SISTEM TERDISTRIBUSI

RAHMAT ROBI WALIYANSYAH, M.KOM.

Universitas PGRI Semarang

KATA PENGANTAR

Bismillahirrahmaanirrahiim,

Puji dan syukur penulis panjatkan kehadirat Allah SWT karena berkat Rahmat dan Hidayah-Nya, serta segala kemudahan yang telah diberikan, sehingga penulis dapat menyelesaikan pembuatan modul mata kuliah Sistem Terdistribusi. Modul ini telah kami susun dengan maksimal dan mendapatkan bantuan dari berbagai pihak sehingga dapat memperlancar pembuatan modul ini. Untuk itu kami menyampaikan banyak terima kasih kepada semua pihak yang telah berkontribusi dalam pembuatan tugas ini.

Terlepas dari semua itu, Kami menyadari sepenuhnya bahwa masih ada kekurangan baik dari segi susunan kalimat maupun tata bahasanya. Oleh karena itu dengan tangan terbuka kami menerima segala saran dan kritik dari pembaca agar kami dapat memperbaiki modul ini.

Akhirnya hanya ucapan terima kasih dan doa yang bisa penulis sampaikan kepada semua pihak. Hanya Allah-lah sebaik-baiknya pemberi balasan, semoga Allah memberi rahmat, taufik, hidayah, dan karunia-Nya kepada kita semua, serta memberikan ganjaran yang setimpal atas semua bantuan yang diberikan. Amin Ya Robbal Alamin.

Semarang, Maret 2020

Penyusun

DAFTAR ISI

PENGERTIAN SISTEM TERDISTRIBUSI

1.1 Pengertian Distribusi

Anda pasti pernah melihat seseorang yang memikul barang tertentu untuk ditawarkan kepada pembeli, contoh seperti tukang sayur, tukang bakso. Kegiatan yang dilakukan oleh orangorang tersebut merupakan kegiatan distribusi. Distribusi artinya proses yang menunjukkan penyaluran barang dari produsen sampai ke tangan masyarakat konsumen. Produsen artinya orang yang melakukan kegiatan produksi. Konsumen artinya orang yang menggunakan atau memakai barang/jasa dan orang yang melakukan kegiatan distribusi disebut distributor.

Distribusi merupakan kegiatan ekonomi yang menjembatani kegiatan produksi dan konsumsi. Berkat distribusi barang dan jasa dapat sampai ke tangan konsumen. Dengan demikian kegunaan dari barang dan jasa akan lebih meningkat setelah dapat dikonsumsi. Dari apa yang baru saja diuraikan, tampaklah bahwa distribusi turut serta meningkatkan kegunaan menurut tempatnya (place utility) dan menurut waktunya (time utility).

a. Sistem distribusi jalan pendek atau langsung

   Adalah sistem distribusi yang tidak menggunakan saluran distribusi. Contoh distribusi sistem ini adalah penyaluran hasil pertanian oleh petani ke pasar langsung.

Bagan sistem distribusi ini sebagai berikut.

b. Sistem distribusi jalan panjang atau tidak langsung

   Adalah sistem distribusi yang menggunakan saluran distribusi dalam kegiatan distribusinya biasanya melalui agen. Contoh: motor, mobil, TV.

Bagan sistem distribusi tidak langsung.

Definsi sistem terdistribusi adalah :

1) sebuah sistem dimana komponen hardware atau software-nya terletak dalam suatu jaringan komputer dan saling berkomunikasi dan berkoordinasi mengunakan message pasing.
2) sebuah sistem yang terdiri dari kumpulan dua atau lebih komputer dan memiliki koordinasi proses melalui pertukaran pesan synchronous atau asynchronous.
3) kumpulan komputer independent yang tampak oleh user sebagai satu sistem komputer
4) kumpulan komputer autonom yang dihubungkan oleh jaringan dengan software yang dirancang untuk menghasilkan fasilitas komputasi terintegrasi Dapat terlihat dari

bebarapa pengertian diatas dapat di tarik kesimpulan bahwa sistem terdistribusi adalah sebuah sistem yang terdiri dari beberapa komponen software atau hardware yang independent yang berkomunikasi dan berkoordinasi melalui message parsing baek sinkron maupun asinkron yang telihat satu kesatuan dan dirancang untuk menghasilkan fasilitas komputasi terintegrasi.

contoh dari sistem terdistribusi adalah

1) Internet, global jaringan interkoneksi computer yang berkomunikasi melalui IP (Internet Protocol) Protocol.
2) Intranet, jaringan teradministrasi terpisah dengan batasan pada kebijakan keamanan local.
3) Mobile dan komputasi diberbagai tempat, laptops, PDA, mobile phone,printers, peraltan rumah, dll.
4) World Wide Web (www), sistem untuk publikasi dan akses sumber daya dan layanan melalui Internet.
5) workstation network.
6) automatic banking (mesin teler) system.
7) automotive system(sistem distribusi real-time).

1.2 Tujuan Distribusi

Tujuan kegiatan distribusi baik yang dilakukan oleh individu atau lembaga adalah sebagai berikut :

a. Kelangsungan kegiatan produksi dapat terjamin

Produsen atau perusahaan membuat barang untuk dijual dan mendapatkan keuntungan dari hasil penjualan yang kembali digunakan untuk proses produksi dimana keuntungan tersebut didapatkan jika terdapat distributor.

b. Barang atau Jasa Hasil Produksi dapat bermanfaat bagi konsumen.

Barang atau jasa produksi tidak akan ada artinya jika tetap berada di tempat produsen. Barang atau jasa dapat bermanfaat bagi konsumen jika telah ada kegiatan distribusi.

c. Konsumen Memperoleh Barang dan Jasa dengan Mudah.

Tidak semua barang atau jasa dapat dibeli langsung konsumen dari produsen dimana hal ini membutuhkan penyalur atau distribusi dari produsen ke konsumen.

1.3 Tugas Distribusi

Mengklasifikasi barang atau memilahnya sesuai dengan jenis, ukuran, dan kualitasnya. Memperkenalkan barang atau jasa yang diperdagangkan kepada konsumen, seperti dengan reklame atau iklan. Membeli barang dan jasa dari produsen atau pedagang yang lebih besar.

1.4 Contoh Kegiatan Distribusi

a. Pedagang menjual berbagai jenis barang,seperti beras dan sayuran
b. Ibu Ijah mempunyai perusahaan roti, biasanya ibu ijah menitip ke rotinya di toko untuk dijual dikonsumen
c. Ayah sedang memberi nasi kuning di warung.
d. Pertamina menyalurkan bensin dan solar ke SPBU

MODEL SISTEM TERDISTRIBUSI

2.1 Model Sistem Terdistribusi

Model dalam sistem terdistribusi :

- Model Arsitektur (Architectural Models)
- Model Interaksi (Interaction Models)
- Model Kegagalan (Failure Models)

Resource dalam sistem terdistribusi dipakai secara bersamaan oleh users. biasanya dibungkus dalam suatu komputer dan dapat di akses oleh komputer lain dengan komunikasi. Setiap resource di atur oleh program yang disebut dengan resource manager. Resource manager memberikan kemungkinan komunikasi interface antar resource. Resource manager dapat di generalisasi sebagai proses, kalau sistem di design dengan sudut pandang objek (Object Oriented), resource dibungkus dalam suatu objek.

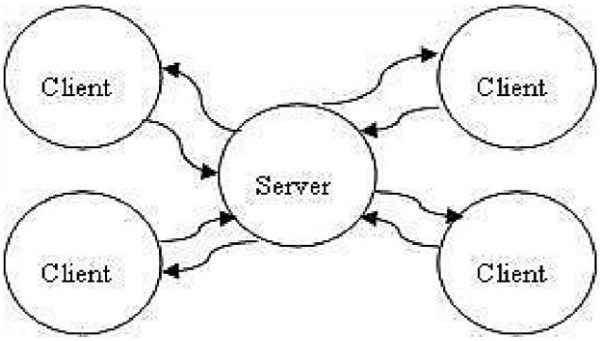
2.2 Model Arsitektur (Architectural Models)

Bagaimana cara kerja sistem terdistribusi antara komponen-komponen sistem dan bagaimana komponen tersebut berada pada sistem terdistribusi :

2.2.1 Client - Server Model

Sistem yang terdiri dari kumpulan - kumpulan proses disebut dengan server, dan memberikan layanan kepada user yang disebut dengan client. Model client-server biasanya berbasiskan protokol request/reply. Contoh implementasinya antara lain:

RPC (Remote Procedure Calling) dan RMI (Remote Method Invocation).



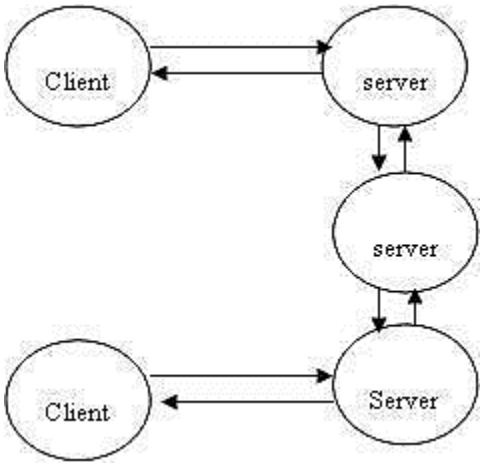
- Client:
- Proses akses data
- Melakukan operasi pada komputer lain
- Server:

- Proses mengatur data
- Proses mengatur resources
- Proses komputasi
- Interaksi:
- Invocation/result

Karakteristik Client-Server

1. *Service* : menyediakan layanan terpisah yang berbeda
2. *Shared Resource* : server dapat melayani beberapa client pada saat yang sama dan mengatur pengaksesan resource
3. *Asymmetrical Protocol* : antara client dan server merupakan hubungan one to many
4. *Mix and Match* : tidak tergantung pada platform
5. *Encapsulation of Service* : message memberitahu server apa yang akan dikerjakan
6. *Transparency Location* : proses server dapat ditempatkan pada mesin yang sama atau terpisah dengan proses client. Client/server akan menyembunyikan lokasi server dari client
7. *Message-based-exchange* : antara client dan server berkomunikasi dengan mekanisme pertukaran message
8. *Scalability* : sistem client/server dapat dimekarkan baik vertikal maupun horisontal
9. *Integrity* : kode dan data server diatur secara terpusat, sedangkan pada client tetap pada komputer sendiri

2.2.2 Multiple Server Model



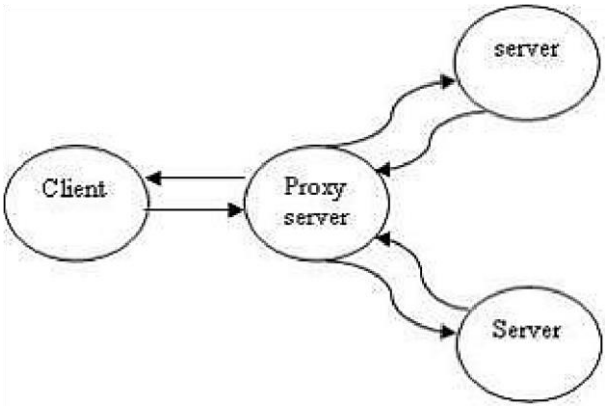
- Service disediakan oleh beberapa server

- Contoh: – Sebuah situs yang jalankan dibeberapa server
- Server menggunakan replikasi atau database terdistribusi

2.2.3 Proxy Server Model

Proxy server menyediakan hasil copy (replikasi) dari resource yang diatur oleh server lain. Biasanya proxy server dipakai untuk menyimpan hasil copy web resources. ketika client melakukan request ke server, hal pertama yang dilakukan adalah memeriksa proxy server apakah yang diminta oleh client terdapat pada proxy server.

Proxy server dapat diletakan pada setiap client atau dapat dipakai bersama oleh beberapa client. Tujuannya adalah meningkatkan performance dan availibity dengan mencagah frekuensi akses server.



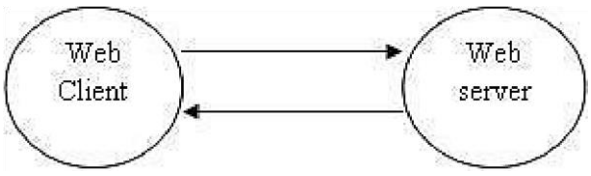
Proxy server membuat duplikasi beberapa server yang diakses oleh client.

Caching :

- penyimpanan lokal untuk item yang sering diakses
- meningkatkan kinerja
- mengurangi beban pada server
- wajib digunakan pada search engine

Contoh model proxy server : searching satu topik namun dilakukan dua kali maka searching terakhir memiliki waktu yang lebuh kecil.

2.2.4 Mobile Code Model

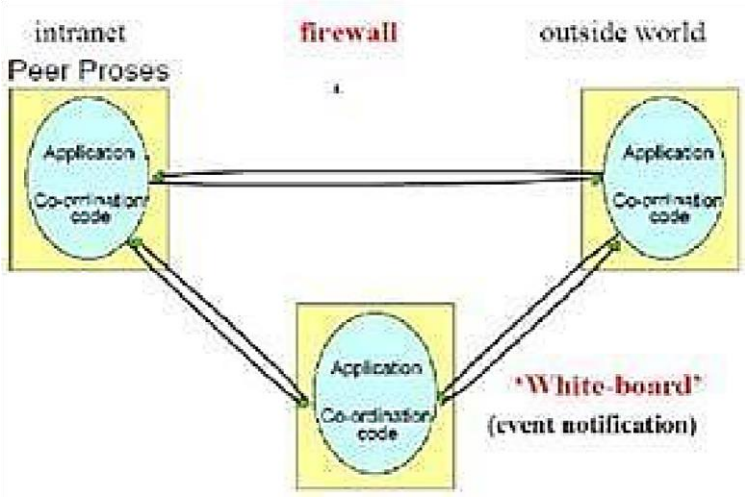


Kode yang berpindah dan dijalankan pada komputer yang berbeda. Contoh : Applet.

2.2.5 Mobile Agent Model

Mobile agent adalah sebuah program yang berpindah (termasuk data dan kode) dari satu komputer ke lainnya dalam jaringan. Biasanya melakukan suatu pekerjaan otomatis tertentu. Beberapa masalahnya antara lain authentication, permission dan keamanan. Alternatif dengan mengambil informasi melalui remote invocation. Contoh : digunakan untuk install dan memelihara software pada komputer dalam suatu organisasi, membandingkan harga produk dari beberapa vendor.

2.2.6 Peer Processes (Peer To Peer)



Bagian dari model sistem terdistribusi dimana sistem dapat sekaligus berfungsi sebagai client maupun server. Sebuah arsitektur di mana tidak terdapat mesin khusus yang melayani suatu pelayanan tertentu atau mengatur sumber daya dalam jaringan dan semua kewajiban dibagi rata ke seluruh mesin, yang dikenal sebagai peer. Pola komunikasi yang digunakan berdasarkan aplikasi yang digunakan. Peer-to-peer merupakan model yang paling general dan fleksible.

2.3 Model Interaksi (Interaction Models)

Untuk interaksinya sistem terdistribusi dibagi menjadi 2 bagian :

a. Synchronous Distributed System

Batas atas dan bawah waktu pengeksekusian dapat di set. Pesan yang dikirim diterima dalam waktu yang sudah ditentukan. Fluktuasi ukuran antara waktu local berada dalam suatu batasan.

Bererapa hal yang penting untuk diperhatikan :

1) Dalam synchronous distributed system terdapat satu waktu global
2) Hanya synchronous distributed system dapat memprediksi perilaku (waktu)

3) Dalam synchronous distributed system dimungkinkan dan aman untuk menggunakan mekanisme timeout dalam mendeteksi error atau kegagalan dalam proses atau komunikasi.

b. Asynchronous Distributed System

Banyak sistem terdistribusi yang menggunakan model interaksi ini (termasuk internet). Tidak ada batasan dalam waktu pengeksekusian, tidak ada batasan dalam delay transmission (penundaan pengiriman), dan tidak ada batasan terhadap fuktuasi waktu lokal. Asynchronous Distributed System secara praktek lebih banyak digunakan.

2.4 Model Kegagalan (Failure Models)

Kegagalan dapat terjadi pada proses atau kanal komunikasi, dan penyebabnya bisa berasal dari hardware ataupun software. Model kegagalan (failure models) dibutuhkan dalam membangun suatu sistem dengan prediksi terhadap kegagalankegagalan yang mungkin terjadi. Kegagalan yang dapat terjadi dan efek yang ditimbulkan :

a. Ommision Failures

Yang dimaksud dengan ommision failures adalah ketika prosesor dan kanal komunikasi mengalami kegagalan untuk melakukan hal yang seharusnya dilakukan. Dikatakan tidak mempunyai ommision failure apabila :

1) terjadi keterlambatan (delayed) tetapi akhirnya tetap tereksekusi
2) sebuah aksi dieksekusi walaupun terdapat kesalahan pada hasil Dengan synchronous system, ommision failures dapat dideteksi dengan timeouts. Kalau kita yakin bahwa pesan yang dikirim sampai, timeout akan mengindikasikan bahwa proses pengiriman rusak, seperti fail-stop behavior pada sistem.

b. Arbitary Failures

Ini adalah kegagalan yang paling buruk dalam sistem. Tahapan proses atau komunikasi diabaikan atau yang tidak diharapkan terjadi dieksekusi sehingga hasil yang diharapkan tidak terjadi atau mengeluarkan hasil yang salah.

c. Timing Failures

Timing Failures dapat terjadi pada synchronous system, dimana batas waktu diatur untuk eksekusi proses, komunikasi dan fuktuasi waktu. Timing Failures terjadi apabila waktu yang telah ditentukan terlampaui

NETWORKING (JARINGAN)

3.1 Pengertian Jaringan Komputer

Jaringan (network) adalah sebuah sistem operasi yang terdiri atas sejumlah komputer dan perangkat jaringan lainnya yang bekerja bersama-sama untuk mencapai suatu tujuam yang sama atau suatu jaringan kerja yang terdiri dari titiktitik (nodes) yang terhubung satu sama lain, dengan atau tanpa kabel. Masingmasing nodes berfungsi sebagai stasiun kerja (workstations). Salah satu nodes sebagai media jasa atau server, yaitu yang mengatur fungsi tertentu dari nodes lainnya. Pada dasarnya teknologi jaringan komputer itu sendiri merupakan perpaduan anatara tenologi komputer dan juga teknologi komunikasi.

3.1.1 Jaringan dibagi menjadi 2 yaitu :

a. Standalone
b. Network

3.1.2 Tujuan dari jaringan komputer:

a. Jaringan memungkinkan manajemen sumber daya lebih efisien: Misalnya, banyak pengguna dapat saling berbagi printer tunggal dengan kualitas tinggi, dibandingkan memakai printer kualitas rendah di masingmasing meja kerja. Selain itu, lisensi perangkat lunak jaringan dapat lebih murah dibandingkan lisensi stand-alone terpisah untuk jumlah pengguna sama, serta berbagi pemakaian CPU, Memori, dan Harddsik.

b. Jaringan membantu mempertahankan informasi agar tetap andal dan up-to-date: Sistem penyimpanan data terpusat yang dikelola dengan baik memungkinkan banyak pengguna mengaskses data dari berbagai lokasi yang berbeda, dan membatasi akses ke data sewaktu sedang diproses.

c. Jaringan membantu mempercepat proses berbagi data (data sharing). Transfer data pada jaringan selalu lebih cepat dibandingkan sarana berbagi data lainnya yang bukan jaringan.

d. Jaringan memungkinkan kelompok-kerja berkomunikasi dengan lebih efisien. Surat dan penyampaian pesan elektronik merupakan substansi sebagian besar sistem jaringan, disamping sistem penjadwalan, pemantauan proyek, konferensi online dan groupware, dimana semuanya membantu team bekerja lebih produktif.

e. Jaringan membantu usaha dalam melayani klien mereka secara lebih efektif. Akses jarak-jauh ke data terpusat memungkinkan karyawan dapat melayani klien di lapangan dan klien dapat langsung berkomunikasi dengan pemasok.

Agar dapat mencapai tujuan yang sama, setiap bagian dari jaringan komputer meminta dan memberikan layanaan (service). Pihak yang meminta layanan disebut klien (client) dan yang memberikan layanan disebut pelayanan (server). Arsitektur ini disebut dengan sistem clientserver, dan digunakan pada hampir seluruh aplikasi jaringan komputer.

3.1.3 Keuntungan Jaringan

Keuntungan utama yang langsung terasa dari network sharing itu adalah, Internet yang mendunia, karena pada hakikatnya Internet itu sendiri adalah serangkaian komputer (ribuan bahkan jutaan komputer) yang saling terhubung satu sama lain. Berevelusi dan berkembang dari waktu ke waktu, sehingga membentuk satu jaringan kompleks seperti yang kita rasakan sekarang ini.

Keuntungan lain dilihat dari sisi internal network adalah :

1. Resource Sharing, dapat menggunakan sumberdaya yang ada secara bersama-sama. Misal seorang pengguna yang berada 100 km jauhnya dari suatu data, tidak mendapatkan kesulitan dalam menggunakan data tersebut, seolah-olah data tersebut berada didekatnya. Hal ini sering diartikan bahwa jaringan komputer mangatasi masalah jarak.

2. Reliabilitas tinggi, dengan jaringan komputer kita akan mendapatkan reliabilitas yang tinggi dengan memiliki sumber-sumber alternatif persediaan. Misalnya, semua file dapat disimpan atau dicopy ke dua, tiga atu lebih komputer yang terkoneksi kejaringan. Sehingga bila salah satu mesin rusak, maka salinan di mesin yang lain bisa digunakan.

3. Menghemat uang, Komputer berukutan kecil mempunyai rasio harga/kinerja yang lebih baik dibandingkan dengan komputer yang besar. Komputer besar seperti mainframe memiliki kecapatan kira-kira sepuluh kali lipat kecepatan komputer kecil/pribadi. Akan tetap, harga mainframe seribu kali lebih mahal dari komputer pribadi. Ketidakseimbangan rasio harga/kinerja dan kecepatan inilah membuat para perancang sistem untuk membangun sistem yang terdiri dari komputer-komputer pribadi.

4. Hardware sharing, Bagi pakai hardware secara bersama-sama. Dengan adanya fasilitas jaringan kemudian menggunakan alat yang bernama printer server. maka sebuah printer laser berwarna yang mahal sekali harganya dapat dipakai secara bersama-sama oleh 10 orang pegawai. Begitu pula halnya dengan scanner, Plotter, dan alat-alat lainnya.

5. Keamanan dan pengaturan data, komputer dalam sebuah lingkungan bisnis, dengan adanya jaringan tersebut memungkinkan seorang administrator untuk mengorganisasi data-data kantor yang paling penting. Dari pada setiap departemen menjadi terpisah-

pisah dan data-datanya tercecer dimana-mana. Data penting tersebut dapat di manage dalam sebuah server back end untuk kemudian di replikasi atau dibackup sesuai kebijakan perusahaan. Begitu pula seorang admin akan dapat mengontrol data-data penting tersebut agar dapat diakses atau di edit oleh orang-orang yang berhak saja.

6. Ke-stabilan dan Peningkatan performa komputasi, Dalam kondisi tertentu, sebuah jaringan dapat digunakan untuk meningkatkan performa keseluruhan dari aplikasi bisnis, dengan cara penugasan komputasi yang di distribusikan kepada beberapa komputer yang ada dalam jaringan.

3.1.4 Kerugian Jaringan

Berbagai keuntungan dari media-media jaringan telah panjang lebar dijelaskan diatas, akan tetapi kerugian belum disinggung sama sekali. Jaringan dengan berbagai keunggulannya memang sangat membantu sekali kerja dalam suatu perusahaan. Tetapi kerugiannya juga banyak apabila tidak di sadari dari awal. Berikut beberapa kerugian dari implementasi jaringan :

1. Biaya yang tinggi kemudian semakin tinggi lagi. pembangunan jaringan meliputi berbagai aspek: pembelian hardware, software, biaya untuk konsultasi perencanaan jaringan, kemudian biaya untuk jasa pembangunan jaringan itu sendiri. Infestasi yang tinggi ini tentunya untuk perusahaan yang besar dengan kebutuhan akan jaringan yang tinggi. Sedangkan untuk pengguna rumahan biaya ini relatif kecil dan dapat ditekan. Tetapi dari awal juga network harus dirancang sedemikian rupa sehingga tidak ada biaya overhead yang semakin membengkak karena misi untuk pemenuhan kebutuhan akan jaringan komputer ini.

2. Manajemen Perangkat keras Dan Administrasi sistem : Di suatu organisasi perusahaan yang telah memiliki sistem, administrasi ini dirasakan merupakan hal yang kecil, paling tidak apabila dibandingkan dengan besarnya biaya pekerjaan dan biaya yang dikeluarkan pada tahap implementasi. Akan tetapi hal ini merupakan tahapan yang paling penting. Karena Kesalahan pada point ini dapat mengakibatkan peninjauan ulang bahkan konstruksi ulang jaringan. Manajemen pemeliharaan ini bersifat berkelanjutan dan memerlukan seorang IT profesional, yang telah mengerti benar akan tugasnya. Atau paling tidak telah mengikuti training dan pelatihan jaringan yang bersifat khusus untuk kebutuhan kantornya.

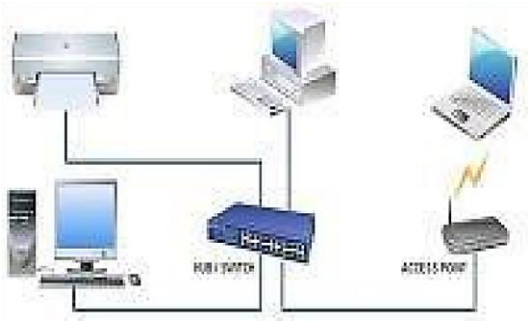
3. Sharing file yang tidak diinginkan : With the good comes the bad, ini selalu merupakan hal yang umum berlaku (ambigu), kemudahan sharing file dalam jaringan yang ditujukan untuk dipakai oleh orang-orang tertentu, seringkali mengakibatkan bocornya sharing folder dan dapat dibaca pula oleh orang lain yang tidak berhak. Hal ini akan selalu terjadi apabila tidak diatur oleh administrator jaringan.

4. Aplikasi virus dan metode hacking : hal-hal ini selalu menjadi momok yang menakutkan bagi semua orang, mengakibatkan network down dan berhentinya pekerjaan. Permasalahan ini bersifat klasik karena system yang direncanakan secara tidak baik. Masalah ini akan dijelaskan lebih lanjut dalam bab keamanan jaringan.

5. Berikut grafik yang menjelaskan prosentase penggunaan berbagai tipe media yang mendukung jaringan komputer beserta keuntungan dan kerugian dari masing-masing media / backbone tersebut.

3.2 Jenis – Jenis Jaringan Berdasarkan Jangkauan

1. Local Area Networking (LAN)



Local Area Network biasa disingkat **LAN** adalah jaringan komputer yang jaringannya hanya mencakup wilayah kecil; seperti jaringan komputer kampus, gedung, kantor, dalam rumah, sekolah atau yang lebih kecil. Saat ini, kebanyakan LAN berbasis pada teknologi IEEE 802.3 Ethernet menggunakan perangkat switch, yang mempunyai kecepatan transfer data 10, 100, atau 1000 Mbit/s. Selain teknologi Ethernet, saat ini teknologi 802.11b (atau biasa disebut *Wi-fi*) juga sering digunakan untuk membentuk LAN. Tempat-tempat yang menyediakan koneksi LAN dengan teknologi *Wi-fi* biasa disebut *hotspot*.
Pada sebuah LAN, setiap node atau komputer mempunyai daya komputasi sendiri, berbeda dengan konsep *dump terminal*. Setiap komputer juga dapat mengakses sumber daya yang ada di LAN sesuai dengan hak akses yang telah diatur. Sumber daya tersebut dapat berupa data atau perangkat seperti

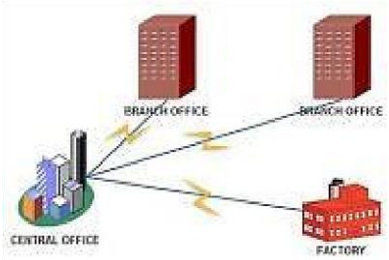
printer. Pada LAN, seorang pengguna juga dapat berkomunikasi dengan pengguna yang lain dengan menggunakan aplikasi yang sesuai.

Berbeda dengan Jaringan Area Luas atau Wide Area Network (WAN), maka LAN mempunyai karakteristik sebagai berikut :

a) Mempunyai pesat data yang lebih tinggi
b) Meliputi wilayah geografi yang lebih sempit
c) Tidak membutuhkan jalur telekomunikasi yang disewa dari operator telekomunikasi

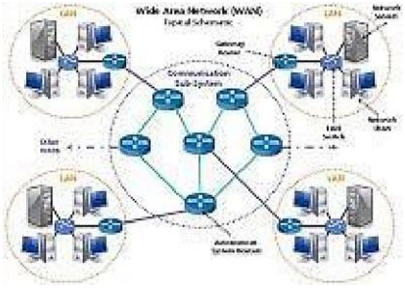
Biasanya salah satu komputer di antara jaringan komputer itu akan digunakan menjadi server yang mengatur semua sistem di dalam jaringan tersebut.

2. Metropolitan Area Networking (MAN)



Yaitu Jaringan yang lebih luas dari LAN, MAN biasanya meliputi area yang lebih besar seperti area propinsi, antar gedung. Mengapa MAN itu dikatakan lebih luas dari LAN?, Yah, karena jaringan MAN itu terhubung dari beberapa jaringan LAN yang dihubungkan melalui switch lagi.

3. Wide Area Networking (WAN)



Yaitu Jaringan yang lingkupnya biasanya sudah menggunakan sarana Satelit ataupun kabel bawah laut sebagai contoh keseluruhan jaringan BANK BNI yang ada di Indonesia ataupun yang ada di Negara-negara lain. Menggunakan sarana WAN, Sebuah Bank yang ada di Bandung bisa menghubungi kantor cabangnya yang ada di Hongkong, hanya dalam beberapa menit. Biasanya WAN agak rumit dan sangat

kompleks, menggunakan banyak sarana untuk menghubungkan antara LAN dan WAN ke dalam Komunikasi Global seperti Internet.

4. Topologi Jaringan

Topologi jaringan adalah, hal yang menjelaskan hubungan geometris antara unsur-unsur dasar penyusun jaringan, yaitu node, link, dan station.

a. Topologi Bus

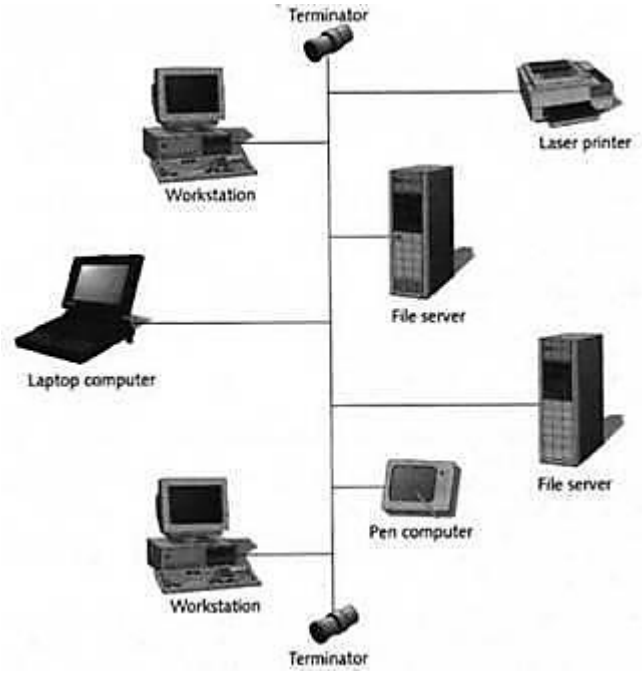
Pada topologi Bus, kedua unjung jaringan harus diakhiri dengan sebuah terminator. Barel connector dapat digunakan untuk memperluasnya. Jaringan hanya terdiri dari satu saluran kabel yang menggunakan kabel BNC. Komputer yang ingin terhubung ke jaringan dapat mengkaitkan dirinya dengan mentap Ethernetnya sepanjang kabel.

Linear Bus: Layout ini termasuk layout yang umum. Satu kabel utama menghubungkan tiap simpul, ke saluran tunggal komputer yang mengaksesnya ujung dengan ujung. Masing-masing simpul dihubungkan ke dua simpul lainnya, kecuali mesin di salah satu ujung kabel, yang masing-masing hanya terhubung ke satu simpul lainnya. Topologi ini seringkali dijumpai pada sistem client/server, dimana salah satu mesin pada jaringan tersebut difungsikan sebagai File Server, yang berarti bahwa mesin tersebut dikhususkan hanya untuk pendistribusian data dan biasanya tidak digunakan untuk pemrosesan informasi.

Instalasi jaringan Bus sangat sederhana, murah dan maksimal terdiri atas 5-7 komputer. Kesulitan yang sering dihadapi adalah kemungkinan terjadinya tabrakan data karena mekanisme jaringan relatif sederhana dan jika salah satu node putus maka akan mengganggu kinerja dan trafik seluruh jaringan.

- Keunggulan topologi Bus adalah pengembangan jaringan atau penambahan workstation baru dapat dilakukan dengan mudah tanpa mengganggu workstation lain.
- *Kelemahan dari topologi ini adalah bila terdapat gangguan di sepanjang kabel pusat maka keseluruhan jaringan akan mengalami gangguan. Topologi linear bus merupakan topologi yang banyak dipergunakan pada masa penggunaan kabel Coaxial menjamur. Dengan menggunakan T-Connector (dengan terminator 50ohm pada ujung network), maka komputer atau perangkat jaringan lainnya bisa dengan mudah dihubungkan satu sama lain. Kesulitan utama dari penggunaan kabel coaxial adalah sulit untuk mengukur apakah kabel coaxial yang dipergunakan benarbenar matching atau tidak. Karena kalau tidak sungguhsungguh diukur secara benar akan merusak NIC (network interface card) yang dipergunakan dan kinerja jaringan menjadi terhambat, tidak mencapai kemampuan maksimalnya. Topologi ini juga sering digunakan pada jaringan dengan basis fiber optic

(yang kemudian digabungkan dengan topologi star untuk menghubungkan dengan client atau node.).



b. Topologi Star (Bintang)

Topologi bintang merupakan bentuk topologi jaringan yang berupa konvergensi dari node tengah ke setiap node atau pengguna. Topologi jaringan bintang termasuk topologi jaringan dengan biaya menengah.

Kelebihan

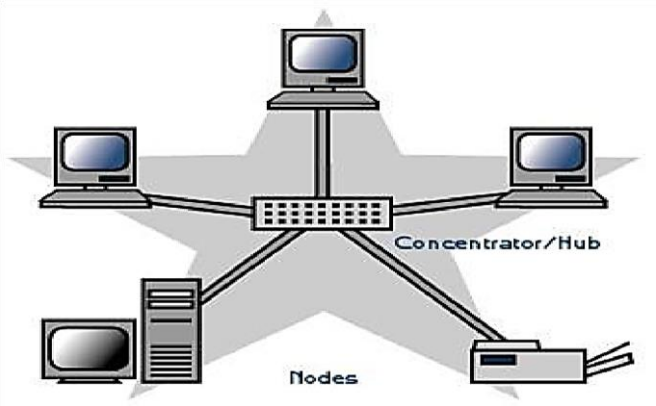
- Kerusakan pada satu saluran hanya akan mempengaruhi jaringan pada saluran tersebut dan station yang terpaut.
- Tingkat keamanan termasuk tinggi.
- Tahan terhadap lalu lintas jaringan yang sibuk.
- Penambahan dan pengurangan station dapat dilakukan dengan mudah.

Kekurangan

- Jika node tengah mengalami kerusakan, maka seluruh jaringan akan terhenti.

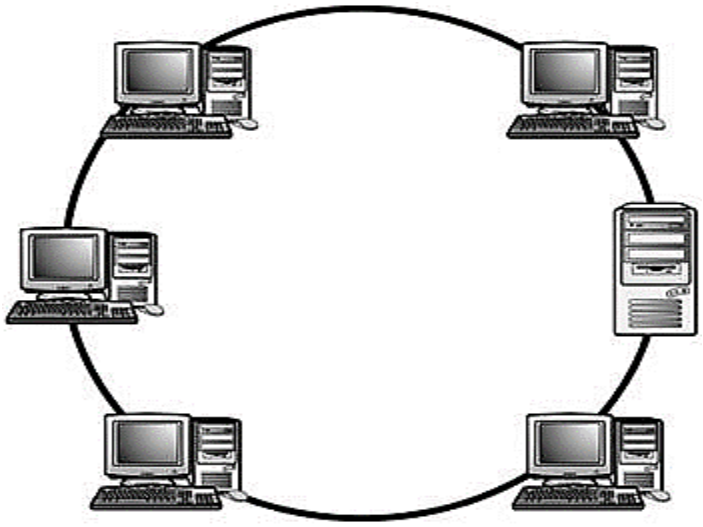
Penanganan

Perlunya disiapkan node tengah cadangan.



c. Topologi Ring

Topologi cincin adalah topologi jaringan berbentuk rangkaian titik yang masingmasing terhubung ke dua titik lainnya, sedemikian sehingga membentuk jalur melingkar membentuk cincin. Pada topologi cincin, komunikasi data dapat terganggu jika satu titik mengalami gangguan. Jaringan FDDI mengantisipasi kelemahan ini dengan mengirim data searah jarum jam dan berlawanan dengan arah jarum jam secara bersamaan.



3.3 PROTOKOL JARINGAN

Protokol adalah sebuah aturan atau standar yang mengatur atau mengijinkan terjadinya hubungan, komunikasi, dan perpindahan data antara dua atau lebih titik komputer. Protokol dapat diterapkan pada perangkat keras, perangkat lunak atau kombinasi dari keduanya. Pada tingkatan yang terendah, protokol mendefinisikan koneksi perangkat keras.

Protokol perlu diutamakan pada penggunaan standar teknis, untuk menspesifikasi bagaimana membangun komputer atau menghubungkan peralatan perangkat keras. Protokol secara umum digunakan pada komunikasi *real-time* dimana standar digunakan untuk mengatur struktur dari informasi untuk penyimpanan jangka panjang.

Sangat susah untuk *menggeneralisir* protokol dikarenakan protokol memiliki banyak variasi didalam tujuan penggunaanya. Kebanyakan protokol memiliki salah satu atau beberapa dari hal berikut:

- Melakukan deteksi adanya koneksi fisik atau ada tidaknya komputer atau mesin lainnya.
- Melakukan metode "jabat-tangan" (*handshaking*).
- Negosiasi berbagai macam karakteristik hubungan.
- Bagaimana mengawali dan mengakhiri suatu pesan.
- Bagaimana format pesan yang digunakan.
- Yang harus dilakukan saat terjadi kerusakan pesan atau pesan yang tidak sempurna.
- Mendeteksi *rugi-rugi* pada hubungan jaringan dan langkah-langkah yang dilakukan selanjutnya
- Mengakhiri suatu koneksi.

Untuk memudahkan memahami Protokol, kita mesti mengerti Model OSI. Dalam Model OSI terdapat 7 layer dimana masing-masing layer mempunyai jenis protokol sesuai dengan peruntukannya.

a) Protokol Komunikasi Antar peralatan Jaringan protokol ini mampu mengatur bentuk dan jenis data yang dikirim, menentukan besaran listrik yang digunakan, jenis, dan banyaknya kabel yang digunakan untuk proses transmisi.
b) Protokol dari SIstem OPerasi yang DIgunakan
   Sistem operasi NetWare menggunakan protokol utama IPX/SPX, MIcrosoft menggunakan protokol NetBeui, sedangkan protokol standar pada internet menggunakan protokol TCP/IP.

3.4 CARA KERJA JARINGAN

a. Jaringan Client-Server

Terdiri atas server yang biasanya menggunakan sistem windows 2000 server.

b. Jaringan Peer to Peer

Setiap komputer dapat membuat account user dan berbagai sumber.contohnya windows 98,windows NT workstation dan windows 2000 profesional.

3.5 Cara Kerja Jaringan Telepon Kabel

Telepon kabel menggunakan sistem wireline. Sehingga membutuhkan kabel supaya dapat berfungsi. Cara kerja telepon kabel antara lain:

a. Suara dari pengirim diterima oleh alat yang disebut microphone.
b. Microphone mengubah gelombang suara menjadi sinyal listrik dan kemudian disalurkan oleh perangkat telepon.
c. Sinyal tersebut disalurkan melalui kabel ke pusat telekomunikasi.
d. Dari pusat telekomunikasi, sinyal tersebut diteruskan kepada penerima.
e. Setelah sampai ke penerima, maka sinyal tersebut diubah lagi menjadi gelombang suara oleh alat yang disebut speaker.

3.6 Cara Kerja Jaringan Telepon Seluler

Telepon seluler menggunakan sistem wireless. Pengirim dan penerima harus tercakup BTS(Base Transcaiver Station). BTS adalah peralatan yang memfasilitasi komunikasi secara wireless antara pengguna telepon seluler. Cara kerja telepon Wireless antara lain:

a. Suara dari pengirim diterima oleh alat yang disebut microphone.
b. Microphone mengubah gelombang suara menjadi sinyal listrik dan kemudian dipancarkan oleh ponsel ke BTS terdekat.
c. Sinyal tersebut diterima oleh BTS dan sinyal tersebut diteruskan ke pusat telekomunikasi.
d. Dari pusat telekomunikasi sinyal diteruskan kepada BTS terdekat kemudian diteruskan ke si penerima.
e. Setelah sampai ke penerima, maka sinyal tersebut diubah lagi menjadi gelombang suara oleh alat yang disebut SPEAKER.

3.7 NETWORK LAYER

3.7.1 Pengertian Network Layer

Network Layer merupakan layer yang bertanggung jawab menentukan alamat jaringan, menentukan rute yang harus diambil selama perjalanan, dan menjaga antrian trafik di jaringan. Data pada layer ini berbentuk paket.

Fungsi utama dari layer tiga, yaitu layer Network adalah pada referensi model OSI untuk enable message untuk melewati antar jaringan local yang terhubung, yang biasanya lebih banyak jaringan lewat link

WAN. Piranti-piranti, protocol-protocol, dan program-program yang berjalan pada layer Network bertanggung jawab untuk mengidentifikasikan, memilah, dan mengarahkan traffic yang melalui antarjaringan.

Jaringan menjelaskan beberapa kumpulan dari piranti terhubung bersama-sama untuk berbagi informasi dan resources dan juga saling berkomunikasi. Secara fisik, jaringan-jaringan diidentifikasikan oleh segmen-segmen media transmisi dan juga oleh address-address jaringan.

3.7.2 Lapisan jaringan (network layer)

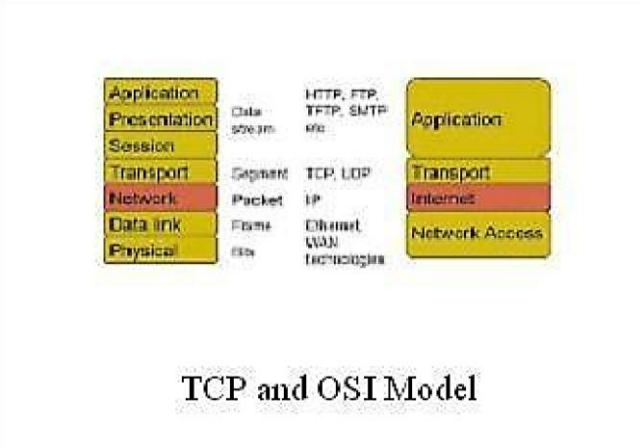
Network layer berfungsi untuk pengendalian operasi subnet. Masalah desain yang penting adalah bagaimana caranya menentukan route pengiriman paket dari sumber ke tujuannya. Route dapat didasarkan pada table statik yang “dihubungkan ke” network Route juga dapat ditentukan pada saat awal percakapan misalnya session terminal. Terakhir, route dapat juga sangat dinamik, dapat berbeda bagi setiap paketnya. Oleh karena itu, route pengiriman sebuah paket tergantung beban jaringan saat itu.

3.7.3 Fungsi dan Tugas Network Layer :

Network layer berfungsi untuk pengendalian operasi subnet. Masalah desain yang penting adalah bagaimana caranya menentukan route pengiriman paket dari sumber ke tujuannya. Route dapat didasarkan pada table statik yang “dihubungkan ke” network. Route juga dapat ditentukan pada saat awal percakapan misalnya session terminal.

Contoh penggunaan dari Lapisan Network Layer B-Router:

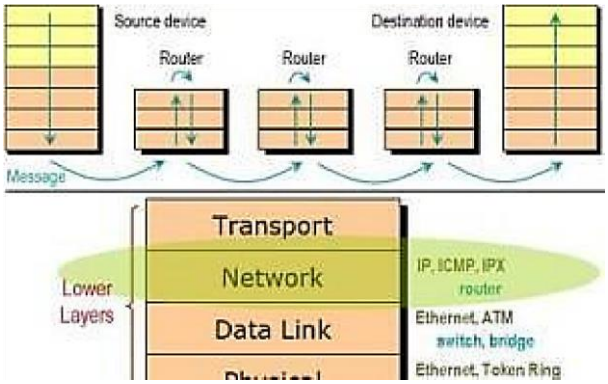
- Network components :
- Brouter
- Router
- Frame Relay Device
- ATM switch
- Advanced Cable Tester



TCP and OSI Model

Apabila kita melihat IP Address dalam TCP/IPModel dan OSI Model maka dapat dilihat seperti pada gambar disamping bahwa pada TCP model IP address masuk pada layer Network, pada OSI model IP address masuk pada Layer 3 yaitu

Internet.



Berfungsi untuk mendefinisikan alamat-alamat IP, membuat header untuk paket-paket, dan kemudian melakukan routing melalui internetworking dengan menggunakan router dan switch layer3.

Network, merupakan layer yang mendefinisikan akhir pengiriman paket data dimana komputer mengidentifikasi logical address seperti IP Addreses, bagaimana meneruskan/routing (oleh router) untuk siapa pengiriman paket data. Layer ini juga mendefinisikan fragmentasi dari sebuah paket dengan ukuran unit yang lebih kecil. Router adalah contoh yang tepat dari definisi layer ini.

3.7.4 Tugas Utama Network Layer

Tugas utama dari layer network adalah menyediakan fungsi routing sehingga paketdapat dikirim keluar dari segment network lokal ke suatu tujuan yang berada padasuatu network lain. IP, Internet Protocol,

umumnya digunakan untuk tugas ini. Protocollainnya seperti IPX, Internet Packet eXchange. Perusahaan Novell telah memprogram protokol menjadi beberapa, seperti SPX (Sequence Packet Exchange) & NCP (NetwareCore Protocol).

INTER-PROCESS COMMUNICATION

4.1 Mengenal Inter-Process Communication

Program aplikasi antarmuka untuk UDP menyediakan pesan lewat abstraksi bentuk sederhana dari komunikasi interprocess . Hal ini memungkinkan proses pengiriman untuk mengirimkan satu pesan ke proses penerimaan . Paket independen yang mengandung pesanpesan ini disebut datagrams . Di Java dan UNIX API , pengirim menentukan tujuan menggunakan soket referensi langsung ke port tertentu yang digunakan oleh proses tujuan pada komputer tujuan .

Program aplikasi antarmuka untuk TCP memberikan abstraksi dari dua arah pasang streambetween proses . Informasi yang dikomunikasikan terdiri dari aliran item data tanpa batas pesan . Streaming menyediakan sebuah blok bangunan untuk komunikasi produsen-konsumen . Sebuah produsen dan konsumen membentuk sepasang proses dimana peran yang pertama adalah untuk memproduksi barang-barang data dan peran yang kedua adalah untuk mengkonsumsinya . Item data yang dikirim oleh produsen ke konsumen yang antri pada saat kedatangan di host penerima sampai konsumen siap untuk menerima mereka . Konsumen harus menunggu ketika ada item data yang tersedia . Produser harus menunggu jika penyimpanan yang digunakan untuk menyimpan item data antri habis .

4.2 The API for the Internet protocols

4.2.1 The characteristics of interprocess communication

Message passing antara sepasang proses dapat didukung oleh dua operasi komunikasi pesan, Kirim dan menerima, didefinisikan dalam hal tujuan dan pesan. Untuk berkomunikasi, satu proses mengirimkan pesan (urutan byte) ke tujuan dan proses lain pada tujuan menerima pesan. Kegiatan ini melibatkan komunikasi data dari proses pengiriman ke proses penerimaan dan mungkin melibatkan sinkronisasi dua proses.

4.2.2 Synchronous and asynchronous communication

Antrian A dikaitkan dengan masing-masing tujuan pesan . Mengirim proses menyebabkan pesan yang akan ditambahkan ke antrian terpencil dan menerima proses menghapus pesan antrian fromlocal . Komunikasi antara proses pengiriman dan penerimaan dapat berupa sinkron atau asinkron . Dalam synchronousform komunikasi , proses pengiriman dan penerimaan sinkronisasi pada setiap pesan . Dalam hal ini , keduanya Kirim dan blockingoperations receiveare . Setiap kali sendis mengeluarkan

proses pengiriman (atau benang) diblokir sampai receiveis sesuai diterbitkan . Setiap kali receiveis dikeluarkan oleh proses (atau benang) , blok sampai pesan tiba.

Dalam asynchronousform komunikasi , penggunaan sendoperation adalah nonblocking dalam proses pengiriman diperbolehkan untuk melanjutkan segera setelah pesan telah disalin ke buffer lokal , dan transmisi pesan hasil secara paralel dengan proses pengiriman . Receiveoperation dapat memiliki blocking dan non blocking varian . Pada varian non -blocking , hasil proses penerimaan dengan program setelah mengeluarkan receiveoperation , yang menyediakan buffer untuk diisi di latar belakang , tetapi secara terpisah harus menerima pemberitahuan bahwa buffer telah diisi , dengan pemungutan suara atau mengganggu .

4.2.3 Message destinations

Sebuah port lokal adalah tujuan pesan dalam komputer , ditetapkan sebagai integer . Port A memiliki tepat satu receiver (port multicast adalah pengecualian) , tetapi dapat memiliki banyak pengirim . Proses dapat menggunakan beberapa port untuk menerima pesan . Setiap proses yang tahu jumlah port dapat mengirim pesan ke itu . Server umumnya mempublikasikan nomor port mereka untuk digunakan oleh klien . Jika klien menggunakan alamat Internet tetap untuk merujuk ke layanan , maka layanan yang harus selalu dijalankan pada komputer yang sama untuk alamat untuk tetap berlaku . Hal ini dapat dihindari dengan menggunakan pendekatan berikut untuk memberikan transparansi lokasi : Program Client merujuk ke layanan dengan nama dan menggunakan nama server atau pengikat untuk menerjemahkan nama-nama mereka ke lokasi server yang saat runtime . Hal ini memungkinkan layanan yang akan direlokasi tetapi tidak untuk bermigrasi yaitu, untuk dipindahkan sementara sistem berjalan

4.2.4 Reliability

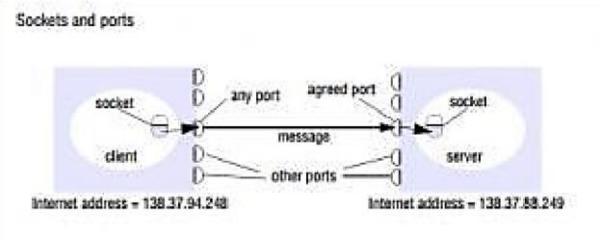
Sejauh properti validitas yang bersangkutan, layanan pesan point-to-point dapat digambarkan sebagai diandalkan jika pesan dijamin akan disampaikan meskipun sejumlah „wajar" dari paket yang terjatuh atau hilang. Sebaliknya, layanan pesan point-topoint dapat digambarkan sebagai tidak bisa diandalkan jika pesan tidak dijamin akan disampaikan dalam menghadapi bahkan satu paket terjatuh atau hilang. Untuk integritas, pesan harus tiba tidak rusak dan tanpa duplikasi.

4.2.5 Ordering

Beberapa aplikasi memerlukan bahwa pesan disampaikan agar pengirim yaitu, urutan di mana mereka dikirim oleh pengirim. Penyampaian pesan dari pengirim agar dianggap sebagai kegagalan oleh aplikasi tersebut.

4.2.6 Sockets

Kedua bentuk komunikasi (UDP dan TCP) menggunakan soket abstraksi, yang menyediakan titik akhir untuk komunikasi antara proses. Sockets berasal dari BSD UNIX tetapi juga hadir di sebagian besar versi lain dari UNIX, termasuk Linux serta Windows dan Macintosh OS. Interprocess komunikasi terdiri dari transmisi pesan antara soket dalam satu proses dan soket dalam proses lain, seperti yang diilustrasikan pada Gambar berikut.



Untuk proses untuk menerima pesan, soket harus terikat ke port lokal dan salah satu alamat internet dari komputer yang berjalan. Pesan yang dikirim ke alamat Internet tertentu dan nomor port dapat diterima hanya oleh proses yang socket dikaitkan dengan alamat Internet dan nomor port. Proses dapat menggunakan soket yang sama untuk mengirim dan menerima pesan. Setiap komputer memiliki sejumlah besar (2^{16}) nomor port yang mungkin untuk digunakan oleh proses lokal untuk menerima pesan.

Java API untuk alamat Internet, Seperti paket IP yang mendasari UDP dan TCP dikirim ke alamat Internet, Java menyediakan kelas, InetAddress, yang mewakili alamat Internet. Pengguna kelas ini mengacu pada komputer dengan Domain Name System (DNS) nama host Misalnya, contoh InetAddressthat berisi alamat Internet dapat dibuat dengan memanggil metode statis InetAddress, memberikan hostname DNS sebagai argumen. Metode menggunakan DNS untuk mendapatkan alamat Internet yang sesuai. Misalnya, untuk mendapatkan obyek yang mewakili alamat internet dari host yang namanya DNS adalah bruno.dcs.qmul.ac.uk, gunakan: InetAddress aComputer = InetAddress.getByName (“bruno.dcs.qmul.ac.uk”); Metode ini dapat melempar Unknown Host Exception. Perhatikan bahwa pengguna kelas tidak perlu menyatakan nilai eksplisit dari alamat Internet. Bahkan, kelas merangkum rincian representasi alamat Internet. Jadi antarmuka untuk kelas ini tidak

tergantung pada jumlah byte yang diperlukan untuk mewakili alamat Internet – 4 byte di IPv4 dan 16 byte di IPv6.

4.2.7 UDP datagram communication

Sebuah datagram yang dikirim oleh UDP ditularkan dari proses pengiriman ke proses penerimaan tanpa pengakuan atau retries. Jika terjadi kegagalan, pesan mungkin tidak akan tiba. Sebuah datagram ditransmisikan antara proses ketika salah satu proses sendsit dan receivesit lain. Berikut ini adalah beberapa masalah yang berkaitan dengan komunikasi datagram:

- **Message Size** : Proses penerimaan perlu menentukan sebuah array byte dari ukuran tertentu di mana untuk menerima pesan . Jika pesan terlalu besar untuk array , itu dipotong pada saat kedatangan . Protokol IP yang mendasari memungkinkan paket panjang hingga 2^{16} byte , yang mencakup header serta pesan. Namun, kebanyakan lingkungan memberlakukan pembatasan ukuran 8 kilobyte
- **Blocking** : Sockets biasanya menyediakan non – blocking dan blocking mengirimkan menerima untuk komunikasi datagram (non – blocking receiveis pilihan dalam beberapa implementasi) . Kembali sendoperation ketika telah menyerahkan pesan ke UDP dan IP protokol yang mendasari , yang bertanggung jawab untuk transmisi ke tujuan
- **Timeout:** Blok menerima bahwa selamanya cocok untuk digunakan oleh server yang menunggu untuk menerima permintaan dari klien. Namun dalam beberapa program, tidak tepat bahwa proses yang telah dipanggil receiveoperation harus menunggu selamanya dalam situasi di mana proses pengiriman mungkin telah jatuh atau pesan yang diharapkan mungkin telah hilang.
- **Receive from any:** receivemethod tidak menentukan asal untuk pesan. Sebaliknya, sebuah doa dari receivegets pesan yang ditujukan kepada soketnya dari asal manapun. Receivemethod mengembalikan alamat Internet dan port lokal pengirim, yang memungkinkan penerima untuk memeriksa di mana pesan itu berasal. Hal ini dimungkinkan untuk menghubungkan soket datagram ke port Internet dan alamat remote tertentu, dalam hal ini soket hanya mampu mengirim pesan dan menerima pesan dari alamat tersebut.

4.2.8 Failure model for UDP datagrams

Datagrams UDP menderita kegagalan berikut:

Kegagalan Kelalaian: Pesan bisa turun kadang-kadang, baik karena checksum error atau karena tidak ada ruang buffer yang tersedia di sumber atau tujuan.

Untuk menyederhanakan diskusi, kita menganggap kegagalan kirim-kelalaian dan menerima-kelalaian sebagai omissionfailures dalam saluran komunikasi. Pengurutan: Pesan kadang-kadang dapat disampaikan rusak pengirim. Aplikasi menggunakan datagram UDP yang tersisa untuk memberikan cek mereka sendiri untuk mencapai kualitas komunikasi yang handal yang mereka butuhkan. Sebuah layanan pengiriman yang handal dapat dibangun dari satu yang menderita dari kegagalan pembiaran oleh penggunaan ucapan terima kasih.

4.2.9 Use of UDP

Untuk beberapa aplikasi, dapat diterima untuk menggunakan layanan yang bertanggung jawab terhadap kegagalan kelalaian sesekali. Sebagai contoh, Domain Name System, yang terlihat nama-nama DNS di Internet, diimplementasikan atas UDP. Voice over IP (VOIP) juga berjalan di atas UDP. Datagrams UDP kadang-kadang pilihan yang menarik karena mereka tidak menderita overhead yang terkait dengan pengiriman pesan dijamin.

Ada tiga sumber utama overhead:

- kebutuhan untuk menyimpan informasi negara pada sumber dan tujuan;
- transmisi pesan-pesan tambahan;
- latency untuk pengirim.

4.2.10 Java API for UDP datagrams

The Java API menyediakan datagram komunikasi melalui dua kelas : Datagram Packet dan Datagram Socket . DatagramPacket : Class ini menyediakan konstruktor yang membuat sebuah instance dari sebuah array byte yang terdiri dari pesan , panjang pesan dan alamat Internet dan nomor port lokal dari soket tujuan . Class ini menyediakan konstruktor lain untuk digunakan ketika menerima pesan . Argumen menentukan sebuah array byte di mana untuk menerima pesan dan panjang dari array . Sebuah pesan yang diterima diletakkan di DatagramPackettogether dengan panjang dan alamat Internet dan port soket pengirim . Pesan dapat diambil dari sarana DatagramPacketby metode getData . Metode getPort dan getAddressaccess port dan alamat Internet .

```
import java.net.*;
import java.io.*;
public class UDPClient{
   public static void main(String args[]){
      // args give message contents and server hostname
      DatagramSocket aSocket = null;
      try {
            aSocket = new DatagramSocket();
            byte [] m = args[0].getBytes();
            InetAddress aHost = InetAddress.getByName(args[1]);
            int serverPort = 6789;
            DatagramPacket request =
                  new DatagramPacket(m,  m.length(), aHost, serverPort);
            aSocket.send(request);
            byte[] buffer = new byte[1000];
            DatagramPacket reply = new DatagramPacket(buffer, buffer.length);
            aSocket.receive(reply);
            System.out.println("Reply: " + new String(reply.getData()));
      } catch (SocketException e){System.out.println("Socket: " + e.getMessage());
      } catch (IOException e){System.out.println("IO: " + e.getMessage());
      } finally { if(aSocket != null) aSocket.close();}
   }
}
```

Datagram Socket : Kelas ini mendukung soket untuk mengirim dan menerima datagrams UDP . Ini menyediakan konstruktor thattakes nomor port sebagai argumen , untuk digunakan oleh proses yang harus menggunakan port tertentu . Hal ini juga menyediakan konstruktor tanpa argumen yang memungkinkan sistem untuk memilih port lokal gratis . Konstruktor ini bisa melempar SocketExceptionif port yang dipilih sudah digunakan atau jika port direservasi (nomor di bawah 1024) yang ditentukan ketika berjalan di atas UNIX .

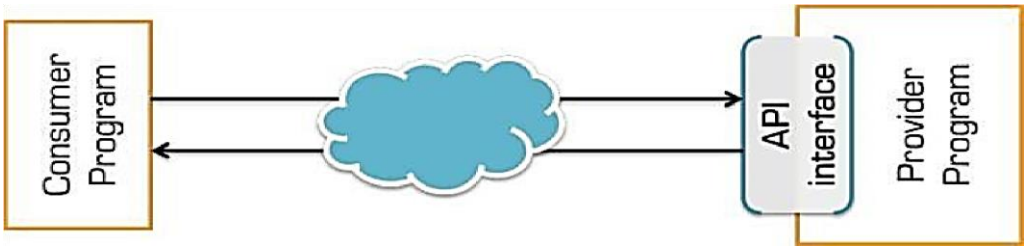
Kelas DatagramSocket menyediakan metode yang meliputi berikut ini:

- Kirim dan menerima: Metode ini untuk transmisi datagram antara sepasang soket. Argumen dari sendis contoh DatagramPacket mengandung pesan dan tujuan. Argumen dari receiveis sebuah DatagramPacketin kosong yang untuk menempatkan pesan, panjang dan asal-usulnya. Metode mengirim dan receivec sebuah IOExceptions melempar.
- setSoTimeout: Metode ini memungkinkan waktu untuk diatur. Dengan set timeout, menerima Metode akan memblokir untuk waktu tertentu dan kemudian melemparkan sebuah InterruptedIOException. Menghubungkan: Metode ini digunakan untuk menghubungkan ke remote port tertentu dan Alamat Internet, dalam hal ini soket hanya mampu mengirim pesan ke dan menerima pesan dari alamat tersebut

4.3 Pengertian API

API (*Application Programming Interface)* adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi lain. *Wikipedia*

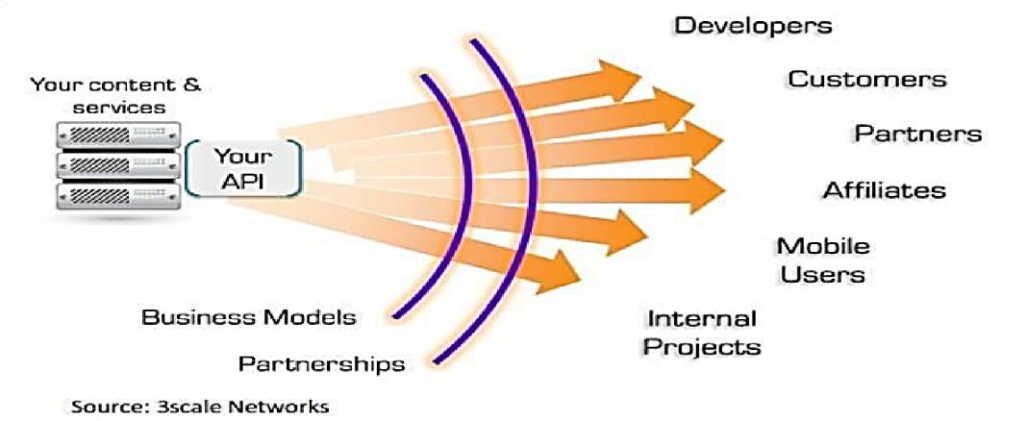
Dalam contoh sederhana, dibutuhkan setidaknya ribuan *system calls* per detik. Oleh karena itu Kebanyakan *programmer* membuat aplikasi dengan menggunakan *Application Programming Interface*(API). Dalam API itu terdapat fungsifungsi/perintah-perintah untuk menggantikan bahasa yang digunakan dalam *system calls* dengan bahasa yang lebih terstruktur dan mudah dimengerti oleh *programmer*. Fungsi yang dibuat dengan menggunakan API tersebut kemudian akan memanggil *system calls* sesuai dengan sistem operasinya. Tidak tertutup kemungkinan nama dari *system calls* sama dengan nama di API.



Gambaran cara kerja API

Keuntungan memprogram dengan menggunakan API adalah:

- **Portabilitas.** Programmer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah ter- *install* API tersebut. Sedangkan *system call*berbeda antar sistem operasi, dengan catatan dalam implementasinya mungkin saja berbeda.
- **Lebih Mudah Dimengerti.** API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal editing dan pengembangan.



API Network

System call interface ini berfungsi sebagai penghubung antara API dan *system call* yang dimengerti oleh sistem operasi. *System call interface* ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil *system calls* yang diperlukan.

Untuk membuka suatu *file* tersebut *user* menggunakan program yang telah dibuat dengan menggunakan bantuan API, maka perintah dari *user* tersebut diterjemahkan dulu oleh program menjadi perintah *open()*.

Perintah *open()* ini merupakan perintah dari API dan bukan perintah yang langsung dimengerti oleh kernel sistem operasi. Oleh karena itu, agar keinginan *user*dapat dimengerti oleh sistem operasi, maka perintah *open()* tadi diterjemahkan ke dalam bentuk *system call* oleh *system call interface*. Implementasi perintah *open()* tadi bisa bermacam-macam tergantung dari sistem operasi yang kita gunakan. Ada banyak penyedia layanan API, seperti contohnya Info Cuaca, kita bisa membuat aplikasi tentang cuaca yang selalu update, dan data yang di ambil dari API penyedia layanan. Berikut 3 penyedia layanan API info cuaca yang bisa digunakan untuk membuat wheater apps API.

Berinteraksi dengan API Speech di Windows Phone dalam "Windows Phone"

Mempelajari Lebih Jauh Tentang CloudKit di iOS Part II - FoodPin Apps dalam "iOS"

Local Storage di Windows Phone dalam "Windows Phone"

4.4 Komunikasi dalam Sistem Terdistribusi

Interproses komunikasi adalah jantung dari semua sistem terdistribusi. Tidak masuk akal untuk mempelajari sistem terdistribusi tanpa hati-hati dengan cara bahwa proses pada mesin yang berbeda

dapat saling bertukar informasi. Komunikasi dalam sistem terdistribusi selalu didasarkan pada pesan tingkat rendah yang lewat seperti yang ditawarkan oleh jaringan yang mendasarinya.

Sistem terdistribusi modern biasanya terdiri dari ribuan atau bahkan jutaan proses yang tersebar di seluruh jaringan internet.

Dalam pembahasan kali ini, kita mulai dengan membahas aturan bahwa proses berkomunikasi harus ada yang dikenal sebagai protokol, dan cocencrate. Pada penataan tersebut protokol dalam bentuk lapisan. Dari tampilan diempat luas dan model yang digunakan untuk komunikasi terbagi atas: prosedur panggilan jarak jauh (RMC), remote metode doa (RMI), pesan middleware berorientasi (MOM) dan streams.

Model pertama untuk komunikasi dalam sistem terdistribusi adalah panggilan prosedur remote (RPC). Sebuah RPC bertujuan menyembunyikan sebagian besar seluk-beluk pesan lewat, dan sangat ideal untuk aplikasi clientserver. perbaikan untuk model RPC datang dalam bentuk doa metode remote (RMIs), yang didasarkan pada gagasan obyek terdistribusi. RPC dan RMIs dibahas dalam bagian terpisah.

Pesan-berorientasi middleware (MOM) adalah disebut juga sebagai suatu *messagequeuing sistem*, suatu kerangka pesan, atau sekedar *messaging* sistem. MOM dapat membentuk suatu lapisan middleware yang penting untuk aplikasi perusahaan melalui Internet. MOM dapat menerbitkan dan mendaftar model, suatu klien dapat mendaftarkan sebagai penerbit atau seorang langganan dari pesan. Pesan dikirimkan hanya untuk tujuan yang relevan dan hanya sekali, dengan berbagai metoda komunikasi yang mencakup komunikasi *one-to-many* atau *manytomany*. Sumber data dan tujuan dapat *decoupled* di bawah model tersebut.

A. Protocol Layer

Protokol merupakan sebuah rule, prosedur dan pengaturan sejumlah operasi peralatan komunikasi data, dalam komunikasi data, aturan-aturan meliputi cara membuka hubungan, mengirim paket data, menginformasi jumlah data yang diterima, dan meneruskan pengiriman data. Protokol dapat diterapkan pada perangkat keras, perangkat lunak atau kombinasi dari keduanya. Pada tingkatan yang terendah, protokol mendefinisikan koneksi perangkat keras.

Prinsip dalam membuat protokol ada tiga hal yang harus dipertimbangkan, yaitu efektivitas, kehandalan, dan Kemampuan dalam kondisi gagal di network. Protokol distandarisasi oleh beberapa organisasi yaitu IETF, ETSI, ITU, dan ANSI. Tugas yang biasanya dilakukan oleh sebuah protokol dalam sebuah jaringan diantaranya adalah :

1. Melakukan deteksi adanya koneksi fisik atau ada tidaknya komputer / mesin lainnya.
2. Melakukan metode “jabat-tangan” (handshaking).
3. Negosiasi berbagai macam karakteristik hubungan.
4. Bagaimana mengawali dan mengakhiri suatu pesan.
5. Bagaimana format pesan yang digunakan.
6. Yang harus dilakukan saat terjadi kerusakan pesan atau pesan yang tidak sempurna.
7. Mendeteksi rugi-rugi pada hubungan jaringan dan langkah-langkah yang dilakukan selanjutnya.
8. Mengakhiri suatu koneksi.
9. Dalam Model OSI terdapat 7 layer dimana masing-masing layer mempunyai jenis protokol sesuai dengan peruntukannya. Sebuah standar protokol yang dikenal sebagai OSI (Open System Interconection) model dengan aisitrktur sebagai berikut.

Arsitektur OSI dibuat berlapis-lapis dengan fungsi yang berbeda pada setiap lapisannya. Lapisan yang lebih tinggi menyembunyikan kerumitan dari operasi di lapisan yang lebih rendah dan suatu lapisan hanya dapat di akses oleh lapisan yang ada di atasnya atau di bawahnya.Hal tersebut di maksudkan untuk memberi kemudahankepada para pembuat perangkat keras dan perangkat lunak komunikasi dalam mengembangkan berbagai protokol yang berbeda sesuai kebutuhan. Namun tetap mereka harus mematuhi standar yang telah di berikan OSI.

Lapisan layer protokol tersebut dapat di golongkan lebih jauh menjadi: A. Low level Layers

Yang termasuk Low layers adalah lapisan-lapisan sebagi berikut:

1. Physical Layer: Spesifikasi dan implementasi dari bit-bit dan proses transmisi dari pengirim ke penerima. Berfungsi untuk mendefinisikan media transmisi jaringan, metode pensinyalan, sinkronisasi bit, arsitektur jaringan (seperti halnya Ethernet atau Token Ring), topologi jaringan dan pengkabelan. Selain itu, level ini juga mendefinisikan bagaimana Network Interface Card (NIC) dapat berinteraksi dengan media kabel atau radio.

2. Data Link Layer: Bertanggung jawab mengurusi perubahan bit=bit data menjadi frame untuk mengatasi error dan penontrolan pengiriman frame.

   Befungsi untuk menentukan bagaimana bit-bit data dikelompokkan menjadi format yang disebut sebagai *frame*. Data link layer melakukan tugasnya dengan meletakkan pola bit khusus pada awal dan akhir dari setiap frame untuk menandai mereka, serta komputasi checksum dengan menjumlahkan semua byte dalam bingkai dengan cara tertentu.

Ketika frame tiba, penerima recomputes checksum dari data dan membandingkan hasilnya dengan checksum mengikuti frame. Jika setuju, frame dianggap benar dan diterima. Jika mereka tidak setuju, penerima meminta pengirim untuk retrasmit itu. Frame ditugaskan nomor urut (di header), sehingga semua orang dapat memberitahukan yang mana.

3. Network Layer: Mengatur bagaimana paket-paket di arahkan berdasarkan alamat logik. lapisan bertanggung jawab untuk menerjemahkan alamat logis jaringan ke alamat fisik jaringan. Berfungsi untuk mendefinisikan alamatalamat IP, membuat *header* untuk paket-paket, dan kemudian melakukan routing melalui *internetworking* dengan menggunakan router dan switch layer. Lapisan ini juga member identitas alamat, jalur perjalanan pengiriman data, dan mengatur masalah jaringan misalnya pengiriman paketpaket data. Saat ini, mungkin protokol jaringan yang paling banyak digunakan adalah IP connectionless (Internet Protocol), yang merupakan bagian dari protokol internet. Sebuah paket IP dapat dikirim tanpa pengaturan apapun. Setiap paket IP diarahkan ke tujuan yang independen dari semua orang lain. Tidak ada jalur internal dipilih dan diingat.

B. Transport Layers

Merupakan lapisan yang memberi fasilitas komunikasi bagi kebanyakan sistem tersebar. Berfungsi untuk memecah data ke dalam paket-paket data serta memberikan nomor urut ke paket-paket tersebut sehingga dapat disusun kembali pada sisi tujuan setelah diterima. Selain itu, pada level ini juga membuat sebuah tanda bahwa paket diterima dengan sukses (acknowledgement), dan mentransmisikan ulang terhadap paket-paket yang hilang di tengah jalan.

Pada lapisan ini terdapat dua macam protokol yang sering digunakan, yaitu:

1. Transport Control Protocol (TCP) adalah protocol yang connection-oriented, yang berarti komunikasi yang melewatinya membutuhkan handshaking untuk mengatur koneksi endto-end. Koneksi dapat dibuat dari client ke server, dan kemudian banyak data dapat dikirimkan melalui konesi itu. TCP memiliki karakteristik sebagai berikut:
   - Connection-oriented merupakan sistem yang akan berkomunikasi harus terlebih dulu saling mengetahui dan sepakat
   - Reliable, tersedia mekanisme menjamin paket yang rusak atau hilang dikirim ulang

- Stream –oriented communication
- Membutuhkan sumberdaya komputasi dan jaringan lebih dari UDP

2. User Datagram Protocol (UDP) adalah protocol connectionless message-based yang lebih sederhana. Di protocol connectionless, tidak ada usaha yang dibuat untuk koneksi end-to-end. Koumikasi dicapai dengan mengirimkan informasi satu arah, dari source ke destination tanpa mengecek untuk melihat apakah tujuan masih ada, atau apakah koneksi disiapkan untuk menerima informasi. Paket UDP melewati jaringan dalam unit-unit yang berdiri sendiri. UDP memiliki karakteristik sebagai berikut:
   - Connectionless, tidak memerlukan adanya saling mengetahui dan kesepakatan
   - Unreliable datagram communication, tidak tersedianya mekanisme yang menjamin paket rusak atau dikirim ulang.

Manfaat TCP dibandingkan dengan UDP adalah bahwa ia bekerja andal melalui jaringan apapun. Kelemahan yang jelas adalah bahwa TCP memperkenalkan overhead yang jauh lebih, terutama dibandingkan dengan kasus-kasus di mana jaringan yang mendasarinya sangat handal, seperti dalam sistem area lokal. Ketika kinerja dan kehandalan yang dipertaruhkan, solusi alternatif selalu untuk menggunakan UDP, dan mengkombinasikannya dengan kesalahan tambahan dan kontrol aliran yang dioptimalkan untuk aplikasi tertentu. Kelemahan dari pendekatan ini adalah bahwa pekerjaan pembangunan banyak ekstra perlu dilakukan, tetapi juga bahwa solusi proprietary diperkenalkan, yang mempengaruhi keterbukaan sistem.

Apa yang membuat TCP begitu menarik dalam banyak kasus, adalah bahwa hal itu tidak disesuaikan untuk mendukung jawaban perilaku permintaan sinkron interaksi klien yang paling server. Dalam keadaan normal, ketika pesan tidak tersesat, menggunakan TCP untuk interaksi client server hasil seperti yang ditunjukkan pada gambar 2-4 (a). Pertama, klien memulai setup sambungan, yang dilakukan dengan menggunakan tiga cara protokol jabat tangan, ditampilkan sebagai tiga pesan pertama di gambar 2-4 (a). Protokol ini diperlukan untuk kedua belah pihak untuk mencapai kesepakatan pada urutan penomoran untuk paket yang akan dikirim melalui sambungan. Ketika koneksi telah dibentuk, client mengirimkan permintaannya (pesan 4), langsung diikuti oleh paket memberitahu server untuk menutup koneksi (pesan 5).

Server merespon dengan segera mengakui bahwa ia menerima permintaan klien, piggybacked dengan pengakuan bahwa koneksi akan clossed bawah (pesan 6). Server kemudian melakukan pekerjaan yang diminta dan mengirimkan jawaban kepada klien (pesan 7), diikuti dengan permintaan untuk

melepaskan koneksi juga (pesan 8). Klien hanya perlu merespon dengan pengakuan untuk menyelesaikan komunikasi dengan server (pesan 9).

Jelas, banyak overhead dalam TCP berasal dari benar-benar mengelola koneksi. Ketika TCP digunakan untuk interaksi client server, itu jauh lebih murah untuk menggabungkan pengaturan koneksi dengan segera mengirimkan permintaan, dan seperti bijaksana untuk menggabungkan mengirimkan jawaban dengan menutup koneksi. Protokol yang dihasilkan disebut TCP untuk Transaksi, disingkat T / TCP, dan esensi dari bagaimana ia beroperasi. Apa yang terjadi dalam keadaan normal, adalah bahwa klien mengirimkan pesan tunggal (ditampilkan sebagai pesan 1) yang berisi tiga potongan informasi: permintaan untuk mengatur koneksi, permintaan layanan yang sebenarnya, dan permintaan memberitahu server yang segera bisa merobek turun koneksi sesudahnya.

Server merespon setelah ia dilayani permintaan yang sebenarnya, sehingga dapat mengirim jawaban bersama dengan data yang diperlukan untuk menerima koneksi, dan segera meminta rilis, ditampilkan sebagai pesan

C. Higher Level Layer

Di atas lapisan transport. OSI membedakan tiga lapisan tambahan. Dalam prakteknya, hanya lapisan aplikasi yang pernah digunakan. Bahkan, di suite internet protocol, segala sesuatu di atas lapisan transport dikelompokkan bersama-sama. Dalam menghadapi sistem middleware, akan kita lihat dalam bagian ini bahwa baik OSI maupun pendekatan internet benar-benar tepat.

Dalam Higher Level Layer ini terbagi menjadi tiga lapisan tambahan yaitu:

1. Sesi dan Presentasi Protokol

Lapisan sesi dasarnya merupakan versi yang disempurnakan dari lapisan transport. Menyediakan kontrol dialog, untuk melacak pihak mana saat berbicara, dan menyediakan fasilitas sinkronisasi. Sehingga jika terjadi kecelakaan, yang terakhir ini berguna untuk memungkinkan pengguna memasukkan pos-pos pemeriksaan dalam transfer yang panjang. Hal ini diperlukan untuk kembali hanya untuk pos pemeriksaan terakhir, daripada semua jalan kembali ke awal. Dalam prakteknya, beberapa aplikasi tertarik dalam lapisan sesi dan jarang didukung. Hal ini tidak hadir dalam acara suite protokol Internet.

Berbeda dengan lapisan bawah, yang prihatin dengan mendapatkan bit dari pengirim ke penerima andal dan efisien, lapisan presentasi berkaitan dengan arti dari bit. Pesan yang paling tidak terdiri dari string bit acak, tetapi informasi lebih terstruktur seperti nama orang, alamat, jumlah uang, dan sebagainya. Pada lapisan presentasi adalah mungkin untuk menentukan catatan yang berisi bidangbidang seperti

ini dan kemudian memiliki Sener memberitahukan penerima bahwa pesan berisi catatan tertentu dalam format tertentu. Hal ini memudahkan untuk mesin dengan representasi internal yang berbeda untuk berkomunikasi.

2. Protocol Application

Menyediakan layanan untuk aplikasi misalnya transfer file, email, akses suatu komputer atau layanan. Lapisan aplikasi OSI awalnya dimaksudkan untuk menampung koleksi aplikasi jaringan standar seperti untuk surat elektronik, transfer file, dan emulasi terminal. Sampai saat ini telah menjadi wadah untuk semua aplikasi dan protokol yang dalam satu cara atau yang lain tidak masuk ke dalam salah satu lapisan yang mendasarinya. Dari perspektif model referensi OSI, hampir semua sistem terdistribusi hanya aplikasi.

Ada juga banyak tujuan umum protokol yang berguna untuk banyak aplikasi, tetapi yang tidak dapat dikualifikasikan sebagai protokol transport. Dalam banyak kasus, protokol seperti jatuh ke dalam kategori protokol middleware, yang akan kita bahas selanjutnya.

3. Middleware Protocol

Middleware adalah sebuah aplikasi yang logis tinggal di lapisan aplikasi, tetapi yang mengandung banyak tujuan umum protokol yang menjamin lapisan mereka sendiri, independen lainnya, aplikasi yang lebih spesifik. Perbedaan dapat dibuat antara tingkat tinggi protokol komunikasi dan protokol untuk membangun layanan middleware berbagai.

Dibuat untuk menyediakan layanan layanan protokol yang seragam dan dapat digunakan oleh aplikasi-aplikasi yang berbeda-beda. Pada lapisan ini terdapat sekumpulan protokol komunikasi yang beragam yang memungkinkan berbagai macam aplikasi dapat berkomunikasi. Middleware juga memberi fasilitas marshalling dimana terdapat proses pengubahan data dalam komunikasi antar proses menjadi bentuk yang siap dikirim melalui jaringan sehingga dapat tetap konsisten sampai di si penerima data dan kebalikannya.

Middleware komunikasi protokol mendukung tingkat tinggi layanan komunikasi. Misalnya, di bagian dua berikutnya kita akan membahas protokol yang memungkinkan proses untuk memanggil prosedur atau invok obyek pada mesin remote dengan cara yang sangat transparan. Demikian juga, ada layanan komunikasi tingkat tinggi untuk menetapkan dan sinkronisasi aliran untuk mentransfer data real-time, seperti yang diperlukan untuk aplikasi multimedia. Sebagai contoh terakhir, beberapa sistem middleware menawarkan layanan multicast handal yang skala untuk ribuan penerima tersebar di wide

area network. Beberapa protokol komunikasi middleware yang sama bisa juga termasuk dalam lapisan transport, tapi mungkin ada alasan spesifik untuk menjaga mereka pada tingkat yang lebih tinggi. Misalnya, multicasting layanan yang handal Taht skalabilitas jaminan dapat diimplementasikan hanya jika persyaratan aplikasi diperhitungkan. Akibatnya, sistem middleware mungkin menawarkan yang berbeda (merdu) protokol, masingmasing pada gilirannya diimplementasikan dengan menggunakan protokol transport yang berbeda, tapi mungkin menawarkan antarmuka tunggal.

4.5 Konsep objek terdistribusi dan object interface

Meskipun teknologi RPC ini relatif sudah memberikan kenyamanan bagi developer namun seiring dengan perkembangannya, sistem ini dinilai tidak efisien lagi. Dalam membuat aplikasi client server, programmer masih harus membuat fungsifungsi yang sama untuk aplikasi yang berbeda. Kadang kala kode program yang sama digunakan dengan melakukan copy paste dan melakukan sedikit perubahan untuk menyesuaikan dengan aplikasi yang baru dibuat. Jika ada perubahan, fungsi tersebut dalam masing-masing aplikasi harus di update satu persatu lagi. Hal ini mengakibatkan perawatan program menjadi susah dan fungsi-fungsi tersebut dapat menjadi tidak konsisten satu sama lain.

4.5.1 CORBA

Common Object Request Broker Architecture (CORBA) merupakan standar yang dikeluarkan oleh Object Management Group (OMG). Spesifikasi CORBA ini berisi sebuah spesifikasi infrastruktur yang disebut Object Request Broker (ORB) yang memungkinkan aplikasi klien untuk dapat berkomunikasi dengan obyek secara remote. Spesifikasi ini meliputi antarmuka program, protokol komunikasi dan model obyek atau layanan yang memungkinkan aplikasi yang ditulis dengan berbagai macam bahasa pemrograman.

CORBA membungkus kode program yang dibuat dengan bahasa pemrograman tertentu menjadi sebuah obyek yang ditambah dengan informasi mengenai kemampuan kode program dan cara mengaksesnya. Obyek tersebut dapat dipanggil oleh program lain melalui jaringan. CORBA menggunakan interface definition languange (IDL) untuk menunjukkan interface atau antarmuka yang dapat digunakan oleh program atau obyek lain. Dari IDL tersebut CORBA akan memetakannya ke implementasi yang lebih spesifik dari masing-masing bahasa pemrograman.

4.5.2 COM

Component Object Model (COM) adalah teknologi yang diciptakan oleh Microsoft untuk memungkinkan komunikasi antaraplikasi. Teknologi ini sudah disediakan untuk beberapa platform tetapi kebanyakan digunakan untuk platform Windows. Teknologi ini sudah diperkenalkan oleh microsoft pada tahun 1993 tetapi baru populer pada tahun 1997. Perkembangan teknologi COM ini bermula dari teknologi OLE (Object Linking and Embedding) yang dibuat untuk memungkinkan aplikasi dapat saling bertukar data.

4.5.3 DCOM

Pada tahun 1996 diperkenalkan Distributed Component Object Model (DCOM) sebagai jawaban Microsoft atas CORBA. DCOM dibandingkan dengan COM memiliki kelebihan mampu untuk terdistribusi dan berkomunikasi antarkomponen melalui jaringan. DCOM dan CORBA saling berkompetisi untuk menjadi standar dalam distribusi komponen melalui internet. Namun dibalik kesulitan dalam hal keamanan, sebuah browser yang berjalan menggunakan teknologi http sudah dapat menggantikan teknologi tsb.

OBJECT INTERACTION: RMI AND RPC

5.1 Definisi RMI

Remote Method Invocation (RMI) adalah sebuah teknik pemanggilan method remote yang lebih secara umum lebih baik daripada RPC. RMI menggunakan paradigma pemrograman berorientasi obyek (Object Oriented Programming). RMI memungkinkan kita untuk mengirim obyek sebagai parameter dari remote method. Dengan dibolehkannya program Java memanggil method pada remote obyek, RMI membuat pengguna dapat mengembangkan aplikasi Java yang terdistribusi pada jaringan.

RMI menyediakan mekanisme dimana server dan client berkomunikasi dan memberikan informasi secara timbal balik. Aplikasi semacam ini seringkali disebut aplikasi objek terdistribusi. Langkah-Langkah Pembuatan Program dengan RMI. Dalam RMI, semua informasi tentang satu pelayanan server disediakan dalam suatu definisi remote interface. Dengan melihat pada definisi interface, seorang pemrogram dapat memberitahukan method apa yang dapat dikerjakan oleh server, meliputi data apa yang diterima dan data apa yang akan dikirim sebagai tanggapan.

Definisi yang ada pada remote interface menentukan karakteristik methods yang disediakan server yang dapat dilihat oleh client. Client programmer harus dapat mengetahui methods apa yang disediakan server dan bagaimana memanggilnya langsung dengan melihat ke remote interface. Client mendapatkan referensi ke remote object melalui RMI registry.

Membangun suatu aplikasi terdistribusi menggunakan RMI meliputi 6 langkah.

Keenam langkah tersebut adalah:

1. Mendefinisikan remote interface
2. Implementasi remote interface dan server
3. Pengembangan client (atau applet) yang menggunakan remote interface
4. Mengkompilasi source files dan mem-buat stub and skeletons
5. Memulai (start) RMI registry
6. Menjalankan server dan client

5.1.1 Langkah Pembuatan Program dengan RMI

Dalam RMI, semua informasi tentang satu pelayanan server disediakan dalam suatu definisi remote interface. Dengan melihat pada definisi interface, seorang pemrogram dapat memberitahukan method apa yang dapat dikerjakan oleh server, meliputi data apa yang diterima dan data apa yang akan dikirim sebagai tanggapan.

Definisi yang ada pada remote interface menentukan karakteristik methods yang disediakan server yang dapat dilihat oleh client. Client programmer harus dapat mengetahui methods apa yang disediakan server dan bagaimana memanggilnya langsung dengan melihat ke remote interface. Client mendapatkan referensi ke remote object melalui RMI registry.

Membangun suatu aplikasi terdistribusi menggunakan RMI meliputi 6 langkah.

Keenam langkah tersebut adalah:

1. Mendefinisikan remote interface
2. Implementasi remote interface dan server
3. Pengembangan client (atau applet) yang menggunakan remote interface
4. Mengkompilasi source files dan mem-buat stub and skeletons
5. Memulai (start) RMI registry
6. Menjalankan server dan client

5.1.2 Implementasi RMI

Remote Modul Reference Remot referensi modul bertanggung jawab untuk menerjemahkan antara local dan referensi remote objek dan untuk menciptakan referensi remot objek. Untuk mendukung tugasnya ini, referensi modul remote dalam setiap proses memiliki tabel remote objek yang mencatat korespondensi antara objek lokal referensi dalam proses dan referensi remote objek (dimana system-wide) (George Coulouris dkk, hal 176) **Tindakan remote referensi modul adalah sebagai berikut:**

Ketika remote objek untuk diteruskan sebagai argumen atau hasil untuk pertama kalinya, modul referensi remot modul diminta untuk membuat referensi objek remote, yang menambahkan tabel.

Ketika referensi objek remote tiba dalam permintaan atau membalas pesan, remote modul referensi diminta untuk menyesuaikan referensi obyek lokal, yang mungkin mengacu baik pada proxy atau ke objek remot. Dalam hal objek remote referensi tidak ada dalam tabel, menciptakan perangkat lunak RMI proxy baru dan meminta remote referensi modul untuk menambahkannya ke tabel.

Software RMI ini terdiri dari suatu lapisan perangkat lunak antara application level objek dan komunikasi dan reeferensi remot modul. Peran middleware objek adalah sebagai berikut:

- Proxy: Peran proxy adalah untuk membuat permohonan metoderemot transparan untuk klien dengan bertingkah seperti objek lokal ke invoker, tetapi selain melaksanakan suatu permintaan ini akan diteruskan sebuah pesan ke objek remote. Itu menyembunyikan

rincian remot objek referensi, yang menyusun argumen, menguraikan hasil dan pengiriman dan penerimaan pesan dari klien.

- Operator/Dispatcher : Sebuah server memiliki satu operator dan kerangka untuk masingmasing mewakili kelas remote objek. Dalam contoh kita, server memiliki operator dan kerangka untuk kelas remot objek B. operator menerima pesan permintaan dari modul komunikasi. Ia menggunakan methodId untuk memilih metode yang tepat dalam kerangka kemudian menyampaikan pesan permintaan. Operator dan proxy menggunakan sama alokasi methodld terhadap metode antarmuka remote.

- Skeleton: Kelas jauh objek memiliki kerangka, yang mengimplementasikan metode dalam antarmuka remote. Mereka dilaksanakan cukup berbeda dari metode-metode di objek remote. Sebuah metode menguraikan kerangka argumen dalam pesan permintaan dan memanggil metode yang sesuai dalam objek remote.

Contoh implementasi dari RMI di antaranya :

Perusahaan programming Avitek yang berlokasi di Amerika Serikat, membuat program sistem accounting untuk intranet yang memungkinkan klien untuk mengupdate dan mengubah data dengan mudah. Tujuan dari proyek ini adalah untuk membuat dan mendukung pembuatan dari bukti nyata untuk konsep penggunaan Java yang dikombinasikan dengan database.

Perusahaan CEAS Consulting yang menyediakan jasa custom re-engineering dan otomasi proses untuk perusahaan-perusahaan manufakturing dan teknik, telah membuat program sistem terdistribusi untuk klien mereka. Gambaran program mereka adalah seperti berikut :

5.1.3 Cara Kerja RMI

Dalam model ini, sebuah proses memanggil method dari objek yang terletak pada suatu host/computer remote. Dalam paradigma ini, penyedia layanan mendaftarkan dirinya dengan server direktori pada jaringan. Proses yang menginginkan suatu layanan mengontak server direktori saat runtime, jika layanan tersedia, maka referensi ke layanan akan diberikan. Dengan menggunakan referensi ini, proses dapat berinteraksi dengan layanan tsb. Paradigma ini ekstensi penting dari paradigma RPC. Perbedaannya adalah objek yang memberikan layanan didaftarkan (diregister) ke suatu layanan direktori global, sehingga memungkinkan untuk ditemukan dan diakses oleh aplikasi yang meminta layanan tersebut.

Contoh aplikasi untuk meremote pada teknik RMI (Remote Method Invocation) menggunakan teamviewer untuk meremote computer lain.

Teamviewer adalah suatu program yang cukup sederhana dan sangat mudah digunakan untuk beberapa keperluan terutama melakukan akses PC secara remote melalui internet.

5.1.4 Cara Meremote RMI

Tampilan utama TeamViewer, jika sudah tampil (Ready to connect (secure connection)) maka sudah siap melakukan koneksi ke PC lain, kemudian masukan ID PC klien jika koneksi berhasil maka akan muncul kotak "Password", isi password teamviewer PC yang akan anda remote. Jika berhasil maka akan tampil desktop PC yang diremote tersebut. Jika ingin melakukan

File Transfer, maka pilih "File Transfer" pada bagian pilihan yang terlihat dibawah kotak ID, kemudian klik "Connect to partner".

Jika ingin menggunakan password dan ID yang tetap maka cukup tempatkan kursor mouse pada bagian kotak password, maka akan tampil seperti gambar diatas, pilih "Set user definied password" kemudian masukan password yang anda inginkan. Hasilnya Setelah kita melakukan setting pada teamviewer dan setelah login dan memasukkan password tujuan dengan benar maka kita bisa mengakses computer yang kita remote secara penuh.

5.1.5 Keuntungan dan Kekurangan RMI Keuntngan RMI

- Salahsatu keuntungan RMI adalah kemampuan untuk download zytecodes (code) dari suatu object‟s class, jika class tsb tidak terdefinisikan di VM-nya penerima.
- Type-type dan metode-metode object (class), yang terletak dalam satu VM, dapat dikirim ke VM yang lain, yang mungkin saja remote.
- Sifat-sifat object yang terkirim ini tidak berubah sama sekali Kelemahan RMI: proses pembukaan socket yang kadang-kadang tidak dapat diimplementasikan lewat jaringan internet, tapi hal ini bisa diatasi dengan menggunakan Spring HttpInvoker, sama persis dengan RMI tapi lewat protokol HTTP.

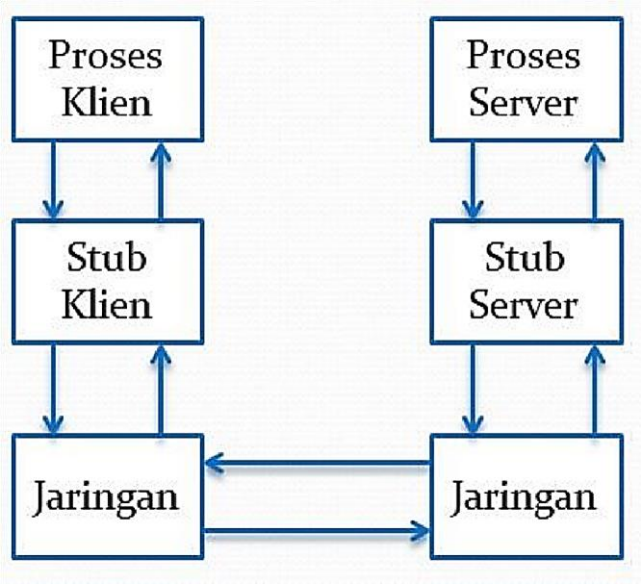
5.2 Definisi RPC

Remote Procedure Call (RPC) adalah sebuah metode yang memungkinkan kita untuk mengakses sebuah prosedur yang berada di komputer lain. Untuk dapat melakukan ini sebuah server harus menyediakan layanan remote procedure. Pendekatan yang dilakuan adalah sebuah server membuka socket, lalu

menunggu client yang meminta prosedur yang disediakan oleh server. Bila client tidak tahu harus menghubungi port yang mana, client bisa me- request kepada sebuah matchmaker pada sebuah RPC port yang tetap. Matchmaker akan memberikan port apa yang digunakan oleh prosedur yang diminta client.

RPC masih menggunakan cara primitif dalam pemrograman, yaitu menggunakan paradigma procedural programming. Hal itu membuat kita sulit ketika menyediakan banyak remote procedure. RPC menggunakan socket untuk berkomunikasi dengan proses lainnya. Pada sistem seperti SUN, RPC secara default sudah ter- install kedalam sistemnya, biasanya RPC ini digunakan untuk administrasi sistem. Sehingga seorang administrator jaringan dapat mengakses sistemnya dan mengelola sistemnya dari mana saja, selama sistemnya terhubung ke jaringan.

Langkah-langkah dalam RPC

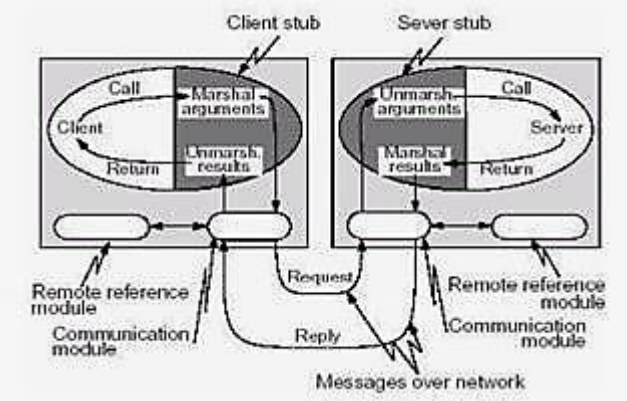


1. Klien memanggil prosedur stub lokal. Prosedur Stub akan memberikan parameter dalam suatu paket yang akan dikirim ke jaringan. Proses ini disebut sebagai marshalling.
2. Fungsi Network pada O/S (Operating system - Sistem Operasi) akan dipanggil oleh stub untuk mengirim suatu message.
3. Kemudian Kernel ini akan mengirim message ke sistem remote. Kondisi ini dapat berupa connectionless atau connection-oriented.
4. Stub pada sisi server akan melakukan proses unmarshals pada paket yang dikirim pada network.
5. Stub pada server kemudian mengeksekusi prosedur panggilan lokal.

6. Jika eksekusi prosedur ini telah selesai, maka eksekusi diberikan kembali ke stub pada server.
7. Stub server akan melakukan proses marshals lagi dan mengirimkan pesan nilai balikan (hasilnya) kembali ke jaringan.
8. Pesan ini akan dikirim kembali ke klien.
9. Stub klien akan membaca message ini dengan menggunakan fungsi pada jaringan.
10. Proses unmarshalled kemudian dilakukan pada pesan ini dan nilai balikan akan diambil untuk kemudian diproses pada proses lokal. Proses tersebut akan dilakukan berulangulang (rekursif) dalam pengeksekusian RPC dalam suatu remote sistem.

5.2.1 Implementasi RPC

Untuk proses nya kurang lebih sama dengan RMI. Kalau RMI kita mengenal proxy dan skeleton, pada RPC dikenal dengan Stub(Client stub dan Server stub)



Gambar Ilustrasi Implementasi RPC

Remote Reference Modul dan Communication Modul berada pada tatanan sistem operasi. Contoh implementasi adalah Sun Microsystems Open Network Computing (ONC) : RPC specification, XDR (eXternal Data Representation) standard, UDP atau TCP transport protocol. Xerox Courier : RPC model, Data representation standard, XNS (Xerox Network Systems) SPP (Sequenced Packet Protocol) sbg transport protocol, Apollo s Network Computing Architecture (NCA), RPC protocol, NDR (Network Data Representation).

5.2.2 Cara Kerja RPC

Tiap prosedur yang dipanggil dalam RPC, maka proses ini harus berkoneksi dengan server remote dengan mengirimkan semua parameter yang dibutuhkan, menunggu balasan dari server dan melakukan

proses kemudian selesai. Proses di atas disebut juga dengan stub pada sisi klien. Sedangkan Stub pada sisi server adalah proses menunggu tiap message yang berisi permintaan mengenai prosedur tertentu.

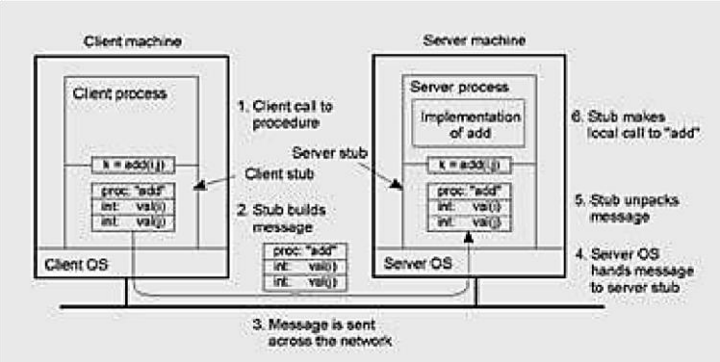


Diagram diatas memberikan gambaran mengenai flow dari eksekusi dalam proses RPC. Berikut ini adalah diagram yang akan menjelaskan secara rinci mengenai proses yang terjadipada klien dan server dalam eksekusi suatu prosedur RPC : penjelasan dari diagram diatas :

1. Klien memanggil prosedur stub lokal. Prosedur Stub akan memberikan parameter dalam suatu paket yang akan dikirim ke jaringan. Proses ini disebut sebagai marshalling.
2. Fungsi Network pada O/S (Operating system – Sistem Operasi) akan dipanggil oleh stub untuk mengirim suatu message.
3. Kemudian Kernel ini akan mengirim message ke sistem remote. Kondisi ini dapat berupa connectionless atau connection-oriented.
4. Stub pada sisi server akan melakukan proses unmarshals pada paket yang dikirim pada network.
5. Stub pada server kemudian mengeksekusi prosedur panggilan lokal.
6. Jika eksekusi prosedur ini telah selesai, maka eksekusi diberikan kembali ke stub pada server.
7. Stub server akan melakukan proses marshals lagi dan mengirimkan message nilai balikan (hasilnya) kembali ke jaringan.
8. Message ini akan dikirim kembali ke klien.
9. Stub klien akan membaca message ini dengan menggunakan fungsi pada jaringan.
10. Proses unmarshalled kemudian dilakukan pada message ini dan nilai balikan akan diambil untuk kemudian diproses pada proses lokal.

Proses diatas akan dilakukan berulang-ulang (rekursif) dalam pengeksekusian RPC dalam suatu remote sistem. Contoh aplikasi untuk meremote pada teknik
RPC (Remote Procedure Call) adalah menggunakan putty untuk melakukan SSH.

5.2.3 Kelebihan dan Kekurangan pada RPC Kelebihan RPC Kelebihan

1) Relatif mudah digunakan
2) Pemanggilan remote procedure tidak jauh berbeda dibandingkan pemanggilan local procedure. Sehingga pemrogram dapat berkonsentrasi pada software logic, tidak perlu memikirkan low level details seperti soket, marshalling dan unmarshalling.
3) Robust (Sempurna)
4) Sejak th 1980-an RPC telah banyak digunakan dalam pengembangan mission-critical application yang memerlukan scalability, fault tolerance, dan reliability.

Kelemahan RPC

1) Tidak fleksibel terhadap perubahan
2) Static relationship between client & server at run-time.
3) Berdasarkan prosedural/structured programming yang sudah ketinggalan jaman dibandingkan Object Oriented Proggraming
4) Kurangnya location transparency
5) Misalnya premrogram hanya boleh melakukan pass by value, bukan pass by reference.
6) Komunikasi hanya antara 1 klien & 1 server (one-to-one at a time).
7) Komunikasi antara 1 klien & beberapa server memerlukan beberapa koneksi yang terpisah.

5.3 Middleware

Tujuan utama layanan middleware adalah untuk membantu memecahkan interkoneksi beberapa aplikasi dan masalah interoperabilitas. Middleware sangat dibutuhkan untuk bermigrasi dari aplikasi mainframe ke aplikasi client/server dan juga untuk menyediakan komunikasi antar platform yang berbeda.

Perangkat lunak ini terdiri dari serangkaian pelayanan yang mengizinkan bermacammacam proses berjalan dalam satu atau lebih mesin untuk dapat saling berinteraksi satu sama yang lainnya. Lambat laun teknologi ini menyediakan kemampuan interoperabilitas yang mendukung pada perpindahan ke

arsitektur distribusi yang berhubungan, yang biasanya sering digunakan untuk mendukung dan menyederhanakan kerumitan, aplikasi terdistribusi.

Termasuk didalamnya, web server, aplikasi server dan peralatan sama yang mendukung pengembangan dan pengantaran aplikasi. Middleware secara khusus menjadi bagian dari teknologi informasi modern berbasis XML, SOAP, web service dan pelayanan berbasis arsitektur. Middleware berada diantara aplikasi perangkat lunak yang mungkin bekerja pada system operasi yang berbeda. Middleware serupa dengan middle layer dari sebuah tiga baris sistem arsitektur tunggal, kecuali usahanya melewati bermacam-macam system atau aplikasi. Contohnya perangkat lunak EAI (Enterprise Application Integration), perangkat lunak telekomunikasi, monitor transaksi dan perangkat lunak pemesanan dan pengantrian.

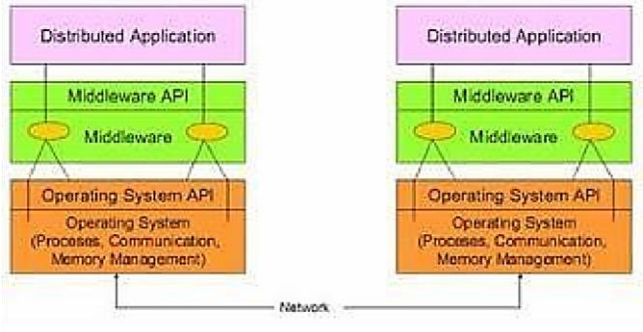
Middleware adalah Software yang berfungsi sebagai lapisan konversi atau penerjemah.Juga sebagai consolidator dan integrator. Middleware saat ini dikembangkan untuk memungkinkan satu aplikasi berkomunikasi dengan lainnya walaupun berjalan pada platform yang berbeda. Saat ini terdapat bermacam produk yang menawarkan middleware.

5.3.1 Tujuan dan asalusul Middleware

Middleware adalah S/W penghubung yang berisi sekumpulan layanan yang memungkinkan beberapa proses dapat berjalan pada satu atau lebih mesin untuk saling berinteraksi pada suatu jaringan

Middleware sangat dibutuhkan untuk bermigrasi dari aplikasi mainframe ke aplikasi client/server dan juga untuk menyediakan komunikasi antar platform yang berbeda Middleware yang paling banyak dipublikasikan :

- Open Software Foundation's Distributed Computing Environment (DCE),
- Object Management Group's Common Object Request Broker Architecture (CORBA),
- Microsoft's COM/DCOM (Component Object Model)



Lapisan middleware

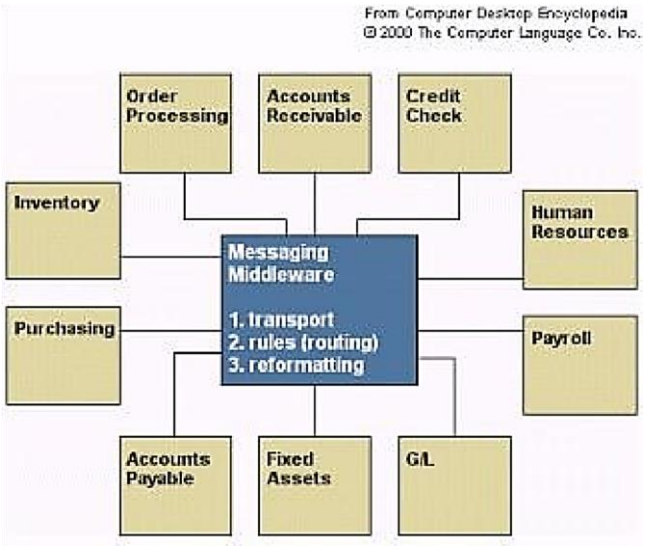
5.3.2 Arsitektur Teknis

1. Layanan Middleware merupakan sekumpulan S/W terdistribusi yang menempati lapisan antara aplikasi dan sistem operasi serta layanan jaringan di suatu node pada jaringan komputer
2. Menyediakan kumpulan fungsi API (Application Programming Interfaces) yang lebih tinggi daripada API yang disediakan sistem operasi dan layanan jaringan yang memungkinkan suatu aplikasi dapat :
   - Mengalokasikan suatu layanan secara transparan pada jaringan
   - Menyediakan interaksi dengan aplikasi atau layanan lain
   - Tidak tergantung dari layanan jaringan
   - Handal dan mampu memberikan suatu layanan
   - Diperluas (dikembangkan) kapasitasnya tanpa kehilangan fungsinya

5.3.3 Messaging Middleware

Merupakan antarmuka dan transportasi antar aplikasi

- Menyimpan data dalam suatu antrian message jika mesin tujuan sedang mati atau overloaded
- Mungkin berisi business logic yang merutekan message ke tujuan sebenarnya dan memformat ulang data lebih tepat.
- Sama seperti sistem messaging email, kecuali messaging middleware digunakan untuk mengirim data antar aplikasi



gambar massaging middleware:

5.3.4 Produk Messaging Middleware

Produk utama messaging (pengiriman pesan) untuk pengaturan komunikasi asinkronus antar aplikasi adalah MQSeries dari IBM. MQSeries telah dipasangkan pada semua platform server. Microsoft memperkenalkan sistem messagingnya sendiri yang digabungkan dengan Component Object Model(COM), yaitu Microsoft Message Queue Server (MSMQ). MSMQ dan MQSeries menawarkan fungsi yang sama.

5.3.5 Distributed Processing

- Sistem objek terdistribusi, seperti CORBA, DCOM dan EJB memungkinkan proses-proses untuk dijalankan di sembarang node pada jaringan
- Sistem Objek terdistrbusi tersebut berbeda dari messaging middleware, yang menyebabkan proses proses (komponen/objek) dijalankan dalam mode synchronous daripada pengiriman data secara asynchronous

5.3.6 Remote Procedure Calls

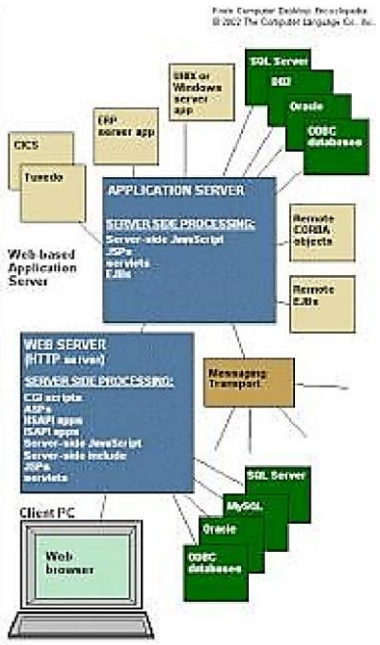
a. Remote Procedure Calls (RPC) memungkinkan suatu bagian logika aplikasi untuk didistribusikan pada jaringan. Contoh :
    - SUN RPC, diawali dengan network file system (SUN NFS),
    - DCE RPC, sebagai dasar Microsoft‟s COM.

b. Object Request Brokers (ORBs) memungkinkan objek untuk didistribusikan dan dishare pada jaringan yang heterogen.
    - Pengembangan dari model prosedural RPC,
    - Sistem objek terdistribusi, seperti CORBA, DCOM,EJB, dan .NET memungkinkan proses untuk dijalankan pada sembarang jaringan.

5.3.7 Middleware Application Server

a. Sebuah Web-based Application server, yang menyediakan antarmuka untuk berbagai aplikasi, digunakan sebagai middleware antara browser dan aplikais.

b. J2EE adalah contoh application serverA wide range of server-side processing has been supported by appservers(i.e.;J2EE).



Pertimbangan Pemakaian

Tujuan utama layanan middleware adalah untuk membantu memecahkan interkoneksi beberapa aplikasi dan masalah interoperabilitas. Bagaimanapun juga middleware bukanlah “obat mujarab” :

- Ada jarak antara prinsip dan praktek. Beberapa middleware membuat suatu aplikasi tergantung pada suatu produk tertentu
- Sedikitnya jumlah middleware menjadikan rintangan tersendiri. Untuk menjaga lingkungan komputasi mudah diatur,pengembang biasanya memilih sejumlah kecil layanan yang memenuhi kebutuhan mereka
- Selama layanan middleware masih memunculkan abtraksi pemrograman terdistrbusi, middleware masih akan memberikan bagi si pengembang suatu pilihan rancangan aplikasi yang cukup sulit. Contoh : pengembang masih harus menentukan layanan atau fungsi apa yang harus diletakkan pada client ataupun server.

OPERATING SYSTEM SUPPORT

6.1 Definisi Operating System Support

Sistem operasi atau dalam bahasa Inggris: Operating System atau OS adalah perangkat lunak sistem yang bertugas untuk melakukan control dan manajemen perangkat keras serta operasi-operasi dasar sistem,termasuk menjalankan software aplikasi seperti program-program pengolah kata dan browser web. Secara umum, Sistem Operasi adalah software pada lapisan pertama yang ditaruh pada memori komputer pada saat komputer dinyalakan. Sedangkan softwaresoftware lainnya dijalankan setelah Sistem Operasi berjalan,dan Sistem Operasi akan melakukan layanan inti umum untuk software-software itu. Layanan inti umum tersebut seperti akses ke disk manajemen memori, skeduling task, dan antar-muka user.Sehingga masing-masing software tidak perlu lagi melakukan tugas-tugas inti umum tersebut, karena dapat dilayani dan dilakukan oleh Sistem Operasi.Bagian kode yang melakukan tugas-tugas inti dan umum tersebut dinamakan dengan kernel suatu Sistem Operasi. Sistem Operasi secara umum terdiri dari beberapa bagian :

a. Mekanisme Boot, yaitu meletakan kernel ke dalam memory kernel, kernel dapat dikatakan sebagai inti dari Sistem Operasi.
b. Command Interpreter atau Shell, bertugas untuk membaca input berupa perintah dan menyediakan beberapa fungsi standar dan fungsi dasar yang dapat dipanggil oleh aplikasi/program maupun piranti lunak lain.
c. Driver untuk berinteraksi dengan hardware sekaligus mengontrol kinerja hardware.
d. Resource Allocator Sistem Operasi bertugas mengatur dan mengalokasikan sumber daya dari perangkat.
e. Handler berperan dalam mengendalikan sistem perangkat agar terhindar dari kekeliruan (error) dan penggunaan sumber daya yang tidak perlu.

6.2 SEJARAH DAN PERKEMBANGAN SISTEM OPERASI

Menurut Tanebaum, Sistem Operasi mengalami perkembangan yang dapat dibagi ke dalam 4 generasi:

a. Generasi Awal

Perkembangan awal Sistem Operasi masih dilakukan secara manual dalam artian belum muncul adanya Sistem Operasi yang secara otomatis artinyabelum mendukung layanan pekerjaan yang dapat dilakukan dalam 1 rangkaian.

b. Generasi Kedua

Di generasi ini sudah diperkenalkannya perkejaan yang dapat dilakukan dalam 1 rangkaian atau biasa disebut dengan Batch Proccessing System.

c. Generasi Ketiga

Pada generasi ketiga, Sistem Operasi sudah mendukung layanan Multi-User, MultiProgramming dan Batch Proccessing System (Multi-Task).

d. Generasi Keempat

Di masa ini, sudah diperkenankannya GUI (Graphical User Interface) yang artinya Sistem Operasi memiliki tampilan dan dengan bermodalkan mouse,End-User dapat menjalankan aplikasi/porgram atau piranti lunak.

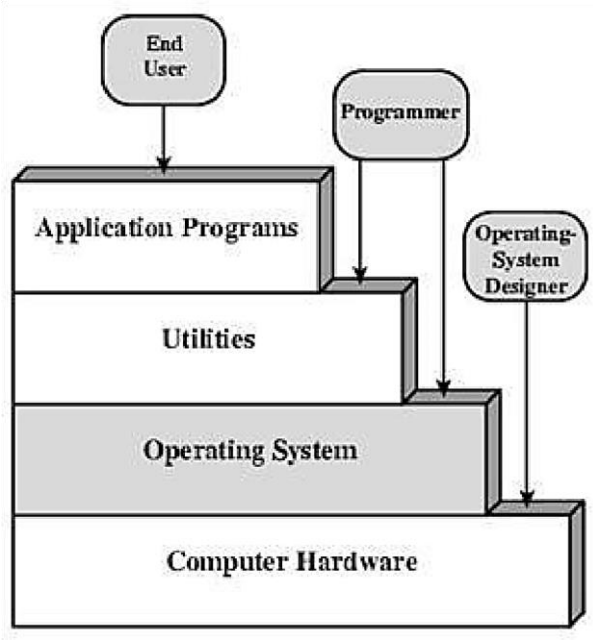
e. Generasi Selanjutnya

Pada generasi selanjutnya diperkenalkan Sistem Operasi yang berada dalam sebuah Sistem Operasi, ini adalah contoh sebuah Sistem Operasi berbasikan Website yang berkerja di dalam sebuah Sistem Operasi. Dan generasi selanjutnya diperkenalkanlah Sistem Operasi bergerak (Mobile) pada perangkat bergerak seperti PDA, Poket PC, dan lain sebagainya.

Digenerasi selanjutnya diperkenalkan juga teknologi Sistem Operasi jaringan yang sifatnya virtual, sehingga dalam 1 jaringan hanya diinstal 1 buah Sistem Operasi pada Perangkat yang bertugas menjadi Server.Selain itu, diperkenalkan pula Cross Platform Operating System yang artinya dapat menggabungkan 2 Sistem Operasi berbeda seperti : Linux danWindows

6.3 Dukungan Sistem Operasi

Sistem operasi adalah sebuah program yang mengatur sumber daya komputer, yang menyediakan layanan untuk program dan menjadual eksekusi program lain. Sistem operasi merupakan program yang mengontrol eksekusi program aplikasi dan bertindak sebagai antarmuka antara pengguna komputer dengan hardware. Sistem operasi mempunyai peran dalam memberdayakan semua resource (layanan, interrupt, dan eksekusi), sehingga semua komponen berfungsi dengan baik (ALU, Memori, I/O), di dalam melakukan suatu proses operasi yang harus dikerjakan.



Lapisan Sebuah Sistem Komputer

Tujuan Sistem Operasi :

1) Mengendalikan eksekusi program-program aplikasi
2) Memudahkan penggunaan sistem komputer
3) Mengelola sumber daya sistem komputer agar dapat digunakan secara efisien Sistem operasi dianggap sebagai mediator pada sistem komputer, karena fungsi dari sistem operasi itu sendiri yakni memudahkan bagi pemrogram dan programprogram aplikasi untuk mengakses dan menggunakan fasilitas-fasilitas dan layanan-layanan. Sistem operasi juga dapat mempermudah pekerjaan seorang programmer karena menyembunyikan detail hardware dari pemrogram dan menyediakan interface yang nyaman untuk menggunakan sistem bagi pemrogram.

Sistem Operasi disebut manajer sumber daya komputer. Sistem operasi bertindak sebagai pengelola sumberdaya sistem komputer berdasarkan layanan yang dapat disediakannya, yakni akses terkendali ke file dan akses sistem bersama/publik, yang digunakan untuk pemindahan, penyimpanan dan pengolahan data serta mengendalikan fungsi-fungsi lain di dalam menggunakan sumberdaya sistem komputer.

Manfaat Sistem Operasi :

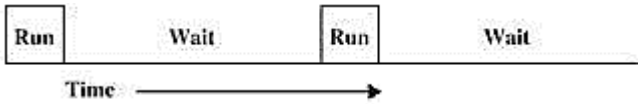
a. Kenyamanan : Suatu sistem operasi membuat komputer lebih mudah
b. digunakan, lebih mudah diperintah oleh pengguna (bukan programmer)

c. Efisiensi : Suatu sistem operasi memungkinkan sumber daya sistem komputer dapat digunakan dengan cara yang efisien.

d. Layanan Sistem Operasi Jenis Sistem Operasi :

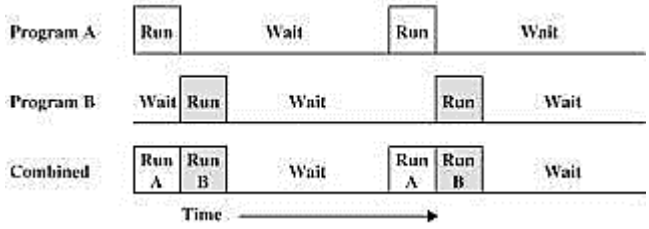
1. *Interactive* pemrogram/pengguna berinteraksi secara langsung dengan komputer melalui keyboard/layar monitor dalam meminta eksekusi tugas atau membentuk transaksi, selain itu pengguna dapat berkomunikasi dengan komputer selama proses eksekusi berlangsung.
2. *Batch* program penggguna ditampung bersama program pengguna lainnya dan kemudian diserahkan ke operator komputer, setelah program diselesaikan hasilnya dicetak.
3. *Single program (Uni-programming)*

   Prosesor akan mengerjakan sebuah program pada suatu saat. Hal ini akan mengakibatkan pemborosan waktu karena komputer tidak bisa menjalankan lebih dari satu program bersamaan (uniprogramming) single programming

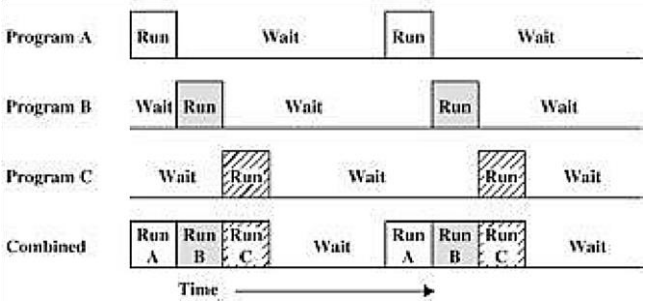


4. *Multi-programming (Multi-tasking)*

   Menjaga prosesor selalu berada dalam keadaan sibuk dengan membiarkan prosesor mengerjakan lebih dari satu program pada suatu saat, seolah-olah secara bersamaan. Dengan cara memuat beberapa program ke dalam memori, dan prosesor beralih dengan cepat dari satu program ke program yang lainnya. Hal ini akan membuat lebih hemat waktu. Akan tetapi jenis Sistem Operasi ini lebih boros resource (pada penggunaan CPU).



Multiprogramming dengan 2 program



Multiprogramming dengan 3 program

Sistem operasi tidak mengendalikan CPU, akan tetapi mengarahkan CPU dalam menggunakan sumberdaya sistem dan pewaktuan eksekusi program lainya. Ketika CPU menjalankan salah satu tugas tersebut di atas, sistem operasi berhenti dieksekusi dan CPU mengeksekusi program lainnya. Ketika CPU berhenti mengeksekusi sistem operasi, Sistem operasi tetap mengarahkan CPU dengan mengeluarkan kendali ke CPU, untuk melakukan bagian pekerjaan selanjutnya.

5. *Simple Batch Systems*

- Jaman dahulu (1940an s/d 1950an), komputer tanpa S/O, program (rangkaian proses)
- berinteraksi langsung dengan hardware melalui bahasa mesin
- Operator komputer ingin bisa memasukkan banyak program sekaligus tanpa harus ada operator standby: tiap satu program selesai, memasukkan program berikutnya.
- Dikembangkanlah simple batch system untuk mengatur jalannya program (inilah S/O awal)
- Bentuk modern: DOS
- Prosesor masa awal sangat mahal, dan oleh karena itu sangat penting untukmemaksimalkan pemanfaatan prosesor. Waktu yang disia-siakan dalam kaitan dengan waktu penjadwalan dan waktu setup sangat sulit untuk diterima. Untuk meningkatkan pemanfaatan dikembangkan sistem operasi batch sederhana.
- Batch artinya berurutan, sequentially queued, Batch operating system berbentuk sebuah program stay resident di memory.
- Program/job disusun dulu oleh user dalam kartu plong atau magnetic tape, kemudian diberikan kepada operator
- Operator memasukkan program ke sistem
- Tiap program/job dibaca oleh monitor, disimpan ke memory

- Program/job dijalankan oleh monitor sesuai dengan urutan masuknya (batched)

Komputer dengan sistem operasi yang bersifat interaktif mempunyai permasalahan ketika ada tambahan hardware dan software pada sistem komputer, antara lain :

1. Penjadualan, yakni pemesanan waktu secara blok untuk menyelesaikan sebuah program. Bila waktu yang dibutuhkan eksekusi program kurang dari blok waktu yang dipesan maka ada sisa waktu , demikian juga bila sebuah program dieksekusi dengan jatah waktu lebih dari blok waktu, maka program dipaksa untuk dihentikan.
2. Waktu setup, yakni sebuah pekerjaan/job meliputi pemuatan compiler dan program bahasa tinggi (progam sumber) ke dalam memori, penyimpanan program yang telah dikompilasi, dan membuat dan melakukan link program object dengan fungsi-fungsi program. Dimana, masing-masing langkah meliputi pemasangan dan penanggalan pita/tumpukan kartu memerlukan waktu dalam setiap pekerjaan setup program agar dapat berjalan. Permasalahan sistem operasi bersifat interaktif diatasi dengan menggunakan sistem operasi bersifat batch yang menempatkan monitor/pengamat program yang ditempatkan secara resident di memori utama dan selalu tersedia untuk melakukan eksekusi. Fungsi monitor/pengamat adalah membaca job satu per satu, menempatkan job pada program pengguna dan kendali diberikan ke job tersebut, bila job telah selesai, akan terjadi interup yang mengembalikan kendali ke monitor, yang segera membaca job berikutnya.

Fitur-fitur pada monitor :

1. Proteksi memori, yakni area memori yang berisi monitor yang tidak dapat diubah ketika program pengguna dieksekusi. Bila tejadi perubahan area memori, maka hardware CPU akan mendeteksi adanya error dan memindahkan kendali ke monitor.
2. Pewaktu digunakan untuk mencegah sebuah job memonopoli sistem. Bila waktu telah habis, terjadi interrupt dan kendali kembali ke monitor.
3. Privileged instruction, merupakan instruksi-instruksi tertentu yang ditandai khusus dan hanya dapat dieksekusi oleh monitor. Instruksi tersebut berupa instruksi I/O sehingga monitor menguasai kendali semua perangkat I/O

DISTRIBUTED FILE SYSTEM

7.1 Pengertian Sistem File Terdistribusi

Sistem file terdistribusi merupakan implementasi terdistribusi dari model time sharing klasik dari suatu sistem file dimana dalam manajemen sistem file terdistribusi sejumlah pengguna akan melakukan share file dan penyimpanan resource. File server pertama kali didevelop pada tahun 1970 dan Sun NFS (Network File System) menjadi DFS pertama yang banyak digunakan setelah awal pemunculannya di tahun 1985. DFS yang terkenal selain NFS adalah AFS (Andrew File System) dan CIFS (Common Internet File System).

Kelebihan lain dari DFS adalah peningkatan performa. Yang menjadi tolak ukur pengukuran performa DFS adalah waktu yang dibutuhkan untuk merespon request layanan.

Saling berbagi media penyimpanan informasi sudah menjadi sesuatu hal yang penting dalam resource sharing. Desain service file terdistribusi yang baik adalah menyediakan akses distribusi file dengan performansi dan realibilitas yang sama atau lebih baik dari penyimpanan file-file dalam disk local dalam bentuk transparent.

Desain skala besar dari proses sistem penyimpanan baca tulis file pada wide area menimbulkan masalah pada load balancing, reliabilitas, avaibility dan security. File system yang terdistribusi mengemulasikan fungsionalitas dari file sistem tak terdistribusi untuk program client yang berjalan pada komputer remote. File sistem terdistribusi juga menyediakan hal-hal pokok untuk pengorganisasian komputer yang berbasiskan jaringan intranet.

7.2 Tujuan Diterapkannya Sistem File Terdistribusi

Tujuan diterapkannya dari sistem file terdistribusi dalam manajemen proses adalah untuk mensupport sharing dengan tipe sama yang disebabkan karena file-file secara fisik tersebar pada halaman website yang berada dalam sistem terdistribusi.

Struktur sistem file terdistrbusi terdiri dari tiga bagian yaitu: service, server, dan client.

1. Service

   Entitas software yang berkerja pada satu untuk lebih mesin dan dilengkapi suatu tipe fungsi khusus untuk prioritas client yang tidak diketahui identitasnya.

2. Server

   Server adalah service software yang bekerja pada mesin tunggal.

3. Client

Client adalah suatu proses yang dapat memanggil suatu service dengan menggunakan sejumlah operasi yang dibentuk oleh interface client.

Keperluan sistem file terdistribusi :

1. Transpansi File service biasanya merupakan service yang harus di load paling berat dalam sebuah intranet, sehingga fungsionalitas dan performance_nya sangat penting.
   a. Transparansi akses
   b. Transparansi lokasi
   c. Transparansi mobilitas
   d. Transparansi performance
   e. Transparansi pengukuran
2. Update file konkuren Perubahan pada sebuah file oleh seorang klien seharusnya tidak menganggu operasi dari klien lain yang pada saat bersamaan mengakses atau mengubah file yang sama.
3. Replikasi file Beberapa file service mendukung penuh replikasi, tetapi kebanyakan mendukung caching file atau portion file secara lokal, bentuk replikasi yang terbatas.
4. Ke_heterogen_an sistem operasi dan hardware Antarmuka service sebaiknya didefinisikan sehingga software klien dan server dapat diimplementasikan untuk sistem operasi dan komputer yang berbeda.
5. Toleransi kesalahan Server bisa menjadi stateless, sehingga dapat di_restart dan service di_restore kembali setelah mengalami failure tanpa perlu me _recover state sebelumnya.
6. Konsistensi Ketika file_file direplikasi atau di cache pada site yang berbeda, ada delay yang tak bisa dihindari pada propagasi modifikasi dari satu site ke set lain yang membawa copy, dan ini bisa menghasilkan beberapa deviasi dari one copy semantic.
7. Keamanan Secara virtual, semua sistem file menyediakan mekanisme kontrol akses berdasarkan kegunaan dari daftar kontrol akses.
8. Efisiensi File service terdistribusi sebaiknya menawarkan fasilitas yang paling tidak, sama bagusnya dengan yang ditemukan pada sistem file konvensional, dan sebaiknya mendapat level performance yang dapat diperhitungkan.

7.3 Naming dan Transparansi

7.3.1 Naming

Naming adalah Pemetaan (pemisahan) antara obyek logika ke obyek fisik. Berfungsi untuk abstraksi lokasi file pada jaringan dan disk secara fisik. Misal **:** Penggunaan akan mengakses data yang secara logika direpresentasikan dengan nama file, sehingga sistem akan memanipulasi blok data yang secara fisik tersimpan dalam disk.

Transparansi adalah suatu dimensi baru yang ditambahkan untuk abstraksi, lokasi suatu file dalam jaringan.

Struktur Naming pada manajemen sistem file terdistribusi ada 2 jenis, yaitu:

1. Location Transparency (LT), adalah nama file tidak memberitahukan suatu petunjuk tentang lokasi penyimpanan file secar fisik. LT merupakan suatu skema naming yang statis (tetap), karena pengguna melakukan share data dengan cara konvensional.
2. Location Independence (LI), adalah nama file tidak perlu diubah jika lokasi penyimpanannya berubah. LI merupakan suatu skema naming yang bersifat dinamis, karena LI dapat melakukan pemetaan nama file yang sama ke lokasi yang berbeda pada waktu berbeda.

Perbedaan struktur naming LI dan LT pada sistem file terdistribusi :

1. Pemisahan Data dari Lokasi

   Nama file hanya berisi atribut signifikan (isi) dari pada lokasi. Pada LI, file dapat dipandang sebagai wadah data logika yang tidak dikaitkan ke lokasi penyimpanan tertentu. Sedangkan, jika pada sistem hanya ada LT statis saja yang di support, maka nama file tetap berisi kumpulan blok – blok secar fisik.

2. Share Data
   a. Pada LT, pengguna melakukan share data dengan cara konvensional. Pengguna dapat melakukan share file jarak jauh dengan naming sederhana dalam LT statis, sehingga file di anggap lokal. Tetapi sharing ruang penyimpanan sukar,sebab nama logikal secara satatis masih terhubungkan dengan penyimpanan fisik.
   b. Pada LI, pengguna melakukan share data dengan mempromosikan sharing ruang penyimpanan seperti obyek data. Bila file-file dapat dimobilisir, maka sistem ruang penyimpanan yang banyak dianggap sebagai sebuah penyimpanan tunggal, yang dikenal dengan sebutan “sumber virtual”.

3. Media Penyimpanan

   LI memisahkan hirarchi naming dari hirarchi media penyimpanan dan dari struktur inter computer, hal ini berlainan dengan menggunakan LT statis, kita dapat dengan mudah mengekspose korespodensi antar unit-unit komponen dan mesin. Dalam sistem client-server jika telah terjadi pemisahan nama & lokasi, maka file dalam server jarak jauh dapat di akses oleh banyak client.

7.3.2 Naming scheme

Dalam manajemen sistem file terdistribusi terdapat 3 pendekatan utama untuk naming scheme :

1. Secara sederhana
   a. Nama file merupakan kombinasi antar nama host dan nama lokasi.
   b. Cara ini tisak mengenal LT/LI.
   c. Operasi file yang sama dapat digunakan untuk file local atau jarak jauh.
   d. Komponen unit tetap terisolasi untuk menjaga tetap terpeliharanya file jarak jauh.
2. Pendekatan oleh Suns Network File Sistems
   a. Sistem ini di support oleh linux.
   b. Berusaha menghubungkan direktori jarak jauh dengan direktori lokal.
   c. Mensupport transparent sharing.
   d. Tidak ada batasan yang seragam tiap mesin, bisa menghubungkan direktori jarak jauh yang berbeda dengan pohon directorinya.
3. Pendekatan total integrasi komponen sistem file.

   Sebuah struktur nama global tunggal menyimpan semua file dalam sistem.

7.4 Remote File Access Antar Site

Pada manajemen sistem file terdistribusi ada 2 teknik untuk mengakses suatu file antar site pada jarak jauh, yaitu layanan jarak jauh (remote service) dan teknik caching.

7.4.1 Remote Service

1. Cara kerja : request dari client akan dikirim ke sever, lalu mesin server menjalankan akses dan hasilnya akan dikirimkan kembali ke client.

2. Dengan cara kerja seperti ini sangat membebani jaringan, karena jika setiap pengguna ingin mengakses suatu file server, maka sistem akan menjalankan akses tersebut walaupun file yang diakses beberapa pengguna itu sama.
3. Namun dari aspek konsistensian data maka teknik ini lebih menjamin.

7.4.2 Teknik Caching

1. Cara kerja : Akses dijalankan pada cached-copy, jika data yang dikehendaki untuk memenuhi request akses dari client belum siap di chace, maka copy dari data tersebut akan dibawa dari server kesistem pengguna. Tetapi bila blok disk yang diakses client ada dalam cache, maka akses berikutnya hanya bersifat local sehingga hal ini akan mengurangi traffic jaringan (Hampir sama dengan prinsip kerja proxy).
2. Suatu file berada pada mesin server, maka copynya dapat disebarkan pada cache. Jika ada perubahan pada cache, maka hanya sebagai file saja yang diubah pada mesin server.
3. Pada sistem file konvensional, caching mereduksi traffic disk I/O, sedangkan pada sistem file terdistribusi caching akan mereduksi traffic jaringan & traffic disk I/O.

Dalam manajemen sistem file terdistribusi ada 2 kebijaksanaan, yaitu:

1. Write-through adalah perubahan pada cache diikuti perubahan pada disk.
2. Wwrite back adalah perubahan pada cache tidak langsung diikuti perubahan pada disk, yaitu penulisan pada disk dilakukan jika pada cache terjadi tumpukan atau proses selesai. Hal ini kurang reliable, karena data bisa hilang. Dalam manajemen sistem file terdistribusi juga perlu memperhatian konsistensi file:
    a. Client dihadapkan pada masalah konsistensi data yang ada pada cache local terhadap copy master diserver.
    b. Jika client mengalami kegagalan akses dan mengakibatkan data cache out of date, maka client harus dilakukan up date data.

7.5 Pengertian File Service terdistribusi

File Sistem Terdistribusi (Distributed File System , disingkat) adalah file sistem yang mendukung sharing files dan resources dalam bentuk penyimpanan persistent di sebuah network. File server pertama kali didevelop pada tahun 1970 dan Sun NFS (Network File System) menjadi DFS pertama yang banyak

digunakan setelah awal pemunculannya di tahun 1985. DFS yang terkenal selain NFS adalah AFS (Andrew File System) dan CIFS (Common Internet File System).

Sebuah file server menyediakan file service ke client. Dari sisi client terdapat interface untuk file service dalam hal operasi primitif file, seperti membuat file (create), menghapus (delete) dan read / write file. Komponen perangkat keras utama yang mana file server mengontrolnya adalah sebuah local storage (umumnya disk drive / HDD). Ditempat itulah filefile tersimpan dan dari tempat tersebut request client meretrive file. Pada DFS client, server dan juga perangkat penyimpanan merupakan mesin terpisah dalam sebuah lingkungan terdistribusi (Intranet).

7.5.1 Layanan File Terdistribusi

1. Layanan Dasar
    - Tempat penyimpanan tetap untuk data dan program
    - Operasi terhadap file (create, open, read,…)
    - Multiple remote clients (dalam intranet)
    - File sharing
    - Menggunakan semantic one-copy update umum, melalui RPC
2. Perkembangan baru
    - Persistent object stores (storage of objects)
3. Persistent Java, Corba, …
    - Replikasi, caching keseluruhan file
    - Multimedia terdistribusi (contoh: file server Tiger video)

7.5.2 Keperluan sistem file terdistribusi

1. Transpansi

   File service biasanya merupakan service yang harus di load paling berat dalam sebuah intranet, sehingga fungsionalitas dan performance nya sangat penting.

    a. Transparansi akses
    b. Transparansi lokasi
    c. Transparansi mobilitas
    d. Transparansi performance
    e. Transparansi pengukuran

2. Update file konkuren

   Perubahan pada sebuah file oleh seorang klien seharusnya tidak menganggu operasi dari klien lain yang pada saat bersamaan mengakses atau mengubah file yang sama.

3. Replikasi file

   Beberapa file service mendukung penuh replikasi, tetapi kebanyakan mendukung caching file atau portion file secara lokal, bentuk replikasi yang terbatas.

4. Ke heterogen an sistem operasi dan hardware

   Antarmuka service sebaiknya didefinisikan sehingga software klien dan server dapat diimplementasikan untuk sistem operasi dan komputer yang berbeda.

5. Toleransi kesalahan

   Server bisa menjadi stateless, sehingga dapat di restart dan service di_restore kembali setelah mengalami failure tanpa perlu me recover state sebelumnya.

6. Konsistensi

   Ketika file file direplikasi atau di_cache pada site yang berbeda, ada delay yang tak bisa dihindari pada propagasi modifikasi dari satu site ke set lain yang membawa copy,an ini bisa menghasilkan beberapa deviasi dari one_copy semantic.

7. Keamanan

   Secara virtual, semua sistem file menyediakan mekanisme kontrol akses berdasarkan kegunaan dari daftar kontrol akses.

8. Efisiensi

   File service terdistribusi sebaiknya menawarkan fasilitas yang paling tidak, sama bagusnya dengan yang ditemukan pada sistem file konvensional, dan sebaiknya mendapat level performance yang dapat diperhitungkan.

7.5.3 Opsi Perancangan Layanan File

1. Stateful
   - server menyimpan informasi tentang file yang open, posisi sekarang (current position) dan file locks
   - open (dibuka) sebelum access dan kemudian ditutup
   - performa yang lebih baik

- pesan yang lebih pendek, dimungkinkan untuk read-ahead
- server failure
- kehilangan state
- client failure - tables fill up
- menyediakan file locks

2. Stateless
   - server tidak menyimpan state informasi
   - file operations idempotent, harus mengandung semua yang diperlukan (longer message)
   - perancangan file server yang lebih simple
   - dapat dengan mudah di-recovery apabila client ataupun server crash
   - locking membutuhkan extra lock server untuk mempertahankan

7.5.4 File Service Architecture

Pembagian tanggung jawab antar modul didefinisikan sebagai berikut ini :

- Layanan file flat
- Layanan file flat berkonsentrasi pada pengimplementasian operasi dari konten suatu file.
- Layanan direktori
- Layanan direktori menyediakan pemetaan antara nama teks untuk file dan UFID • Modul klien
- Modul klien berjalan pada tiap komputer klien, mengintegrasi dan meng_extend operasi dari layanan file flat dan layanan direktori dibawah antarmuka pemrograman aplikasi tunggal yang bisa digunakan oleh program tingkat pengguna di komputer klien.
- Antarmuka layanan file flat
- Merupakan antarmuka RPC yang digunakan oleh modul klien. Tidak digunakan secara langsung oleh program tingkat pengguna.

NAME SERVIS

8.1 Definisi Name Service Pada Sistem Terdistribusi

Name Service pada Sistem Terdistribusi adalah merupakan layanan penamaan yang berfungsi untuk menyimpan naming context, yakni kumpulan binding nama dengan objek, tugasnya untuk me-resolve nama. Dalam sistem terdistribusi, nama digunakan untuk menunjuk ke suatu sesumber yang beragam dan tersebar seperti komputer, layanan (services), file, dan remote object.

Pengaksesan resource pada sistem terdistribusi memerlukan:

a Nama resource (untuk pemanggilan),

b Alamat (lokasi resource tsb),

c Pemetaan antara nama dan alamat.

Manfaat dari Name Service adalah :

a. Komunikasi: Nama domain sebagai bagian dari email.

b. Resource Sharing: Nama domain internet. Proses tidak dapat mengakses suatu sesumber, jika sesumber tersebut tidak diberi nama.

c. Location Independence: Perubahan lokasi tidak menuntut perubahan nama, asalkan lokasi tidak menjadi bagian dari nama resource tsb.

d. Security: Jika sebuah nama dipilih secara acak dari himpunan besar interger, maka nama tsb hanya bisa diketahui dari legitimate source, bukan dari menebak. Jadi jika seseorang mengetahui nama obyek tsb, maka dia memang diberitahu, karena sulit sekali menebak nama tsb.

Kebutuhan terhadap Name Service adalah :

a Penamaan unik yang standard

b Scalability

c Consistency

d Performance dan Availability

e Mudah menyesuaikan terhadap perubahan perlindungan kegagalan.

Jenis Name Service adalah :

- Pure name: nama yang tidak perlu di terjemahkan, karena pada nama tersebut sudah menunjuk alamat objek langsung.

Contoh : IP.

- Non-pure name: dalam nama mengandung suatu informasi (atribut misalnya) tentang suatu objek.

  Contoh : URL, alamat email, X.500 Directory Service, IOR (Interoperability Object Reference).

Name Service memiliki konsentrasi pada aspek penamaan dan pemetaan antara nama & alamat, bukan pada masalah rute, yang dibahas di Jaringan Komputer. Resource yang dipakai dalam Name Service adalah: komputer, layanan, remote object, berkas, pemakai.

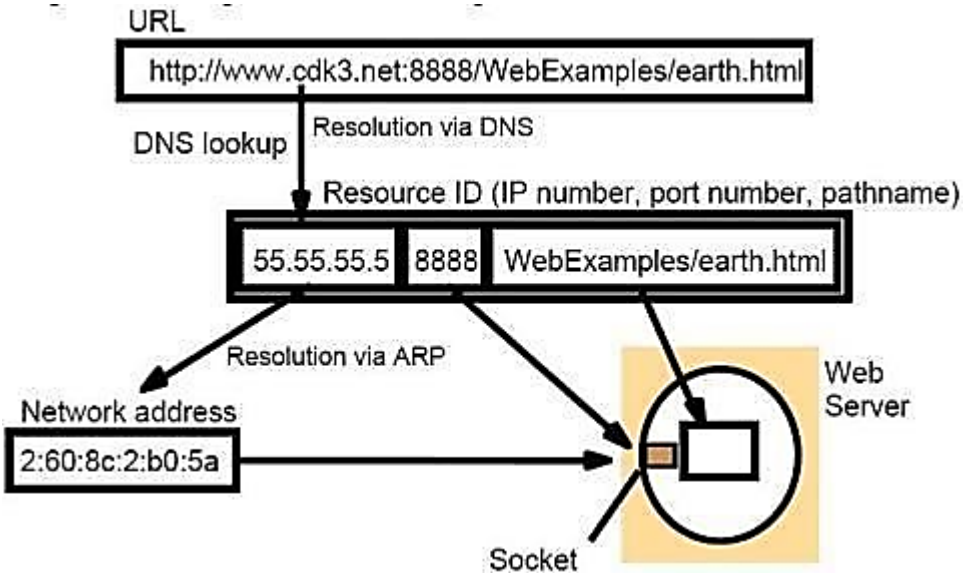
Contoh penamaan pada aplikasi sistem terdistribusi:

- URL untuk mengakses suatu halaman web.
- Alamat e-mail utk komunikasi antar pemakai.

8.1.1 Name Resolution, Binding, Attributes

- Name resolution: Nama ditranslasikan ke data ttg resource/object tsb.
- Binding: Asosiasi antara nama & obyek, dan biasanya nama diikat (bound) ke attributes dr suatu obyek.
- Address: atribut kunci dari sebuah entitas dalam sistem terdistribusi.
- Attribute: nilai suatu object property.

8.1.2 Penguraian Naming Domains untuk mengakses resource dari URL



8.2 Tujuan Penamaan

1. Mengidentifikasi
2. Memungkinkan terjadinya sharing
3. Memungkinkan location independence
4. Memberikan kemampuan keamanan (security)

8.3 Jenis Nama

1. User names: Merujuk pada suatu obyek atau layanan; Terdiri dari strings of characters.

   Contoh: hp201 untuk pencetak, ~bettyp/tmp/test.c untuk berkas.

2. System names: Terdiri dari bit string; Internal untuk sistem, tidak ditujukan untuk manusia.

8.4 Struktur Nama

1. Primitive/flat names (Unique Identifiers = UIDs)
2. Partitioned Names (PN)
3. Descriptive names (DN)

8.5 Name Context

Nama selalu diasosiasikan dengan konteks, yang mendefinisikan di mana nama tsb valid. Ada 2 macam konteks:

1. Universal context
2. Relative context

8.6 Name List

Name Lists terdiri dari 2 komponen yaitu:

1. Name agents
2. Name servers

8.7 Bentuk Name List

1. Name List Tersentralisasi : Adalah Name list yang berada pada satu mesin.
2. Name List Tereplikasi Penuh : Digunakan untuk mengatasi kekurangan name list tersentralisasi.

3. Name List Tereplikasi Sebagian : Sebagian name lists disimpan dalam cache setiap mesin dan memerlukan mekanisme petunjuk (hint), yang biasanya benar.

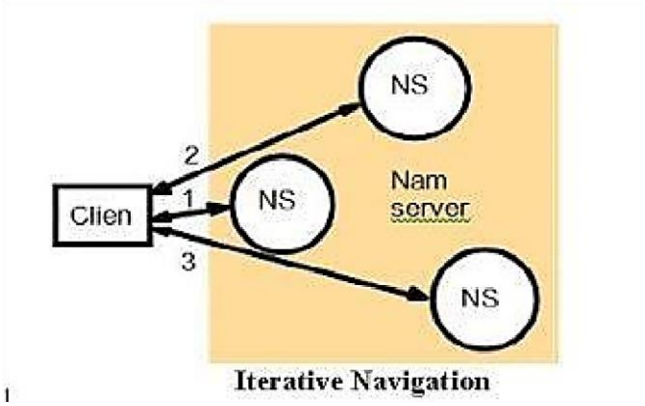
8.8 Contoh Name Service

1. DNS (Domain Name Service) – memetakan nama domain ke alamat
2. GNS (Global Name Service) – memetakan global name ke atribut-atribut dan skalabilitas, dapat menangani perubahan
3. X500 directory service – memetakan nama orang ke dalam alamat suatu e-mail dan nomor telepon
4. Jini discovery service – mencari objek sesuai dengan atribut yang ada Dalam rangka memenuhi kebutuhan tersebut, sebuah name server setidaknya dapat menerapkan mekanisme berikut :
    a. Partitioning : Tidak ada satu name server yang dapat menyimpan seluruhnama dan atribut untuk seluruh jaringan. Data nama dipartisi berdasarkan domain.
    b. Replication : Sebuah domain biasanya memiliki lebih dari satu name server. Untuk meningkatkan availability dan performance.
    c. Caching : Sebuah name server dapat melakukan mekanisme caching terhadap data nama dari name server lain. Hal ini dilakukan untuk mencegah operasi permintaan sama berulang ulang.

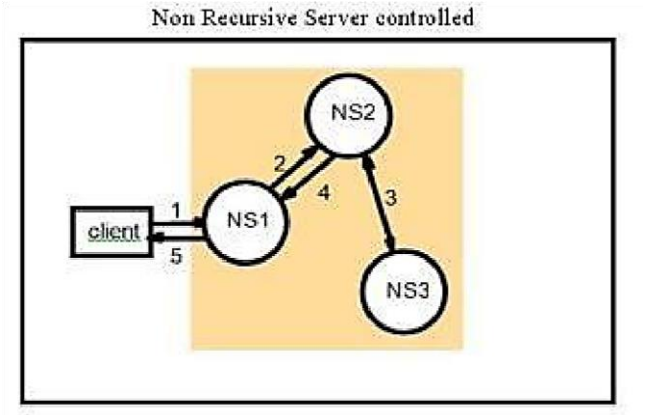
8.9 Name servers and navigation

Navigasi dan name servers di sini memiliki arti petunjuk pengaksesan nama data dari lebih dari satu name server untuk menyelesaikan suatu pemetaan nama (*resolve a name*). Yang didalamnya terdapat 3 metode, yaitu :

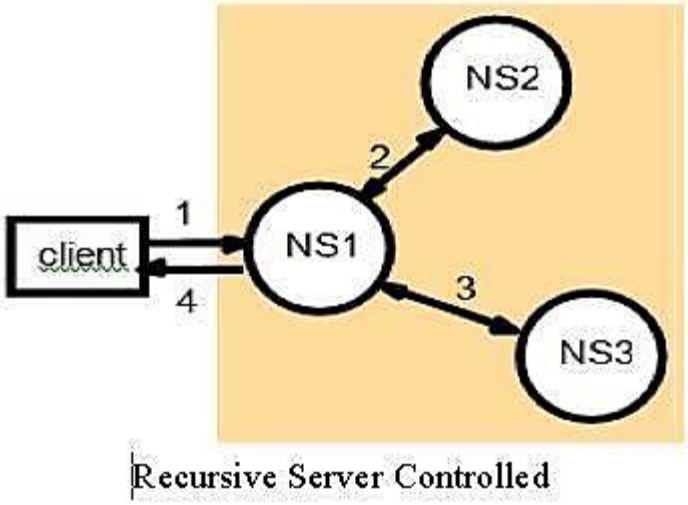
1. Iterative Navigation



2. Non-recursive, server-controlled



3. Recursive, server-controlled;



Kebutuhan akan penamaan tersebut memicu munculnya layanan penamaan (*Naming Services*) yang menyediakan mekanisme dan struktur penamaan objek itu sendiri. Contoh : DNS (Domain Name Service). Dan juga kebutuhan untuk pencarian objek berdasar nama dan juga atribut objek itu sendiri (*Directory Service*). Suatu nama akan diterjemahkan ke dalam suatu data tentang sesumber atau objek yang dimaksudkan. Gabungan antara objek dan nama disebut *binding*. Dalam nama objek terdapat beberapa *atribut* yang merupakan properti suatu objek. Contoh :

- DNS : memetakan dari nama ke atribut alamat IP host
- X.500 : memetakan suatu nama seseorang ke beberapa atribut, seperti email, telepon, dsb.
- CORBA Naming Service yang memetakan nama *remote objek* ke *remote object reference*.

8.10 Jenis Nama

1. User names: Merujuk pada suatu obyek atau layanan; Terdiri dari strings of characters.
2. Contoh: hp201 untuk pencetak, ~bettyp/tmp/test.c untuk berkas.
3. System names: Terdiri dari bit string; Internal untuk sistem, tidak ditujukan untuk manusia.

8.11 Ancaman terhadap Nama Layanan

Nama layanan adalah target serangan potensial pada mobile agent infrastruktur mendatang. DoS (Denial of Service) menyerang pada koneksi jaringan server nama yang sangat mengganggu karena semua agen dilayani oleh itu dipengaruhi simulta-neously.

Ini salah satu model dari name service, contohnya ada linux, oracle, corba dll. 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). Beberapa contoh perangkat pengembangan berbasis CORBA yang berjalan di Linux antara lain: MICO dari mico.org (bahasa yang didukung: C++), Fnorb dari DSTC, Australia (Python), JacORB oleh Gerard Brose dari Freie Universitat, Berlin (Java), OmniORB2 dari AT&T (C++), serta tak ketinggalan pula ORBit keluaran laboratorium riset RedHat (mendukung bahasa C) yang dipakai dalam proyek Gnome. CORBA dirancang oleh Object Management Group (OMG) untuk terutama memberikan berorientasi objek interoperabilitas aplikasi dalam sistem terdistribusi heterogen. Itu penggunaan Object-Oriented desain, analisis, dan pengembangan

menggunakan CORBA memungkinkan lebih besar usabilitas di sistem. Keuntungan dari Berorientasi Objek fitur seperti warisan, enkapsulasi, redefinisi dan dinamis mengikat diimplementasikan dalam CORBA. Obyek- fitur berorientasi objek CORBA juga efektif lebih mudah untuk memperluas dan memodifikasi tanpa mempengaruhi aplikasi lain dan objek. CORBA merangkum aplikasi dan menyediakan infrastruktur umum untuk komunikasi menggunakan ORB CORBA dan digunakan untuk menerima permintaan dan menemukan server objek. CORBA mendorong pengembangan aplikasi yang terbuka, aplikasi yang dapat terintegrasi dengan sistem yang lebih besar. Kelebihan CORBA juga meliputi: transparansi lokasi, bahasa pemrograman transparansi, Sistem Operasi transparansi dan transparansi Komputer

Dalam satu sistem, Layanan Penamaan diimplementasikan menggunakan mendasari perusahaan- lebar penamaan server seperti CDS DCE. Layanan Penamaan digunakan untuk membangun besar, perusahaan-lebar penamaan grafik mana NamingContexts model "direktori" atau "Folder" dan nama lain mengidentifikasi "dokumen" atau "file" jenis objek. Di lain katakata, layanan penamaan digunakan sebagai tulang punggung sebuah sistem pengarsipan perusahaan-lebar.

8.12 Model Penamaan Layanan

Penamaan adalah gudang yang referensi nama toko objek. Layanan Penamaan digunakan untuk terletak objek dalam CORBA. Layanan Penamaan dapat digunakan untuk mengambil objek referensi. Operasi layanan dapat dipanggil, melalui referensi objek. Penamaan mendukung dua operasi dasar: toko (mengikat) dan mengambil (menyelesaikan). Layanan mengiklankan ketersediaan mereka untuk Layanan Penamaan dengan referensi objek mereka nama.

8.13 Keuntungan dari Layanan Penamaan

Penamaan mendukung lingkungan didistribusikan. Penamaan Service adalah layanan standar dan implementasi yang platform independen.

8.14 Kekurangan Penamaan Layanan

Layanan penamaan tidak tersedia untuk semua implementasi, tetapi dapat dengan mudah diimplementasikan jika diperlukan.

DISTRIBUTED ALGORITHMS

9.1 Algoritma Koordinasi Ikhtisar :

1. Koordinasi di dalam sistem terdistribusi mengapa [yang] diperlukan, sumber permasalahan
2. Pengeluaran timbal balik
    - ring-based
    - multicast & waktu logis [Ricart Dan Agrawala]
3. Pemilihan Pemimpin
    - ring-based [Chang&Roberts]
    - secara acak [Itai&Rodeh]
    - pohon identifikasi [FIREWIRE IEEE 1394]

Algoritma Koordinasi adalah pokok membagi-bagikan sistem:

1. karena sumber daya yang terbagi: [yang] bersamaan memperbaharui
    - arsip di (dalam) suatu database ([record/ catatan] mengunci)
    - file (kunci file di (dalam) file server tak berkewarganegaraan)
    - suatu buletin bersama yang diberitahu
2. untuk bermufakat melakukan tindakan: apakah untuk
    - commit/abort transaksi database
    - bermufakat untuk melakukan suatu pengenalan dari suatu kelompok sensor
3. untuk dengan dinamis re-assign peran guru
    - memilih server waktu utama setelah roboh/hancur
    - memilih koordinator setelah bentuk/ wujud jaringan kembali

Kenapa sulit?

- Solusi Centralised tidak sesuai komunikasi bottleneck
- Master-Slave yang ditetapkan memperbaiki Pengaturan yang tidak sesuai proses kehancuran
- Macam-Macam topologi jaringan [cincin/arena], pohon, sewenang-wenang dan permasalahan yang saling berhubungan
- Jika mungkin, kegagalan harus dimaklumi kegagalan mata rantai proses kehancuran

Ketidakmungkinan akan menghasilkan

– kegagalan di dalam kehadiran, esp model yang tak serempak

- Pengeluaran timbal balik
  a. format yang didistribusikan dari bagian permasalahan yang kritis
  b. harus menggunakan pesan yang telah lewat
- Pemilihan Pemimpin
  a. setelah roboh/hancur kegagalan telah terjadi
  b. setelah bentuk/wujud jaringan kembali

- Konsensus (juga disebut Persetujuan): yaitu ceramah kuliah berikutnya
  a. serupa untuk mengkoordinir serangan
  b. beberapa berdasarkan pada multicast komunikasi
  c. varian yang tergantung pada jenis kegagalan, jaringan, dll

Asumsi Kegagalan mengasumsikan mata rantai yang dapat dipercaya, tetapi prosesnya mungkin mengalami kehancuran.

- [Jasa;Layanan] Pendeteksian Kegagalan
  – menyediakan query jika suatu proses telah gagal – bagaimana?
- proses mengirimkan „ P di sini‟ merupakan pesan T per tiap-tiap detik
- jawaban dari arsip detektor kegagalan
  – tak dapat dipercaya, terutama di dalam sistem yang tak serempak

• Pengamatan atas kegagalan:

- Yang dicurigai: tidak ada komunikasi terbaru, tetapi bisa melambat
- Yang tak dicurigai: tetapi tidak ada jaminan bahwa sejak itu belum digagalkan
- Yang digagalkan: hancurnya telah ditentukan

Pengeluaran timbal balik yang dibagi-bagikan

• Masalah:

- N proses yang tidak sama, karena kesederhanaan tidak mempunyai kegagalan
- penyerahan pesan yang dijamin (mata rantai yang dapat dipercaya)
- untuk melaksanakan bagian kritis (CS), yang masing-masing proses tersebut adalah:
  - masuk()
  - resourceAccess()

- keluar()
- Kebutuhan (MC1) Paling banyak ada satu proses dalam CS yang dilakukan pada waktu yang sama.

(MC2) Permintaan untuk masuk dan keluar harus secepatnya dilakukan. (MC3- Opsional, lebih kuat) Permintaan untuk masuk dilakukan menurut pesanan yang mempunyai hubungan sebab akibat.

Pemusatan dalam pengeluaran yang berhubungan timbal balik Pemusatan dalam pelayanan?

- Server tunggal mengimplementasi tanda khayal
  - hanya proses memegang tanda itu yang dapat dilakukan CS
  - server menerima permintaan untuk tanda
  - akses akan mengabulkan jawaban jika CS cuma-cuma; atau dengan cara meminta queued
  - ketika suatu proses melepaskan tanda, permintaan paling tua dari antrian akan dikabulkan
- meskipun demikian...
  - tidak menghormati order yang memiliki hubungan sebab akibat dari permintaan (MC3) mengapa?
- tetapi
  - server adalah pencapaian dari bottleneck!
  - apa akibatnya jika server tersebut hancur?

Algoritma Ring-based prosesnya disusun didalam suatu cincin logis, yang membiarkan mereka melewati tanda.

9.2 Algoritma Ring-based

A. Tidak ada server bottleneck, tidak ada guru Proses:
   - secara terus menerus tanda lewat di sekitar [cincin/arena], dalam satu arah
   - jika tidak memerlukan akses untuk CS, lewat ke atas tetangga
   - cara lainnya, nantikan tanda dan pertahankannya selagi dalam CS
   - untuk pergi, memberikan . pada tetangga

B. Bagaimana [itu] bekerja
   - penggunaan lebar jalur jaringan berlanjut

- menunda untuk masuk tergantung pada ukuran [cincin/arena] – order yang memiliki hubungan sebab akibat dari permintaan yang tidak sesuai (MC3) - mengapa?

Ricart & Agrawala Algoritma

A. Multicast yang didasarkan pada Komunikasi
   - proses inter-connected N yang tidak terjadi bersamaan, dengan masing-masingnya
     - identitas yang unik
     - waktu Lamport‟S yang logis
   - proses multicast untuk meminta masuk:

B. timestamped dengan waktu Lamport‟S dan proses identitas
   – memasuki untuk dikabulkan
     - ketika semua proses yang lain menjawab
     - permintaan bersama memecahkan dengan timestamp
     - Bagaimana [itu] bekerja
   - membuat baik [properti/milik] yang lebih kuat (MC3)
   - jika perangkat keras mendukung untuk multicast, hanya satu pesan yang masuk

Ricart&Agrawala Algoritma Pada status initialisasi:= YANG DILEPASKAN; Untuk memasuki status bagian yang kritis:= YANG DIINGINKAN; Multicast meminta kepada semua proses; proses permintaan yang ditunda T:= request‟s timestamp; Menunggu sampai (jumlah jawaban yang diterima= (N– 1)); status:= YANG DIPEGANG; Sesudah menerima suatu permintaan < Ti, Pi (22:7)> pada pj (i ?? j) jika(tate = HELD) atau ((state = WANTED) dan ((T, pj) < (Ti, pi)) then queue request from pi without replying; else reply immediately to pi; end if To exit the critical section state := RELEASED; reply to any queued requests; kemudian menunngu permintaan dari pi (22:7) tanpa menjawab; seketika jawaban itu akan muncul ke pi (22:7); berakhir; jika status bagian yang kritis keluar:= YANG DILEPASKAN; menjawab setiap permintaan queued; Multicast pengeluaran timbal balik p1, p2 meminta akses [yang] secara bersamaan, p3 tidak dapat melakukan pengaksesan. p2 tidak menjawab ke p1 karena mempunyai timestamp yang rendah Ringkasan Pengeluaran timbal balik

- ❖ Penampilan
  - satu sudah cukup untuk masuk
  - pemakaian lebar jalur jaringan yang tinggi
  - penundaan klien tergantung pada frekwensi akses dan ukuran jaringan
- ❖ Toleransi pada Kesalahan

- pada umumnya mengasumsikan mata rantai yang dapat dipercaya
- Beberapa, dapat beradaptasi untuk berhadapan dengan kehancuran

❖ Lain solusi

– cukup untuk memperoleh persetujuan dari overlap subsets tertentu dari proses yang memilih di-set (Maekawa Algoritma)

9.3 Algoritma Pemilihan Pemimpin

❖ Masalah

- Proses N, boleh atau tidak boleh mempunyai id yang unik (UIDS)
- untuk kemudahan mengasumsikan tidak ada apapun yang hancur
- harus memilih koordinator guru unik selama proses
- pemilihan dilakukan setelah kegagalan telah terjadi
- satu atau lebih proses dapat melakukan pemilihan yang secara bersamaan

❖ Kebutuhan (LE1) Tiap-Tiap proses mengetahui P, identitas pemimpin, di mana P adalah orang yang tak punya proses id yang unik (pada umumnya maksimum) atau yang tak tergambarkan. (LE2) Semua proses mengambil bagian dan secepatnya menemukan identitas pemimpin (tidak bisa tak tergambarkan). Chang&Roberts Pemilihan Pemimpin algoritma di dalam arena: model tidak bersamaan, yang dikenal sebagai UIDS. Chang&Roberts algoritma

❖ Asumsi

– cincin arena yang searah, tak serempak, masing-masing proses mempunyai UID

❖ Pemilihan

– pada awalnya masing-masing proses bukanlah peserta

– menentukan pemimpin (pesan pemilihan):

- pemrakarsa menjadi peserta dan melewati kepunyaan UID kepada tetangga
- ketika tidak peserta menerima pesan pemilihan, teruskan dengan maksimum pada kepunyaan dan UID yang diterima dan menjadi peserta
- peserta tidak meneruskan pesan pemilihan itu
- mengumumkan pemenang (pesan yang dipilih):
- ketika peserta menerima pesan pemilihan dengan kepunyaan UID, menjadi pemimpin dan bukan peserta, dan lanjutkan UID dalam pesan yang dipilih
    - cara lainnya, arsip leader‟s UID, menjadi bukan peserta dan lanjutkan itu

Chang&Roberts algoritma

• Bagaimana [itu] bekerja

- jika UIDS, kemudian identitas dari pemimpin yang unik
- dua pertukaran di sekitar [cincin/arena]: pemilihan, mengumumkan pemenang
- jika satu proses memulai pemilihan
  - dalam kasus yang terburuk 3 round-trips akan diperlukan- jelaskan?
  - tetapi– tidak memaklumi kegagalan (memerlukan detektor kegagalan yang dapat dipercaya)
  - lihat algoritma yang terbaik (synchronous model)
  - bekerja jika lebih dari satu proses [yang] secara bersamaan memulai pemilihan–apa akibatnya jika tidak (ada) UIDS?
  - persetujuan pada suatu jaringan re-configurable, „ hot-pluggable"

Itai&Rodeh Algoritma

• Asumsi

- Proses N, cicin arena tidak searah, synchronous
- proses tidak mempunyai UIDs

• Pemilihan

- setiap proses memilih ID secara acak dari set { 1,..K}
- proses melewati semua IDs di sekitar [cincin/arena]
- setelah satu putaran, jika di sana ada suatu ID yang unik kemudian pilihlah ID unik yang maksimum
- cara lainnya, mengulangi

• Pertanyaan

– bagaimana cara mengakhiri putarannya?

Itai&Rodeh Algoritma

- Bagaimana cara mengetahui algoritma berakhir?
  a. dari kemungkinan: jika kamu melemparkan suatu koin dengan adil kemudian setelah beberapa kepala- kamu harus mendapatkan ekor
  b. Berapa banyak putaran yang dikerjakan?
     - kemungkinan yang paling besar dari ID yang unik, yang lebih cepat algo

– waktu yang diharapkan: N=4, K=16, mengharapkan 1.01 putaran

c Mengapa menggunakan randomisasi?

– perusak/interuptor simetri

– solusi cepat

d tetapi hanya probabilistic (dengan kemungkinan 1) jaminan untuk berhenti FIREWIRE

IEEE 1394 pohon mengidentifikasi

• Asumsi

– tak satu [cincin/arena] pun : jaringan acyclic

– hot-pluggable, maka tidak (ada) UIDS selama pohon mengidentifikasi – harus memilih suatu koordinator (untuk bertindak sebagai bus manajer) setelah masing-masing keputusan ditambahkan/ diubah

• Bagaimana caranya bekerja?

– jaringan acyclic- hampir suatu pohon

– mulai dengan daun-daun: mengirimkan „ jadilah orangtua ku‟ pesan ke tetangga

– tangkai pohon yang menerima N-1 Permintaan (N adalah banyaknya tetangga), mengirimkan „ jadilah orangtua ku‟ kepada tangkai pohon yang sisanya

• Pertanyaan: akankah itu menentukan akarnya? Akar Gagasan Permasalahan

• Masalah

– dua tangkai pohon boleh secara bersamaan mengirimkan pesan „ jadilah orangtua ku‟ Untuk satu sama lain!

• Solusi

– masing-masing penantang menghempaskan suatu koin, kemudian memilih memilih penundaan yang pendek/singkat atau lama

– setelah penundaan, jika itu telah menerima „ jadilah orangtua ku‟ itu menjadi akar

– cara lainnya, mengirimkan kembali „ jadilah orangtua ku‟

– jika permasalahan timbul kembali, proses mengulangi

Ringkasan

• Algoritma yang didistribusikan

– lebih peka untuk terjadi kehancuran

- pesan yang berdasarkan pada yang disampaikan
  - memakai lebar jalur jaringan
  - Randomisasi
- bermanfaat sebagai perusak/interuptor simetri
- menghasilkan algoritma lebih cepat dan lebih sederhana – tetapi penghentian probabilistic terjamin

9.4 Model Sinkronisasi dan Asinkronisasi

Sinkronisasi adalah Adalah satu kunci kerja dari komunikasi data. Transmiter mengirimkan pesan 1 bit pada satu saat melalui medium ke receiver. Receiver harus menandai awal dan akhir blok dari bit, juga harus diketahui durasi untuk masing-masing bit sehingga dapat sample lajur dari timing untuk membaca masing-masing bit (merupakan tugas dari timming). Contoh : jika ada perbedaan misalkan 1 % (clock receiver 1% lebih lambat atau lebih cepat daripada clock transmitter), maka pada pensamplingan pertama akan meleset dari tengah bit dan setelah jumlah waktu tertentu, akan mengalami error.

Sinkronisasi di bagi menjadi 2, yaitu :

1. Asynchronous

   Untuk mencegah problem timming dengan tidak mengirim aliran bit panjang yang tidak putus putusnya. Bit-bit dikirim per-karakter pada setiap waktu yang mana masing-masing karakter mempunyai panjang 5-8 bit. Timing atau synchronisasi harus dipertahankan antara tiap karakter; receiver mempunyai kesempatan untuk men-synchron-kan awal dari tiap karakter baru.

   - Idle (biasanya =‟1‟) jika tidak ada karakter yang ditransmisikan dan start bit = “0”, sedangkan jumlah karakter yang ditransmisikan antara 5-8 bit.
   - Bit paritas digunakan untuk mendeteksi error, diatur oleh pengirim agar jumlah total „1‟ termasuk bit paritas adalah genap, dan stop bit = „1‟, yang panjangnya 1; 1,5; 2 kali durasi bit pada umumnya
   - Komunikasi asinkron adalah sederhana dan murah, tetapi memerlukan overhead dari 2 ke 3 bit per karakter, prosentasi overhead dapat dikurangi dengan mengirimkan blok-blok bit besar antara bit start dan bit stop

2. Synchronous / timing

Lebih efisien, karena blok-blok karakter / bit-bit ditransmisikan tanpa kode start dan stop, tetapi tiap blok blok dimulai dengan suatu pola preamble bit dan diakhiri dengan pola postamble bit. Pola-pola ini adalah kontrol informasi.Waktu kedatangan dan keberangkatan untuk masing-masing bit dapat diramalkan. Frame adalah data plus kontrol informasi. Format framenya tergantung dari metode transmisi, yaitu:

1. Transmisi orientasi karakter
   - Blok-blok data dikerjakan sebagai barisan karakter (biasanya 8 bit karakter), frame dimulai dengan 1 atau lebih karakter sinkronisasi. Karakter sinkronisasi biasanya disebut dengan “SYN” yang merupakan bit pattern unik sinyal yang diterima penerima permulaan dari blok.
   - Penerima kemudian merubah blok-blok data yang datang oleh karakter SYN dan menerima data sampai karakter postamble (informasi yang terletak pada bagian belakang blok data yang dikirimkan) terlihat dan begitu seterusnya
2. Transmisi bit.
   - Blok-blok data dikerjakan sebagai barisan bit-bit, tidak ada data maupun informasi kontrol diperlukan untuk menginter-prestasikan dalam satuan karakter 8 bit

9.4.1 Perbandingan asinkron dan sinkron

A. Untuk blok-blok data yang cukup besar, transmisi sinkronisasi jauh lebih efisien daripada asinkron. Transmisi asinkron memerlukan overhead 20 % atau lebih.

B. Bila menggunakan transmisi sinkron biasanya lebih kecil dari 1000 bit, yang mengandung 48 bit kontrol informasi (termasuk flag), maka untuk pesan 1000 bit, overheadnya adalah 48 / 1048 X 100% = 4.6% Urutan pengerjaan sinkronisasi yaitu :
   - Sinkronisasi bit: Ditandai awal & akhir untuk masing-masing bit
   - Sinkronisasi karakter / kata: Ditandai awal dan akhir untuk masing-masing karakter / satuan kecil lainnya dari data
   - Sinkronisasi blok / pesan: Ditandai awal dan akhir dari satuan besar data. Dan untuk pesan yang besar, dibagi-bagi menjadi beberapa blok kemudian baru dikirimkan pengurutan blok-blok yang telah dibagi tersebut adalah tugas dari timming. Sedangkan pengaturan level sinyal adalah tugas dari sintax dan untuk melihat arti dari pesan adalah tugas dari semantik.
   - Sinkronisasi yang terjadi pada CPU yaitu:

1. Merupakan salah satu cara komputer mengkoordinir kegiatannya dengan peralatan input dan ouput yang dihubungkan kepadanya. Pada program-controlled Input dan Ouput:
2. Prosesor terus-menerus memeriksa status flag setiap peralatan Input dan Ouput untuk mencapai sinkronisasi antara prosesor dan Input dan Ouput device.

o **Interrup**

Digunakan untuk meningkatkan kinerja processor dalam penanganan peralatan Input dan Ouput yang dimilikinya. CPU tidak harus selalu mengecek status peralatan Input dan Ouput. Bila terdapat peralatan Input dan Ouput yang meminta pelayanan, maka peralatan tersebut akan mengirim sinyal interrupt kepada CPU. Sinyal interrupt ini disebut juga sebagai Interrupt Request. Sinyal interrupt dikirim melalui jalur kontrol bus yang disebut interrupt request line. Processor akan merespon interrupt yang diterima dengan menghentikan proses yang sedang berjalan dan mengecek status peralatan Input dan Ouput untuk mencari peralatan yang meminta interrupt. Selanjutnya processor akan menjalankan suatu program untuk merespon interrupt yang diterimanya. Program ini disebut sebagai interrupt service routine (ISR). Berikut contoh processor dalam menjalankan ISR untuk merespon interrupt pencetakan ke printer.

9.4.2 Sinkronisasi dan Asinkronisasi

Opsi ini menunjukkan bagaimana proses input dan output ke file system dilakukan. sinkornisasi berarti dilakukan secara sinkronisasi. Contohnya digunakan pada floopy, artinya ketika anda menyalin sebuah file ke floopy, perubahan secara fisik langsung ditulis ke floppy saat anda memberikan perintah copy. Jika anda menggunakan opsi async, input dan output dilakukan secara asinkronisasi. Dengan contoh diatas, opsi ini membuat proses penyalinan ke floopy mungkin dilakukan jauh setelah perintah copy anda berikan. Hal ini tidaklah buruk, bahkan terkadang menjadi pilihan, dikarenakan misalnya jika anda memindahkan floopy tanpa melakukan unmounting terlebih dahulu, file-file yang disalin mungkin secara fisik belum masuk ke dalam floopy tersebut.

CLOCKS AND TIME

10.1 Time And Coordinaton

Time and Coordination adalah mengkordinasikan waktu dalam transfer data, agar tidak terjadi ketimpangan pada proses transfer data. Selain itu juga, berguna untuk mengukur penundaan antara komponen terdistribusi, menyinkronkan aliran data misalnya: suara dan video, dan sebagai penanda keakuratan waktu untuk mengidentifikasi atau mengotentikasi transaksi bisnis dan serializability dalam database terdistribusi dan keamanan protocol.

10.1.1 Time

Time adalah pengembangan dari sistem multiprogram. Beberapa job yang berada pada memory utama dieksekusi oleh CPU secara bergantian. CPU hanya bisa menjalankan program yang berada pada memory utama. Perpindahan antar job terjadi sangat sering sehingga user dapat berinteraksi dengan setiap program pada saat dijalankan. Suatu job akan dipindahkan dari memori ke disk dan sebaliknya. Time juga disebut dengan sistem komputasi interaktif, dimana sistem komputer menyediakan komunikasi on-line antara user dengan sistem. User memberikan instruksi pada sistem operasi atau program secara langsung dan menerima respon segera. Perangkat input berupa keyboard dan perangkat output berupa displayscreen, seperti cathode-ray tube (CRT) atau monitor. Bila sistem operasi selesai mengeksekusi satu perintah, maka sistem akan mencari pernyataan berikutnya dari user melalui keyboard. Sistem menyediakan editor interaktif untuk menulis program dan sistem debug untuk membantu melakukan debugging program.

10.1.2 Coordination

Coordnaion Sekumpulan algoritma yang tujuannya bermacam-macam namun men-share tujuannya, sebagai dasar dalam sistem terdistribusi : berupa sekumpulan proses untuk mengkoordinasikan tindakan atau menyetujui satu atau beberapa nilai. Contohnya pada kasus mesin seperti pesawat ruang angkasa. Hal itu perlu dilakukan, komputer mengendalikannya agar setuju pada kondisi tertentu seperti apakah misi dari pesawat luar angkasa dilanjutkan atau telah selesai. Komputer tersebut harus mengkoordinasikan tindakannya secara tepat untuk berbagi hal yang penting dalam Coordination and Agreement adalah apakah system terdistribusi asinkron atau sinkron. Algoritma–algoritma yang digunakan juga harus mempertimbangkan kegagalan yang terjadi, dan bagaimana caranya untuk berhubungan satu sama lain ketika sedang mendesaian algoritma. Selanjutnya di makalah ini juga akan

dijelaskan mengenai masalah dalam mendistribusikan mutual exclusion, election, multicast communication, dan mengenai masalah dalam persetujuan(agreement).

10.1.3 Contoh Time And Coordination Protokol Waktu Jaringan (Network Time Protocol)

Metode Cristian dan algoritma Berkeley pada dasarnya digunakan untuk komunikasi intranet. Protokol Waktu Jaringan (NTP) mendefinisikan arsitektur untuk pelayanan waktu dan protocol untuk distribusi informasi waktu lewat internet.

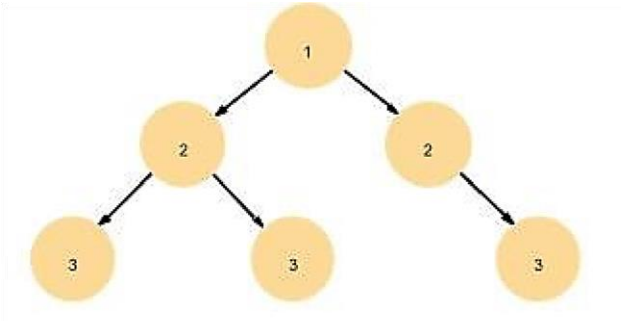
Tujuan dan fitur NTP, antara lain:

a) *To provide a service enabling clients across the Internet to be synchronized accurately to UTC*: NTP menyediakan layanan agar klien di internet dapat bersinkronisasi dengan UTC.

b) *To provide a reliable service that can survive lengthy losses of connectivity:* NTP menyediakan layanan yang bisa bertahan di jaringan mengalami loss karena jarak.

c) *To enable clients to resynchronize sufficiently frequently to offset the rate*s *of drift found in most computers:* NTP memungkinkan klien untuk sinkronisasi ulang secara berkala.

d) *To provide protection against interference with the time service*, *whether malicious or accidental:* NTP menyediakan perlindungan terhadap interferensi dari layanan waktu, baik galat maupun ketidaksengajaan.



Gambar 3 Contoh sinkronisasi subnet di NTP

Layanan NTP tersebar pada banyak server di internet. Server utama tersambung langsung ke sumber waktu, seperti penerima sinyal radio UTC. Server sekunder disinkronisasi dengan server primer. Server-servernya tersambung dalam hierarkikal logika yang disebut *synchronization subnet* seperti Gambar 3. Semakin atas levelnya akan semakin akurat *clock*-nya. Galat terjadi setiap melewati satu level.

Server-server NTP bersinkronasi satu sama lain dengan tiga cara, antara lain *multicast*, *procedure-call*, dan *symmetric*.

1. *Multicast*

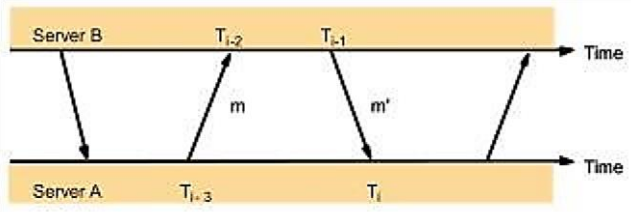
   Multicast ditujukan untuk LAN berkecepatan tinggi. Satu atau lebih server secara periodik menyebar waktu *clock* ke server di komputer lain yang tersambung di LAN. Mode ini akurasinya rendah tetapi cocok untuk berbagai kepentingan.

2. *Procedure-call*

   Procedure-call hamper sama dengan algoritma Cristian. Server menerima request dari komputer lain dan membalasnya dengan pembacaan *clock* saat pengiriman. Mode ini cocok ketika keakurasian tinggi dibutuhkan atau ketika *multicast* tidak dappat dilakukan.

3. *Symmetric*

   Mode *symmetric* ditujukan untuk server yang mensuplai waktu dalam LAN atau pada level tertinggi dari sebuah synchronization subnet.



Gambar 2. Message Exchange between a pair NTP peers

Pada mode *procedure-call* dan *symmetric* mode, memroses pertukaran bagianbagian pesan. Tiap pesan memiliki catatan waktu dari peristiwa yang baru saja terjadi, yaitu waktu local ketika pesan tersebut dikirimkan. Seperti pada Gambar 3, pesan *m* menyimpan catatan waktu setiap akan ditransmisikan, yaitu T_{i-3} dan T_{i-1}, dan ketika diterima, yaitu T_{i-2} dan T_i. Kemudian NTP menghitung jeda waktu antara dua *clock* komputer.

10.2 Share Data

10.2.1 Konsep dan operasi Shared Data antara server dan client

Dalam sistem terdistribusi, beberapa komputer yang berbeda saling terhubung satu sama lain melalui jaringan sehingga komputer yang satu dapat mengakses dan menggunakan sumber daya yang terdapat dalam situs lain. Misalnya, user di komputer A dapat menggunakan laser printer yang dimiliki komputer B dan sebaliknya user di situs B dapat mengakses file yang terdapat di komputer A.

1. Konsep Sharing Client – Server

Jaringan client atau server adalah jaringan dimana komputer client bertugas melakukan permintaan data dan server bertugas melayani permintaan tersebut.

A. Client

- User akan membuat permintaan melalui software client. Aplikasi ini berfungsi : Memberikan interface bagi user untuk melakukan jobs.
- Format request data ke bentuk yang dapat dimengerti oleh server
- Menampilkan hasil yang diminta pada layar

B. Server

Jaringan client atau server, server khusus digunakan untuk pemrosesan, penyimpanan dan manajemen data. Server bertugas menerima request dari client, mengolahnya, dan mengirimkan kembali hasilnya ke client.

Untuk itu, server membutuhkan komputer khusus dengan spesifikasi hardware yang jauh lebih baik dan bertenaga dibandingkan hardware untuk client karena komputer harus mampu melayani :

- Request secara simultan dalam jumlah besar
- Aktivitas manajemen jaringan
- Menjamin keamanan pada resource jaringan

10.2.2 Proses Layanan pada Saat Terjadi Crash atau Fault Tolerance & Data Transaction dan Urutan Operasi yang Dijalani Oleh Server

Sebuah kecelakaan (atau sistem crash) dalam komputasi adalah suatu kondisi di mana sebuah komputer atau program, baik aplikasi atau bagian dari sistem operasi, berhenti berfungsi dengan baik, sering keluar setelah menghadapi kesalahan. Seringkali program menyinggung mungkin muncul untuk membekukan atau hang sampai layanan pelaporan kecelakaan dokumen rincian kecelakaan itu. Jika program adalah bagian penting dari kernel sistem operasi, seluruh komputer dapat kecelakaan. Hal ini berbeda dari hang atau membekukan dimana aplikasi atau OS terus berjalan tanpa respon jelas untuk masukan.

Banyak crash adalah hasil dari eksekusi instruksi mesin tunggal, tetapi penyebab ini berlipat ganda. Penyebab khas adalah ketika program counter diatur ke alamat yang salah atau buffer overflow menimpa sebagian kode program karena bug sebelumnya. Dalam kedua kasus, itu cukup umum untuk

prosesor untuk mencoba untuk mengeksekusi data atau nilai memori acak. Karena semua nilai data adalah mungkin tetapi hanya beberapa nilai instruksi valid, ini sering mengakibatkan pengecualian instruksi ilegal.

10.2.3 Konsep Dasar Replication

Replikasi adalah suatu teknik untuk melakukan copy dan pendistribusian data dan objekobjek database dari satu database ke database lain dan melaksanakan sinkronisasi antara database sehingga konsistensi data dapat terjamin. Dengan menggunakan teknik replikasi ini, data dapat didistribusikan ke lokasi yang berbeda melalui koneksi jaringan lokal maupun internet. Replikasi juga memungkinkan untuk mendukung kinerja aplikasi, penyebaran data fisik sesuai dengan penggunaannya, seperti pemrosesan transaksi online dan DSS (Desiscion Support System) atau pemrosessan database terdistribusi melalui beberapa server.

Replikasi adalah proses menyalin dan memelihara objek database dalam beberapa database yang membentuk suatu sistem database terdistribusi. Replikasi dapat meningkatkan kinerja dan melindungi ketersediaan aplikasi karena data pilihan alternatif akses ada. Sebagai contoh, sebuah aplikasi biasanya dapat mengakses database lokal daripada server jauh untuk meminimalkan lalu lintas jaringan dan mencapai kinerja maksimum. Selanjutnya, aplikasi dapat terus berfungsi jika server lokal mengalami kegagalan, tetapi server lain dengan data direplikasi tetap dapat diakses.

DAFTAR PUSTAKA

[1] http://lecturer.ukdw.ac.id/anton/download/sister2.pdf
[2] http://www.scribd.com/hmdsq/d/27660443-Model-Sistem-Terdistribusi
[3] https://3mcr.wordpress.com/pengertian-networkjaringan/
[4] https://1nuy4s4.wordpress.com/pengertian-jaringan-komputer/
[5] http://tika.blog.fisip.uns.ac.id/2011/12/07/network-layer/
[6] http://penuhrahmatt.mdl2.com/mod/book/view.php?id=4&chapterid=2
[7] http://damaraaaa.wordpress.com/2013/03/19/konsep-objek-terdistribusidan-objectinterface/
[8] http://gunturvirgenius.student.telkomuniversity.ac.id/mari-mengenalinterprocesscommunication/
[9] http://mbahsecond.blogspot.co.id/2013/11/penertian-dan-cara-kerja-rpcrmi-dan.html
[10] http://zonacomplete.blogspot.co.id/2013/06/middleware-corba-dcomrmi.html
[11] http://foesiredgar.blogspot.com/2011/01/network-attached-storagenas.html
[12] http://opensource.telkomspeedy.com/wiki/index.php/Storage_Server
[13] http://sulistyawan.wordpress.com/2009/06/19/raid-1-vs-raid-5-which-one-better/
[14] http://blografika.wordpress.com/2011/08/04/mengenal-raid/
[15] http://www.scribd.com/doc/37975254/Storage-Area-Network
[16] https://pinkysoshi.wordpress.com/information-system-2/sistem-terdistribusi/file-sistemterdistribusi/
[17] http://forza-septiawan.blogspot.co.id/2012/04/definisi-name-service-padasistem.html
[18] http://muhammad-diak-huddin.blogspot.co.id/2013/05/time-andcoordination-sistem.html
[19] http://afizlah.blogspot.co.id/2014/11/proses-time-dan-coordinationpada.html