, dapat dilakukan secara static maupun dinamis. Koneksi yang static dapat diketahui oleh pengguna atau syatem administrator sebelum akses ke sumber jauh tersebut terjadi. Sumber jauh tersebut akan memberikan local obyek, sumber2 dan nama layanan. Jenis koneksi ini pada dasarnya sangat sukar dikenal dan dikelola. Pengguna ataupun admin harus melakukan konfigurasi lebih dahulu setiap dilakukannya BootsUp. Jika alamat atau nama sumber jauh tersebut berubah, maka konfigurasi pada computer yang menggunakan koneksi statis harus juga dirubah.

Sedangkan yang dimaksud dengan koneksi yang dinamis adalah koneksi dimana terjadi interaksi antara resource locator dengan primary resource registration repository terjadi.

Namun hal yang paling penting dalam prinsip desain protocol jaringan adalah;

- effective
- reliability
- resiliency

Sebelum menjelaskan tentang Interprocess Communication yang pernah disinggung diatas, bagaimana bentuk-bentuk aplikasi komputasi terdistribusi yang sedang berkembang saat ini?

Berikut ini merupakan contoh dari bentuk aplikasi yang bisa dianggap sebagai kelebihan dari komputasi terdistribusi, yaitu;

1. daftar pencarian dari database dapat dibagi menjadi bagian yang lebih kecil antar computer yang ada. Pencarian dapat dilakukan bersamaan untuk setiap irisan database yang ada pada setiap computer.
2. teknik simulasi dan pemodelan yang kompleks dapat meningkatkan hasil yang akurat dengan melakukan banyak percobaan yang dilakukan secara random dan berlangsung bersamaan.
3. tehnik pencarian yang komprehensif dengan menggunakan sejumlah hasil yang telah diketahui untuk mnyelesaikan suatu masalah.
4. Model keuangan yang kompleks, perkiraan cuaca serta eksplorasi geografis, tabrakan mobil adalah contoh simulasi yang dapat dilakukan oleh komputasi terdistribusi

Lalu, apa yang dimaksud Interprocess Communications?

Inter-process Communications, atau singkatnya disebut IPC, adalah berhubungan dengan teknik dan mekanisme dimana akan memfasilitasi komunikasi antar proses. Dimana komunikasi antar proses dengan adanya 'address' yang unik mempermudah komunikasi antar proses yang satu dengan lainnya. Kernel system operasi dimana memiliki akses ke seluruh memory yang ada, akan bertindak sebagai 'communication channel'.

Perbedaan IPC dapat dikategorikan berdasarkan kondisi dan persyaratan yang ada, diantaranya;

- a) pipes
- b) fifos
- c) shared memory
- d) mapped memory
- e) message queues
- f) sockets
- g) RPC: Remote Procedure Calls

Pipes

Pipes merupakan fasilitas yang menyediakan komunikasi satu arah antar proses dalam sebuah system atau disebut half-duplex, yaitu data mengalir hanya terjadi satu arah. Pipes ini dibuat dengan berhubungan langsung dengan *pipe system call*, dimana akan menyusun deskripsi file secara berpasangan. Deskripsi ini, satu ditujukan ke sebuah pipe inode, sedangkan

yang lainya dikembalikan melalui argumen *filedes*. *Filedes* (0) digunakan untuk membaca sedangkan *filedes*(1) digunakan untuk menulis.

Fifos

Fifos (first in First out) adalah memiliki kesamaan kerja dengan pipes. Fifos juga menyediakan alur data half-duplex seperti halnya pipes. Perbedaan antara fifos dengan pipes adalah pada pipes akan selalu mengidentifikasi file systemnya dengan sebuah nama, sedangkan Fifos tidak. Fifos akan diidentifikasi dengan sebuah 'access point' dimana file tetap dalam file system. Perbedaan menyolok antara keduanya tersebut bahwa fifos ada selama proses system berlangsung, sedang pipes ada selama proses dimana pipes terbentuk.

Shared Memory

Shared memory merupakan salah satu dari 3 system V IPC mekanis yang memungkinkan proses yang berbeda dapat berkomunikasi dengan lainnya, hal ini terjadi jika proses tersebut berbagi ruang dalam virtual address, sehingga proses manapun akan berbagi wilayah memory akan mampu menulis dan membacanya.

IPC system akan membeberkan penggunaan mekanisme shared memory dengan cara 4 tahapan. Secara berurutan sebagai berikut;

- menentukan identifier dari areal shared memory
- menggunakan identifier untuk mendapatkan alamat(address) dari shared memory

- memberikan tanda pada shared memory yang telah terpakai
- akhirnya alamat tersebut digunakan untuk mengakses pengendalian, permission, mendapatkan informasi serta menghancurkan areal shared memory tersebut.

Mapped Memory

Mapped memory adalah sebuah mekanisme yang berhubungan dengan mapping sebuah file dalam file system sesuai dengan porsi memory yang ada. Pelibatan langsung yang berlebihan terhadap *system call* untuk permintaan data mungkin dapat ditolak oleh mekanisme mapped memory. Hal ini terjadi disebabkan tersedianya porsi pada virtual memory.

Message Queues

Message Queues merupakan salah satu bentuk dari 3 mekanisme system V IPC. Mekanisme ini memungkinkan proses pengiriman informasi kepada yang lain secara asynchronous. Istilah asynchronous digunakan dalam konteks bahwa proses penguruman berlanjut disertai sebuah eksekusi tanpa harus menunggu penerima menerima atau mengenal informasi tersebut. Dengan kata lain, pengirim tidak akan menunggu jika tidak ada informasi bahwa terjadi antrian disana. Antrian tersebut adalah antrian yang dilakukan dan dikontrol oleh kernel.

Sockets

Istilah socket telah ipopulerkan oleh Berkeley Unix Networking System, sebagai titik akhir dari komunikasi dimana akan melakukan korespodensi kepada sebagian defenisi dari asosiasi.. Asosiasi adalah sebuah komunikasi antar 2 proses yang berjalan pada 2 system computer. Pendefinisian dari sebagian asosiasi meliputi, protocol, local-adress, local process, remote-adress dan remote process.

Tabel Metode IPC

Metode	Disediakan oleh (Operating Systems or lingkungan lainnya)
File	All operating systems
Signal	All operating systems
Socket	Most operating systems
Pipe	All POSIX systems
Named pipe	All POSIX systems
Semaphore	All POSIX systems
Shared memory	All POSIX systems
Message passing (shared nothing)	Used in MPI paradigm, Java RMI, CORBA and others
Memory map	All POSIX systems; may carry 'race condition' risk if a temporary file is used
Message queue	Most operating systems
Mailbox	Some operating systems

Remote Procedure Calls

RPC adalah sebuah protokol yang memungkinkan program komputer berjalan pada satu host dan mengakibatkan kode dapat dieksekusi pada host yang lain tanpa kebutuhan programmer secara eksplisit menkodekan ini. Ketika

kode yang dipertanyakan tersebut ditulis dengan prinsip-prinsip berorientasi obyek, RPC adalah sering disandarkan sebagai perintah jarak jauh atau metode perintah jarak jauh.

RPC adalah paradigma yang mudah dan populer dalam mengimplementasikan model client-server dari komputerisasi terdistribusi. Sebuah RPC ada berkat inisiatif Client (caller) yang mengirimkan pemesanan kepada system jarak jauh (server) untuk mengeksekusi prosedur tertentu yang disertai argumen. Hasil pemesanan tersebut dikembalikan oleh klien (caller). Dalam hal ini banyak sekali variasi dan penyimpangan dalam berbagai macam implementasinya, sebagai akibat bervariasinya protokol RPC yang berbeda.

Untuk memungkinkan server dapat diakses oleh Client yang berbeda, sejumlah standarisasi system RPC telah dibuat. Kebanyakan ini menggunakan IDL (interface Description Language) yang memungkinkan berbagai macam platform menggunakan RPC. Contoh dari system tersebut diantaranya Sun RPC (disebut juga ONC RPC), DCE (Distributed Computing Environment), Microsoft DCOM dan activeX yang berbasis DCE, serta CORBA.

Saat ini banyak vendor telah mulai menggunakan XML sebagai IDL, dan HTTP sebagai network protocols. Kelebihan dari sistem ini, dikenal sebagai web services, adalah standarisasi dan sederhana dimana IDL adalah text file yang dipahami secara luas serta HTTP yang dibangun diatas sistem operasi yang modern. Satu contoh dari system RPC adalah SOAP, dikembangkan dari system sebelumnya XML-RPC.

Untuk memungkinkan server dapat diakses oleh Client yang berbeda, sejumlah standarisasi system RPC telah dibuat. Kebanyakan ini menggunakan IDL (interface Deskription Language) yang memungkinkan berbagai macam platform menggunakan RPC. Contoh dari system tersebut diantaranya Sun RPC (dijuga juga ONC RPC), DCE (Distributed Computing Enviroment), Microsoft DCOM dan activeX yang berbasis DCE, serta CORBA.

Saat ini banyak vendor telah mulai menggunakan XML sebagai IDL, dan HTTP sebagai network protocols. Kelebihan dari sistem ini, dikenal sebagai web services, adalah standarisasi dan sederhana dimana IDL adalah text file yang dipahami secara luas serta HTTP yang dibangun diatas sistem oprasi yang modern. Satu contoh dari system RPC adalah SOAP, dikembangkan dari system sebelumnya XML-RPC.

Apa yang dimaksud dengan SOAP, CORBA, COM, DCOM, COM+, ActiveX tersebut?

SOAP adalah standar untuk bertukar pesan-pesan berbasis XML melalui jaringan komputer atau sebuah jalan untuk program yang berjalan pada suatu sistem operasi (OS) untuk berkomunikasi dengan program pada OS yang sama maupun berbeda dengan menggunakan HTTP dan XML sebagai mekanisme untuk pertukaran data. **Extensible Markup Language (XML)** adalah bahasa markup serbaguna yang direkomendasikan W3C untuk mendeskripsikan berbagai macam data. XML menggunakan *markup tags*

seperti halnya HTML namun penggunaannya tidak terbatas pada tampilan halaman web saja.

SOAP menspesifikasikan secara jelas bagaimana cara untuk meng-*encode header* HHTP dan *file* XML sehingga program pada suatu komputer dapat memanggil program pada komputer lain dan mengirimkan informasi, dan bagaimana program yang dipanggil memberikan tanggapan.

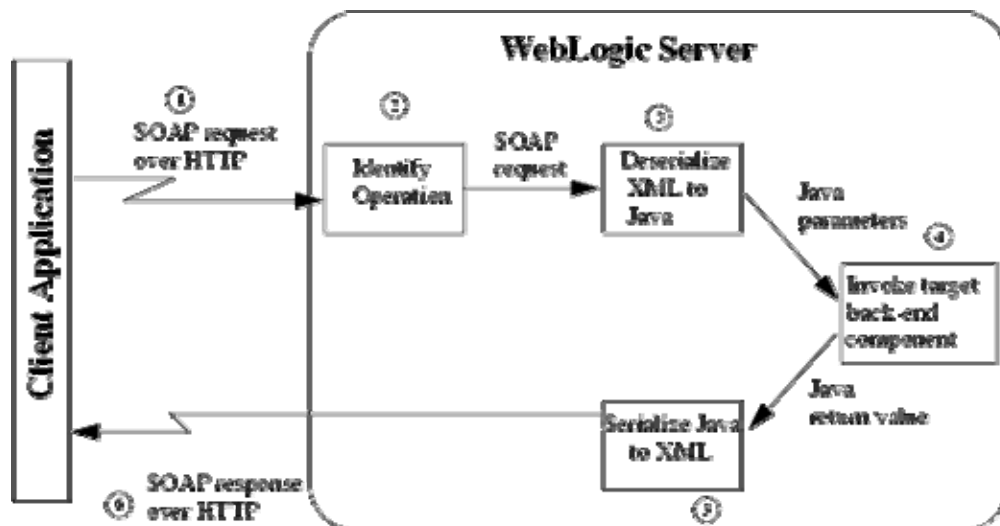
SOAP adalah protokol ringan yang ditujukan untuk pertukaran informasi struktur pada lingkup desentralisasi, dan terdistribusi. SOAP menggunakan teknologi XML untuk mendefinisikan rangka kerja pemesanan terekstrensi di mana menyediakan konstruksi pesan yang dapat dipertukarkan pada protokol berbeda. Rangka kerja dirancang bebas dari model pemrograman dan spesifikasi implementasi semantik.

SOAP menjadi sangat mudah diterima oleh berbagai pihak – terutama oleh berbagai vendor TI – dikarenakan protokol ini memanfaatkan berbagai teknologi yang sudah ada sebelumnya dan sudah banyak digunakan. Misalnya untuk protokol transport, yang paling banyak digunakan adalah HTTP, walaupun dimungkinkan untuk menggunakan protokol transport lainnya. Sedangkan untuk format data atau message digunakan XML yang tidak diragukan lagi manfaat dan perannya di dalam pertukaran data. Dengan demikian, tidaklah terlalu mengherankan bila kemudian SOAP dianggap sebagai solusi penyelamat untuk mengatasi berbagai masalah yang dihadapi oleh teknologi – teknologi pendahulunya.

Implementasi SOAP sendiri kemudian disadari tidaklah mudah dan sesederhana yang dibayangkan. Karena SOAP sendiri tidak lebih dari

sekedar spesifikasi, yang mencoba untuk mendeskripsikan dan mendefinisikan setiap aspek dan mekanisme yang ada dalam sebuah RPC. Hal ini yang kemudian banyak membuat kerancuan dan kebingungan ketika seorang developer pertama kali mempelajari SOAP. Selain harus membiasakan diri dan menjadi fasih dengan spesifikasi SOAP, seorang developer juga harus fasih dengan spesifikasi dari tool atau program yang digunakan untuk menghasilkan sebuah message SOAP dan juga untuk membaca serta mengartikan sebuah message SOAP yang diterima. Benar, seorang developer akan memerlukan software atau tool untuk dapat berinteraksi dengan SOAP.

Di sinilah terjadi persaingan dalam hal implementasi SOAP, di mana beberapa vendor mengeluarkan seperangkat tool untuk menghasilkan dan menerima message SOAP seperti Microsoft yang mengeluarkan SOAP Toolkit dan sekarang sudah mencapai versi 2.0, Apache SOAP, PERL/SOAP Lite Client, dan sebagainya. Di sini terletak beberapa perbedaan dalam hal implementasi SOAP, di mana message SOAP yang dihasilkan oleh suatu toolkit belum tentu dapat diterima dan digunakan oleh pihak lain yang menggunakan platform berbeda. Banyak hal yang menyebabkan terjadinya masalah – masalah tersebut, dan tidak selalu hal tersebut merupakan masalah yang ada di SOAP itu sendiri, melainkan juga masalah – masalah dalam menginterpretasikan kode – kode syntax HTTP maupun syntax dari XML itu sendiri.



Gambar 13.1

Komunikasi Client-server dengan pesan SOAP

1. Aplikasi klien mengirimkan pesan meminta SOAP untuk aplikasi server disepanjang jaringan
2. Berdasarkan URL yang diminta , server mengidentifikasi web service dalam keadaan invoke
3. Web service membaca pesan yg diminta SOAP dan mengidentifikasi operasi yang ingin dijalankan. Operasi menyesuaikan ke method dari penanggung komponen, untuk di invoke ke langkah selanjutnya. Konversi form XML ke java terjadi saat parameter permintaan dari pesan SOAP di web service untuk invoked operasi.
4. Web Service membaca pesan permintaan SOAP dan mengidentifikasi operasi yang dibutuhkan. Operasi ini akan melakukan korepodensi terhadap suatu metode dari komponen terakhir (back-end), yang kemudian akan panggil kemudian. Konversi dari XML ke java akan terjadi

sebagai parameter permintaan atas pesan SOAP pada lapisan web service dalam operasi pemanggilan.

5. Metode komponen back-end yang cocok akan diminta oleh parameter Java yang lalu akan dipanggil kembali.
6. menjelang selesainya komponen back-end akan dikirim kembali sebagai respon web service serta kemudian akan dikonversi kembali menjadi bentuk XML dari bentuk java. Pemaketan tersebut dijadikan sebagai respon pesan SOAP.
7. Web Service akan mengirim kembali respon pesan SOAP ke aplikasi Client yang melakukan pemanggilan web service.

Yang patut disyukuri adalah, walaupun beberapa masalah terus ditemukan, namun jumlah kesuksesan dari hasil uji coba dan implementasi langsung di berbagai project terus menunjukkan angka yang lebih tinggi, sehingga keyakinan akan keuntungan yang dihasilkan dari implementasi SOAP dibandingkan teknologi pendahulunya tetap diyakini oleh berbagai pihak sebagai solusi yang lebih baik. Di samping tentu saja, upaya – upaya untuk menstandarisasikan mekanisme menjadi lebih universal juga terus dilakukan.

CORBA

Interoperabilitas adalah kemampuan saling bekerjasama antar sistem komputer. Sebenarnya interoperabilitas bukanlah barang baru, karena protokol komunikasi datapun (TCP/IP misalnya) pada dasarnya diciptakan

untuk mewujudkan interoperabilitas. Yang belum banyak dikenal adalah interoperabilitas pada level perangkat lunak aplikasi.

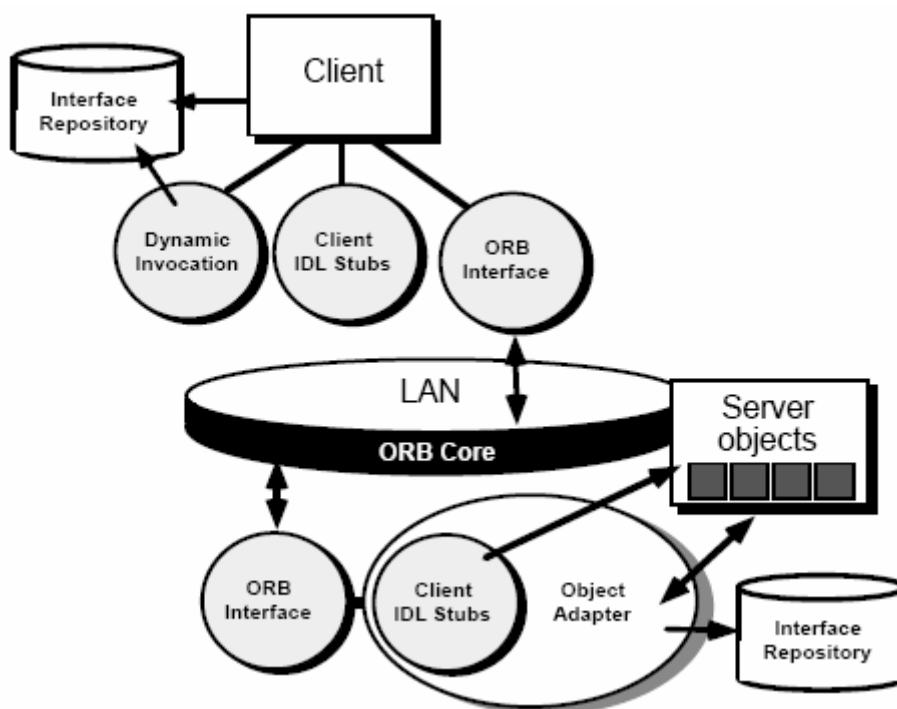
Dalam konteks sistem komputasi terdistribusi, meskipun komponen-komponen aplikasi dibuat dengan bahasa pemrograman yang berbeda, menggunakan development tools yang berbeda, dan beroperasi di lingkungan yang beragam, mereka tetap harus dapat saling bekerjasama.

Interoperabilitas perangkat lunak menuntut homogenitas pada suatu level tertentu. Untuk itu diperlukan semacam 'standarisasi'. Berawal dari keperluan ini lahirlah CORBA (Common Object Request Broker Architecture). CORBA adalah hasil 'kesepakatan' antara sejumlah vendor dan pengembang perangkat lunak terkenal seperti IBM, Hewlett-Packard, dan DEC, yang tergabung dalam sebuah konsorsium bernama OMG (Object Management Group) yang beranggotakan kurang lebih 500 perusahaan. CORBA ini telah mendapatkan pengakuan sebagai standar arsitektur untuk obyek terdistribusi (komponen) oleh ISO-International Organization for Standardization dan X/Open.

CORBA adalah sebuah arsitektur software yang berbasis pada teknologi berorientasi obyek atau *Object Oriented* (OO) dengan paradigma *client-server*. Dalam terminologi OO, sebuah obyek berkomunikasi dengan obyek lain dengan cara pengiriman pesan (message passing). Konteks komunikasi ini kemudian dipetakan ke dalam model *client-server*. satu obyek berperan sebagai *client* (si pengirim pesan) dan yang lain bertindak sebagai *server* (yang menerima pesan dan memroses pesan yang bersangkutan). Sebagai contoh, dalam ilustrasi di awal tulisan ini, jika si pasien memerlukan obat

tertentu, maka obyek aplikasi di tempat praktek dokter berlaku sebagai *client* dan mengirim pesan ke obyek aplikasi di apotik guna mengetahui apakah obat yang diperlukan tersedia di sana.

Keunikan dari CORBA adalah kemampuannya dalam menangani heterogenitas antara *client* dan *server* (dalam terminologi CORBA, obyek *server* dinamakan **implementasi obyek** (*object implementation*)). Keduanya dapat saja diimplementasikan dalam hardware, sistem operasi, bahasa pemrograman, dan di lokasi yang berbeda, tetapi tetap bisa saling berkomunikasi. Kuncinya ada pada sebuah lapisan software yang disebut dengan **ORB** (*Object Request Broker*).



Gambar 13.2 Arsitektur Corba dasar

Keterangan gambar :

ORB adalah middleware yang memungkinkan suatu object yang berada di client mengirim message ke suatu method yang ter-encapsulated oleh suatu object yang ada di server. Artinya,

ORB meneruskan message dan menangani semua komunikasi dan koordinasi yang dibutuhkan untuk menemukan object yang dimaksud oleh message tadi, memanggil methodnya, melewatkan data yang sesuai ke object dan kirim data hasil kembali ke object yang membuat message tadi. Standar yang umum digunakan untuk mengimplementasikan ORB adalah : CORBA, COM, dan JavaBeans. Struktur sederhana CORBA adalah seperti pada Gambar 13.2 Saat CORBA diimplementasikan di sistem C/S, object dan object dan class object pada client dan server didefinisikan menggunakan interface description language (IDL).

Konsep dasar dari CORBA adalah Broker Terhadap Permintaan Obyek (ORB-Object Request Broker). ORB mendukung dalam sebuah jaringan klien dan server pada computer yang berbeda, yang berarti program pada klien dapat meminta layanan atau obyek dari program server tanpa harus memahami dimana server itu berada dalam jaringan terdistribusi serta interface apa yang digunakan. Untuk melakukan timbale balik permintaan dan pengembalian antara ORB, program akan menggunakan General ORB protocols (GIOP) sedang untuk internet, menggunakan Internet Inter-ORB Protocol (IIOP). IIOP akan memetakan permintaan GIOP dan mengembalikannya ke lapisan TCP (Internet's Transmission Control Protocol) dari setiap computer yang ada.

Hal yang perlu dicatat bahwa CORBA sangat mendukung Microsoft, dimana Microsoft memiliki arsitektur komponen obyek terdistribusi sendiri (DCOM-Distributed Component Object Model. Akan tetapi CORBA dan Mocosoft telah bersepakat dalam pendekatan 'Gateway' sehingga obyek klien akan dikembangkan dengan model obyek-komponen akan dapat melakukan komunikasi dengan server CORBA atau sebaliknya.

Lingkungan Komputasi Terdistribusi (DCE-Distributed Computing Environment) adalah arsitektur pemograman terdistribusi yang kelahirannya

mendahului trend pemograman berorientasi obyek dan CORBA. DCE pada saat ini banyak digunakan oleh sejumlah perusahaan besar dan akan tetap keberadaanya bersama dengan CORBA dan dikemudian hari akan muncul program yang menjembatani keduanya.

COM

COM, Component Object Model adalah arsitektur aplikasi lunak yang memungkinkan terbentuknya aplikasi dari komponen aplikasi lunak binari. COM adalah arsitektur utama yang menjadi dasar bagi pelayanan aplikasi level tinggi, seperti pelayanan oleh OLE. Pelayanan OLE meliputi aspek yang sangat variatif yang secara sistem fungsi sangat dibutuhkan, termasuk dokumentasi kompleks, kontrol kebiasaan, aplikasi inter script, trasfer data serta interaksi antar perangkat lunak.

Dalam tehnologi informasi, dokumentasi yang kompleks adalah koleksi tidak terorganisasi dari interface pengguna yang berbentuk dari lingkungan persepsi yang sederhana dan terpadu. Dokumentasi kompleks termasuk struktur data yang terdiri dari bentuk data yang berbeda, seperti text, file audio, dan file video bergerak. Dokumentasi kompleks juga merupakan lingkungan aplikasi yang terdiri dari obyek program yang dapat melakukan intelink dan interaksi dengan pengguna.

Dokumentasi kompleks dapat berbentuk sebagian dari informasi yang berasal dari sumber yang berbeda dan dikembangkan tanpa seorangpun tahu. Desktop Internet Explorer dari Microsoft menggunakan konsep dokumentasi kompleks. OLE nya Microsoft adalah kerangka dasar dalam mengembangkan dan mengelola dokumentasi kompleks. OpenDoc adalah merupakan standar

alternatifnya. Dengan demikian COM adalah kerangka dasar Microsoft dalam mengembangkan dan mendukung program obyek-komponen. Di dalam pemrograman berorientasi obyek dan teknologi obyek terdistribusi, sebuah komponen adalah program yang tidak digunakan dalam mengembangkan bagian yang dapat dikombinasikan dengan komponen lain yang ada pada komputer yang ada pada jaringan terdistribusi untuk membentuk aplikasi.

ActiveX

ActiveX adalah nama yang diberikan Microsoft untuk strategi yang ada pada tool dan teknologi pemrograman berorientasi obyek. Teknologi utamanya adalah Component Object Model (COM), digunakan dalam jaringan dengan dukungan dari directory, COM menjadi DCOM-Distributed Component Object Model. Hal utama dalam membuat program, menkondisikan lingkungan ActiveX adalah merupakan suatu komponen yang mampu menyesuaikan untuk bisa berjalan dengan baik. Komponen ini disebut ActiveX Control. ActiveX adalah tantangan dari Microsoft terhadap teknologi Java buatan Sun Microsystems.

DCOM

DCOM (Distributed Component Object Model) adalah kumpulan konsep Microsoft dan program interface dimana obyek program dari klien dapat meminta pelayanan dari server atas program obyek pada komputer di dalam jaringan. DCOM ini berbasis COM, dimana menyediakan sekumpulan

interface yang memungkinkan klien dan server untuk melakukan komunikasi pada komputer yang sama.

COM+

COM+ adalah pengembangan dari COM, sebagai suatu strateginya Microsoft dalam mengembangkan program tandingan dalam program aplikasi. Keduanya merupakan arsitektur pemograman beroentasi obyek serta sekumpulan pelayanan sistem operasi. COM+ memberikan tambahan pada COM suatu sistem pelayanan yang baru untuk aplikasi komponen ketika sedang berjalan. COM+ cenderung memberikan model guna memudahkan dalam membuat aplikasi bisnis pada Microsoft Transaction Server (MTS) pada Windows NT. Ini merupakan jawaban atas produk Sun Microsystem-IBM-Oracle yang dikenal dengan Enterprise Java Beans (EJB).

Ketika beberapa sumber terdistribusi antar jaringan, bahwa sumber yang satu (pengguna) dengan penyedia (provider) harus saling bertemu.

Mekanisme apa yang diperlukan?

Mekanisme yang diperlukan adalah adanya layanan direktori. **Direktori** adalah sebuah koleksi sistem-sistem terbuka yang saling berhubungan untuk menampung sebuah basis data berisi informasi tentang sekumpulan objek pada dunia nyata. Hodges memandang direktori sebagai sebuah basis data yang memiliki karakteristik antara lain dirancang untuk lebih sering melakukan pencarian dan dibaca dibandingkan ditulis, dan hanya dapat melakukan

update sederhana tanpa adanya transaksi rumit, yang merupakan ciri khas *relational database*.

Contoh dari direktori adalah *Yellow Pages* yang berisi data nama orang, alamat, dan nomor teleponnya, yang dikategorikan menurut lokasi dan nama.

Untuk mempercepat pencarian, direktori biasanya diimplementasikan secara elektronik. Informasi disimpan menggunakan suatu tempat penyimpanan yang terdistribusi yang kemudian dapat di-*query* oleh pengguna dan aplikasi.

Pengimplementasian direktori menggunakan cara di atas berikut layanan pengaksesannya menggunakan sebuah protokol jaringan disebut **directory service**. Salah satu standar directory service yang terkenal adalah LDAP yaitu Lightweight Directory Access Protocols.

Lightweight Directory Access Protokol (LDAP)

International Telecommunication Union (ITU) dan *International Organization for Standardization* (ISO) pada tahun 1990 mengeluarkan standar untuk *directory service* yang diberi nama **X.500**. Ide dasar dari X.500 adalah untuk membangun sebuah sistem terdistribusi secara global yang dapat menawarkan pengaksesan informasi secara homogen dan komprehensif. Dalam X.500 terdefinisi **Directory Access Protocol** (DAP) yang dapat digunakan oleh klien untuk mengakses direktorinya. Secara umum X.500 bertujuan untuk memudahkan pencarian data yang tersusun secara hirarkis. Karena implementasi X.500 *service protocol* terlalu berat untuk *desktop* pada saat itu, pada tahun 1993, akademisi dari Universitas Michigan, dengan bantuan Konsorsium ISODE merancang dan membangun sebuah protokol

yang dapat bekerja di atas TCP/IP. Hasilnya adalah ***Lightweight Directory Access Protocol (LDAP)***. Protokol ini dirancang untuk menyediakan akses ke direktori X.500 tanpa perlu memiliki sumber daya yang dibutuhkan untuk mengimplementasikan DAP.

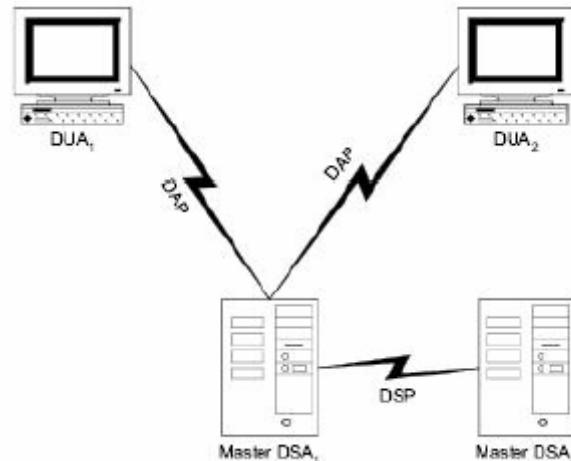
Microsoft, sebuah perusahaan perangkat lunak, melihat layanan direktori ini sebagai komponen kritical dalam sebuah sistem yang mencari dan mengirimkan informasi. Oleh karena itu, pada tahun 1996 Microsoft membuat implementasi layanan ini yang diberi nama ***Active Directory (MSAD)*** dan dirilis pertama kali bersamaan dengan penggunaan *Windows 2000* yang kemudian ditingkatkan fungsionalitasnya pada *Windows Server 2003* [WIK05]. MSAD *men-support* sejumlah standar dalam layanan ini antara lain sebagian dari DAP dan LDAP.

Bagaimana Cara Kerja X.500 tersebut, lalu apa perbedaan dengan LDAP?

X.500 berbasiskan model *client-server*. Secara terminologi dari standar X.500, direktori kliennya disebut dengan *Directory User Agent (DUA)*, dan untuk *server* disebut dengan *Directory System Agent (DSA)*. Protokol yang mengatur interaksi antaradua atau lebih DSA adalah *Directory System Protocol (DSP)*, dan *Directory Access Protocol (DAP)* yang mengatur komunikasi antara DUA dan DSA. DAP melewati semua lapisan jaringan yang didefinisikan pada OSI. Klien mengakses *directory* dengan membangun suatu koneksi dengan *server* dengan mengeluarkan permintaan akses. Klien dapat menghubungi semua *server* dengan sebuah permintaan. Jika data yang dibutuhkan tidak terdapat pada segmen dari *Directory Information Base*

(DIB) pada suatu *server*, maka *server* tadi akan mencarikannya pada *server* lain atau menyambungkan klien (*redirecting*) dengan *server* lain.

Berikut ilustrasinya :



Gambar Cara Kerja X-500

Data yang ada disimpan di X.500 *server* menggunakan struktur data **tree** (yang diberi nama *Directory Information Tree* – DIT) dengan *nodes* yang diberi nama sebagai atribut pencarian. Menurut Coulouris, data tersebut diakses menggunakan dua jenis perintah, *read* dan *search*. Untuk *read*, klien sudah tahu pengenalan data yang diinginkan (*identifier*) memakai nama absolut, sehingga DSA tinggal mengambil data yang diinginkan menggunakan navigasi DIT. Sementara pada *search*, klien hanya memberikan sebuah filter dan basis nama dan DSA diminta untuk mencari data pada DIT yang sesuai dengan filter yang diberikan.

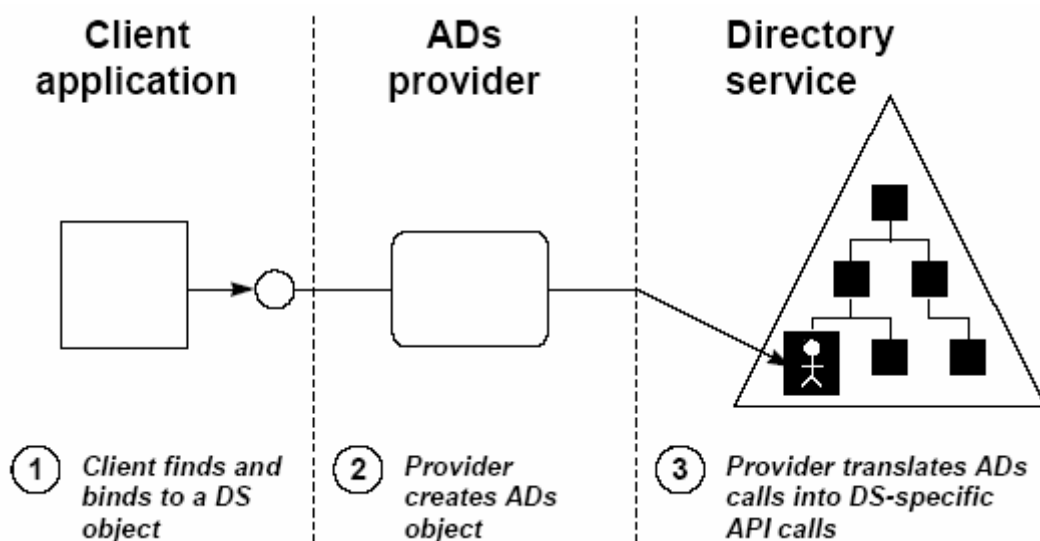
Sedangkan cara kerja LDAP mirip dengan X.500, karena LDAP dapat dikatakan merupakan himpunan bagian dari X.500. LDAP menggunakan model *client-server*.

Pemrosesan *query* dari klien adalah serupa dengan X.500. Untuk melakukan

sebuah *query*, klien mengirimkan *identifier* data (disebut *Relative Distinguished Name*) yang ingin diambil nilai atribut-atributnya. Klien mengirimkan pesan itu di atas TCP, dan *server* mencoba mencarinya pada DIT yang tersimpan di *server* tersebut. Bila ditemukan maka hasilnya langsung diberikan ke klien, bila tidak maka hasilnya berupa sebuah pointer ke *server* lain yang menyimpan data yang dicari.

Membicarakan Directory Service, MS Active Directory (MSAD) merupakan arsitektur yang tidak bisa diabaikan, bisa tolong dijelaskan?

Arsitektur *Active Directory* juga didasarkan atas model *client-server* (*client-provider* menurut istilah yang digunakan Steinberger) dimana *Active Directory API* menjadi pusat dari sebuah *Open Directory Services Interface* (ODSI). Sama seperti X.500, *Active Directory* juga menggunakan struktur data *tree* dengan setiap *leaf* berisi sebuah *domain*. Gambar di bawah ini menggambarkan bagaimana client ingin mendapatkan dan menggunakan interface dari suatu obyek.



Gambar Arsitektur MS Active Directory

Sebuah server Active Directory Interfaces menyimpan implementasi objek dan semua dependensinya untuk sebuah *namespace* tertentu. Dengan demikian, seperti yang terlihat pada gambar 13.4, klien hanya perlu mengambil dan menggunakan antarmuka dari objek tersebut, tanpa perlu memusingkan di mana dan bagaimana objek tersebut diimplementasikan.

	X.500	LDAP	MSAD
<i>Platform</i>	Semua	Semua	Hanya Windows
Keperluan sumber daya	Berat	Ringan	Ringan
Lapisan OSI yang digunakan	Semua	TCP/IP	TCP/IP
Autentikasi	Mendukung	Mendukung	Mendukung, hanya klien Windows
Model Skema	Kompleks	Sederhana	Sederhana

Tabel Perbandingan X-500, LDAP, MSAD

Alasan mengapa LDAP dikembangkan adalah karena X.500 terlalu kompleks untuk dikembangkan dan hasil implementasinya berjalan sangat lambat pada komputer saat itu. Alasan mengapa X.500 lambat adalah karena DAP membutuhkan data melalui semua lapisan jaringan OSI, dan pemrosesannya membutuhkan memori dan waktu prosesor yang besar. MSAD memiliki karakteristik yang serupa dengan LDAP, karena MSAD dibangun berbasis LDAP. Akan tetapi, MSAD memiliki perbedaan dalam cara pemodelan data. Di lain pihak, X.500 memiliki cakupan yang lebih luas dari sekedar protokol untuk mengakses sebuah direktori, karena X.500 juga mendiskusikan konsep sebuah direktori. Dapat juga dilihat pada tabel 1, X.500 dan LDAP dapat

digunakan pada platform manapun, sementara MSAD hanya dapat digunakan pada Windows. Sebagai efek samping, layanan autentikasi pada MSAD yang didukung oleh sistem direktori secara umum hanya dapat digunakan oleh klien Windows. Padahal, sebuah *directory service* sebaiknya mendukung autentikasi lintas *platform* untuk memudahkan sentralisasi *password database* yang bisa digunakan untuk memberikan wewenang kepada pengguna yang memakai *Unix* dan *Windows* maupun sistem lain seperti *Macintosh* atau *Netware*.

Referensi :

1. <http://id.wikipedia.org/wiki/XML>
2. http://en.wikipedia.org/wiki/distributed_computing
3. <http://id.wikipedia.org/wiki/Internet>
4. <http://www.centipendia.com/articles/>
5. <http://www.extremetech.com/>
6. <http://library.gunadarma.ac.id/files/disk1/2/jbptgunadarma-gdl-s1-2004-fritaromau-70-bab2.pdf>
7. Ilham kurnia, Gst Agus komang Lastrawan. 2005. *X.500 – LDAP – MS Active Directory*.
<http://telaga.cs.ui.ac.id/WebKuliah/sisdis2005/SP2/SP2-Directories-02-Summary.pdf>
8. Lukito Edi Nugroho. Mengenal Mengenal CORBA: Platform Untuk Komputasi Terdistribusi.
<http://www.infolinux.co.id/sections.php?op=viewarticle&artid=40>.
9. Sutarman, S.Kom. Pemrograman Client Server.
<http://kuliah.armediaweb.com/clientserver/dasarcsr.pdf>.
10. Stephen D. Burd. 2003. *System Architecture a division of Thomson Learning*. Mexico.
11. Tutang, SE. SOAP. <http://blogs.netindonesia.net/tutang/articles/3074.aspx>.
12. Umi Proboyekti, S.kom, MLIS. Rekayasa Software Client/Server.
<http://lecturer.ukdw.ac.id/othie/bhn10.pdf>