BAB II

TINJAUAN PUSTAKA

2.1 Jaringan Komputer

Jaringan komputer adalah sebuah kumpulan komputer, printer dan peralatan lainnya yang terhubung dalam satu kesatuan. Informasi dan data bergerak melalui kabel-kabel atau tanpa kabel sehingga memungkinkan pengguna jaringan komputer dapat saling bertukar dokumen dan data, mencetak pada printer yang sama dan bersama-sama menggunakan *hardware/software* yang terhubung dengan jaringan. Setiap komputer, printer atau periferal yang terhubung dengan jaringan disebut *node*. Sebuah jaringan komputer dapat memiliki dua, puluhan, ribuan atau bahkan jutaan *node*.

2.2.1 Jenis-Jenis Jaringan Komputer

Jaringan komputer dapat dibedakan berdasarkan cakupan geografisnya. Ada empat kategori utama jaringan komputer yaitu :

a. *Local Area Network* (LAN)

Local Area Network (LAN), merupakan jaringan milik pribadi di dalam sebuah gedung atau kampus yang berukuran sampai beberapa kilometer. LAN seringkali digunakan untuk menghubungkan komputer-komputer pribadi dan *workstation* dalam kantor suatu perusahaan atau pabrik-pabrik untuk memakai bersama sumberdaya (*resource*, misalnya printer) dan saling bertukar informasi.

b. *Metropolitan Area Network* (MAN)

Metropolitan Area Network (MAN), pada dasarnya merupakan versi LAN yang berukuran lebih besar dan biasanya menggunakan teknologi yang sama dengan LAN. MAN dapat mencakup kantor-kantor perusahaan yang letaknya berdekatan atau juga sebuah kota dan dapat dimanfaatkan untuk keperluan pribadi (swasta) atau umum. MAN mampu menunjang data dan suara, bahkan dapat berhubungan dengan jaringan televisi kabel.

c. *Wide Area Network* (WAN)

Wide Area Network (WAN), jangkauannya mencakup daerah geografis yang luas, seringkali mencakup sebuah negara bahkan benua. WAN terdiri dari kumpulan mesin-mesin yang bertujuan untuk menjalankan program-program (aplikasi) pemakai.

d. *Global Area Network* (GAN)

Global Area Network (GAN) merupakan suatu jaringan yang menghubungkan negara-negara diseluruh dunia. Kecepatan GAN bervariasai mulai dari 1.5Mbps sampai dengan 100Gbps dan cakupannya meliputi ribuan kilometer.

2.2 DHCP (*Dynamic Host Configuration Protocol*)

DHCP menggunakan hubungan *server-client*untuk mengalokasikan alamat, melacak penggunannya ,meminta kembali alamat IP yang sudah ditentukan dan informasi komunikasi yang diperlukan dalam *system* jaringan. *Server* DHCP biasa digunakan dengan range alamat IP tertentu. Ketika sebuah *system client*melakukan boot,*system* tersebut mengirimkan permintaan kepada sebuah *server* DHCP untuk mendapatkan alamat IP.

Pada umumnya, *server* DHCP berhubungan dengan alamat IP dan sebuah periode waktu yang ditentukan (disebut juga dengan lease) yang akan digunakan oleh *client*. DHCP mengatur reklamasi dan penugasan dari pengorganisasian alamat IP.

DHCP juga dapat digunakan untuk meneruskan informasi tambahan untuk sebuah *system* boot. Pertama, sebuah *client* melakukan *request* boot, yang termasuk didalamnya informasi konfigurasi yang spesifik mengenai *client*. *Server* menerima paket informasi ini dan membandingkan informasi konfigurasi dengn sebuah database parameter, dan bereaksi dengan data tambahan diluar alamat IP (seperti zona waktu atau informasi spesifik lainnya).

2.3 XDMCP (*X Display Manager Control Protocol*)

X Window System protocol tampilan dan jaringan yang dikembangkan oleh MIT. Sistem X dibangun dengan fungsi jaringan dan dengan kemampuan untuk menjalankan sesi grafis pada computer yang di *remote*. Didalamnya *X Display Manager* digunakan untuk memulai sebuah session dari *system* lokal atau dari computer lain. Permintaan dan permulaan session ditangani oleh XDMCP (X Display Manager Control Protocol) dan protocol jaringannya.XDMCP menyediakan sebuah cara untuk menjalankan X-Terminal untuk PC dan menggunakan X *Server* untuk menyediakan sebuah antarmuka *client*/*server* antara perangkat keras tampilan (mouse, keyboard, dan video) juga *desktop* dan juga menyediakan baik infrastruktur *windowing* dan sebuah antarmuka aplikasi standar. *X-Terminal* dapat ditampilan dengan sebuah jendela (tampilan) tunggal atau banyak jendela, berdasar kepada pengaturan dan kemampuan perangkat lunak *system* X.

2.4 NFS (*Network File System*)

NFS adalah sebuah *file system* yang dikembangkan oleh Sun Micro*system*s dan diperkenalkan pada akhir tahun 1984. NFS pada awalnya dikembangkan untuk workstation Unix namun juga banyak digunakan oleh workstation lain seperti Microsoft Windows. Beberapa istilah yang digunakan dalam NFS adalah sbb:

1. INODE, adalah sebuah struktur data yang mewakili berkas atau direktori yang terdapat pada file *system* UNIX. INODE digunakan untuk mengidentifikasi dan mencari sebuah berkas at au direktori yang terletak didalam file *system* lokal UNIX.
2. RNODE (remote file node), adalah sebuah struktur data yang mewakili berkas atau direktori yang terdapat pada *remote file system* (file *system* tidak terdapat pada komputer lokal)
3. VNODE (the virtual file node), adalah sebuah struktur data yang mewakili berkas atau direktori yang terdapat pada virtual file *system* (VFS)
4. VFS (virtual file *system*), adalah sebuah struktur data (*linked lists* dari VNODE) yang mengandung semua informasi yang diperlukan pada sebuah *file system* real yang diatur oleh NFS. Setiap VNODE berhubungan dengan sebuah file *system* disertakan didalam *linked list* yang berpaut kepada VFS untuk *file system* tersebut.

Server dan *Client*NFS berkomunikasi menggunakan teknik *remote procedure call*. Sebuah spesifikasi *external data representation* (XDR) digunakan untuk menjelaskan protokol RPC didalam sebuah mesin dan secara independen.

2.5 Sistem Terdistribusi

Teknologi Informasi telah mengalami konvergensi (perpaduan), antara tiga kutub utama, yaitu teknologi komputer, telekomunikasi dan multimedia. Konvergensi antara ketiganya melahirkan inovasi-inovasi teknologi, produk dan layanan teknologi informasi yang baru, seperti internet, telepon seluler (ponsel) dan *Personal Digital Assistant* (PDA) dengan aplikasi-aplikasinya. Hal tersebut membuka fenomena baru dalam ekplorasi pengembangan sistem terdistribusi, dimana infrastruktur teknologi baru memungkinkan terjadinya distribusi data, informasi objek, proses, beragam aplikasi, penggunaan bersama dan pengembangan sistem informasi. Konvergensi teknik informasi juga memungkinan terjadinya diversifikasi layanan *ClientServer*, seperti jasa layanan perbankan Anjungan Tunai Mandiri (ATM), *Internet Banking* dan sebagainya.

2.5.1 Pengertian Sistem Terdistribusi

Sistem Terdistribusi terdiri dari dua kata, yaitu Sistem dan Terdistribusi. Sistem merupakan sekumpulan elemen yang saling berhubungan dengan lainnya dan membentuk satu kesatuan untuk menyelesaikan suatu tujuan yang spesifik atau menjalankan seperangakat fungsi. Sedangkan kata terdistribusi berasal dari kata distribusi yang merupakan lawan kata sentralisasi, yang artinya penyebaran, sirkulasi, penyerahan, pembagian menjadi bagian-bagian kecil.

Maka berawal dari pengertian kata-kata pembentuknya, Sistem Terdistribusi dapat didefinisikan sebagai suatu keadaan dari elemen-element yang saling berinteraksi secara sistematis dan teratur untuk mendistribusikan data, informasi, Sistem Operasi, objek dan layanan dari dan kepada pengguna yang terkait didalamnya.

Selain pengertian diatas, sistem komputer terdistribusi didefinisikan pula sebagai sebuah sistem yang memungkinkan aplikasi komputer beroperasi secara terintegrasi pada lebih dari satu lingkungan yang terpisah secara fisis. Ciri khas sistem komputer terdistribusi adalah heterogenitas dalam berbagai hal perangkat keras, Sistem Operasi, dan bahasa pemrograman. Adalah tidak mungkin untuk mengembangkan sistem terdistribusi yang homogen secara paksaan, karena secara alamiah sistem komputer terdistribusi tumbuh dari lingkungan yang heterogen. Kata kunci dalam menjembatani perbedaan-perbedaan yang muncul adalah interoperabilitas *(interoperability)*. Interoperabilitas adalah kemampuan saling bekerjasama antar sistem komputer.

2.5.2 Tantangan Pengembangan Sistem Terdistribusi

Dalam mengembangkan sistem terdistribusi perlu memperhatikan beberapa aspek yang merupakan tantangan dalam pengembangan sistem terdistribusi, yaitu :

1. Keanekaragaman (*heterogencity*)

   Sistem Terdistribusi mampu mendukung berbagai jenis Sistem Operasi, perangkat keras dan perangkat lunak.

2. Keterbukaan (*openness*)

   Pengembangan sistem terdistribusi dilakukan dengan menambahkan komponen-komponen baru dapat dilakukan oleh programer yang berbeda-beda.

3. Keamanan (*security*)

Sistem Terdistribusi harus dapat menyediakan keamanan yang memadai bagi sumber daya yang digunakan bersama dan pesan yang dihantarkan dalam sistem.

4. Skalabilitas (*scalability*)

   Ukuran sistem terdistribusi dan tetap dapat berjalan dengan baik. Perubahan dapat dilakukan dari segi jumlah pengguna maupun dari segi kekuatan perangkat keras komputer-komputer dalam sistem terdistribusi itu sendiri.

5. Penanganan masalah (*error handling*)

   Kesalahan yang terjadi pada suatu komputer dalam sistem terdistribusi mempengaruhi kinerja sistem secara keseluruhan.

6. Kebersamaan (*concurrency*)

   Apabila terjadi permintaan layanan secara bersamaan sistem terdistribusi tidak akan menjadi kacau.

7. Penyembunyian (*transparancy*)

   Penyembunyian membuat aspek distribusi tidak tampil oleh pengguna.

   a. Penyembunyian akses

      Seseorang yang bekerja dalam lingkungan sistem terdistribusi dapat mengakses berbagai sumber daya yang berada dalam lingkungan tersebut untuk menyelesaikan pekerjaannya.

   b. Penyembunnyian lokasi

      Pengguna layanan transaksi tersebar tidak perlu mengetahui basis data yang akan diakses. Pengguna layanan transaksi juga dapat

mengakses basis data dari mana saja sejauh memiliki fasilitas untuk mengakses basis data yang bersangkutan.

c. Penyembunyian kebersamaan

Pengelola data mengetahui cara dalam menangani transaksi yang terjadi secara bersama.

d. Penyembunyian replikasi

Pengguna tidak terpengaruh apakah mengakses data orisinil atau replikasi.

e. Penyembunyian masalah

Apabila salah satu komputer dalam sistem terdistribusi mengalami kerusakan, kerusakan tersebut dapat diatasi secara langsung dan cepat tanpa terlihat dan tidak memerlukan banyak campur tangan dari pengguna.

f. Penyembunyian skala

Pengguna tidak terpengaruh atau direpotkan apabila komputer-komputer dalam sistem terdistribusi di-*upgrade* guna meningkatkan kinerja dan jangkauan layanan.

2.5.3 Arsitektur Sistem Terdistribusi

Arsitektur didefinisikan sebagai suatu rancangan untuk penyusunan dan operasi komponen-komponen suatu sistem dimana rancangan tersebut mengidentifikasikan komponen beserta fungsi masing-masing komponen, konektifitas/hubungan antar komponen dan mendeskripsikan pemetaan fungsionalitas ke dalam komponen. Arsitektur sistem terdistribusi diharapkan mampu menyediakan gambaran fisik dan logikal dari sistem, serta mampu

menyediakan aplikasi dari komponen-komponen sistem beserta hubungan atarkomponen dalam sistem. Beberapa jenis arsitektur sistem terdistribusi adalah sebagai berikut :

1. *ClientServer*

   *Client*menghubungi *server* untuk mendapatkan data, yang kemudian memformat dan menampilkan pada pengguna.

2. *Tightly Coupled* (*clustered*)

   Mesin-mesin terintegrasi yang menjalankan Sistem Operasi yang sama secara bersamaan dengan membagi tugas kedalam beberapa bagian yang dijalankan masing-masing mesin. Apabila Sistem Operasi telah selesai, hasil pekerjaan masing-masing mesin digabungkan menjadi satu.

3. *Peer-to-peer*

   Arsitektur dimana tidak ada mesin yang menyediakan layanan atau mengelola sumber daya jaringan, sehingga segala tanggung jawab dibagikan diantara seluruh mesin dikenal dengan istilah *pe*

2.5.4 Proses

Berikut akan dibahas beberapa materi mengenai proses.

a. Sistem Operasi

Sebuah Sistem Operasi adalah program/instruksi berurutan yang sedang dijalankan. Sistem Operasi mempunyai komponen sebagai berikut :

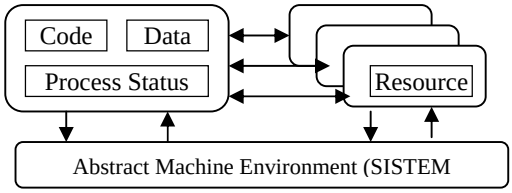
a. Kode program yang akan dieksekusi/dijalankan.

b. Data yang digunakan ketika program dieksekusi (berasal dari *file* ataupun hasil interaksi dengan pengguna).

c. Sumber daya yang dibutuhkan oleh program.

d. Status dari Sistem Operasi tersebut.

Agar Sistem Operasi bisa dijalankan, maka Sistem Operasi harus mempunyai lingkungan abstrak mesin yang sesuai untuk mengatur/mengelola sumber daya yang dapat digunakan secara bersama dan sumber daya yang terisolasi di antara komunitas Sistem Operasi.



Gambar II.1 Lingkungan mesin abstrak mengelola Sistem Operasi dan sumber daya

b. Thread

Motivasi adanya model *thread* adalah untuk membuat abstraksi sistem informasi yang sederhana dimana terdapat beberapa entri untuk menjalankan program yang sama menggunakan *file* dan peralatan/*device* yang sama pula. *Thread* merupakan bentuk lain sistem operasi menurut sistem informasi klasik Dalam model *thread*, Sistem Operasi adalah abstraksi sistem informasi yang dapat mengalokasikan beragam sumber daya, namun tidak mempunyai komponen untuk menjalankan program.

Thread adalah suatu entitas yang berjalan menggunakan program dan sumber daya dari sistem operasi yang memilki *thread* tersebut. Sistem operasi dapat mempunyai beberapa *thread*. *Thread sibling* (*thread* yang berada dalam satu sistem operasi) menggunakan program dan sumber daya sistem operasi secara bersama. *Thread* harus mempunyai alokasi sumber daya-nya sendiri

sehingga status internalnya dapat dibedakan dengan status internal *thread* lain. Secara praktis, sumber daya minimum yang harus dipunyai adalah *stack*, informasi status *thread*, dan beberapa entri tabel sistem informasi. *Thread* dijalankan dalam *address space* yang sama namun merupakan unit komputasi yang berdiri sendiri.

2.5.5 RPC (*Remote Procedure Call*)

Paradigma *remote procedure call* (RPC) merupakan perluasan dari lingkungan pemrograman sekuensial dengan mengambil keuntungan dari teknologi jaringan. Dalam pemrograman sekuensial, prosedur digunakan untuk memodularkan proses komputasi. Pemodularan ini membungkus data dan implementasi fungsi-fungsi ke dalam suatu *interface* prosedur publik.

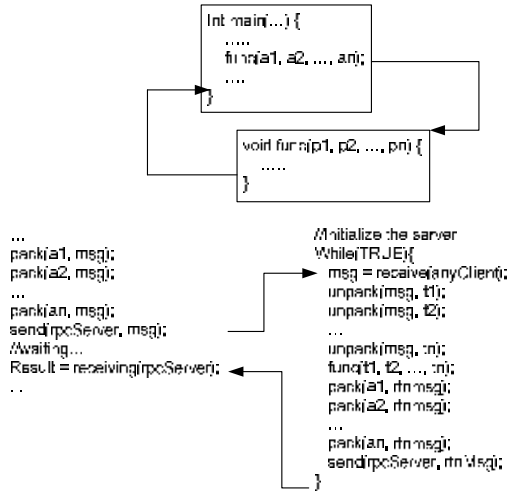
RPC merupakan sebuah variasi dari model penyampaian pesan. Esensi dari teknik ini adalah diperbolehkannya program-program yang terdapat dalam mesin yang berlainan untuk saling berinteraksi dengan menggunakan pemanggilan prosedur, seakan-akan kedua program berada pada satu mesin yang sama. Berikut beberapa penjelasan tambahan mengenai RPC, diantaranya :

a. Mekanisme Kerja RPC

Dalam suatu lingkungan komputasi terdistribusi, RPC diimplementasikan sebagai suatu kumpulan protokol jaringan yang membolehkan satu prosedur untuk memanggil prosedur lain yang dijalankan pada mesin yang berbeda dan mengirimkan salinan paramater untuk digunakan dalam proses komputasi. Karena alasan ini, RPC menjalankan pada ruang alamat yang berbeda dari prosedur pemanggil. RPC adalah suatu pola komunikasi antar proses yang mana program penginisialisasi menjalankan operasi pengiriman yang

langsung diikuti operasi baca-pemblokan. Program penerima akan menjalankan proses baca-pemblok hingga ia mendapat pesan yang dikirim oleh proses pemanggil tadi. Selanjutnya, akan memberikan pelayanan dan mengembalikan hasilnya ke proses asal.

Konsep sinkron dan tidak sinkron pada RPC analog dengan konsep *blocking* (pemblokan) dan *nonblocking* pada pesan. RPC yang tradisional bersifat sinkron yaitu yang mengharuskan proses pemanggil menunggu hingga proses terpanggil mengembalikan nilai. RPC yang sinkron mudah dipahami dan diprogram karena perilakunya dapat diperkirakan akan tetapi tidak berhasil mengeksploitasi keparalelan yang terwaris dalam aplikasi terdistibusi. Ini membatasi macam interaksi yang bisa dimiliki oleh aplikasi yang berakibat pada turunnya unjuk kerja. Untuk memberikan fleksibilitas yang lebih luas, RPC tidak-sinkron diimplementasikan untuk mengoptimalkan keparalelan sambil tetap mempertahankan kesederhanaan RPC. RPC tidak-sinkron tidak memblok pemanggil, nilai-nilai yang dikembalikan dapat diterima saat mereka dibutuhkan, yang memungkinkan eksekusi proses di *client*dilanjutkan secara lokal secara paralel dengan pemanggilan ke *server*.



Gambar II.2 Sinkronisasi RPC

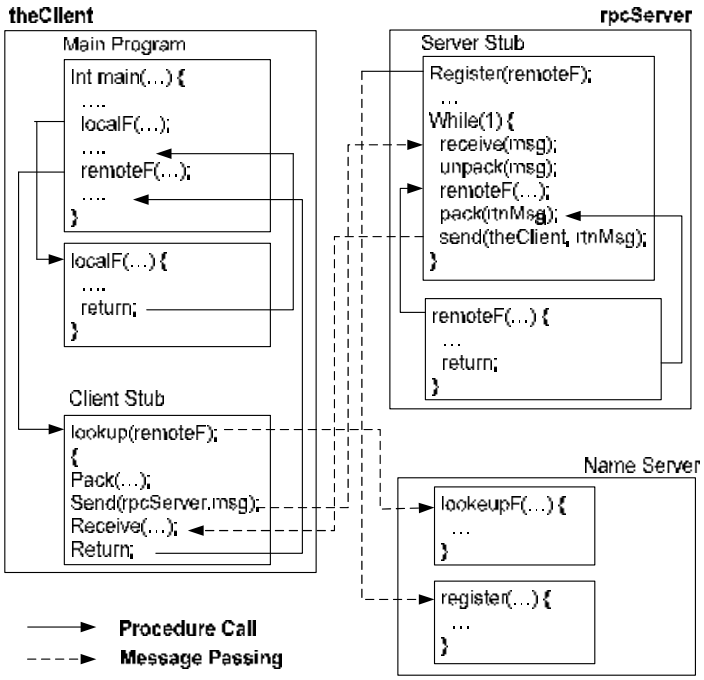
b. Pengimplementasian RPC

Ada beberapa isu yang didapat dalam pengimplementasian RPC, antara lain :

a. Sintaksis RPC sebaiknya memiliki penampilan sintaksis yang sama seperti pemanggilan prosedur lokal dalam bahasa pemrograman tingkat tinggi.
b. Semantik pemanggilan dalam kasus *remote* dan kasus lokal diupayakan seserupa mungkin, walaupun pada pelaksanaannya sulit.
c. Penerima proses RPC sebaiknya dijalankan dalam lingkungan yang sama dengan lingkungan dimana pemanggilan dilaksanakan. Untuk alasan ini, sebuah prosedur dapat mengacu dan mengubah variabel global dalam lingkungan konvensional.

c. Pengorganisasian Umum

Pada proses implementasi RPC, mesin *client*mengeksekusi proses yang terdiri atas kode aplikasi *client, client stub* dan mekanisme transport. Mesin *server* mengimplementasikan proses rpc*Server* dengan sebuah mekanisme transport, program utama *server stub* dan implementasi *server* dari prosedur *remote*. *Client stub* berfungsi menterjemahkan prosedur lokal menjadi aksi pada sisi *client*dari protokol RPC dan *server stub* mengimplementasikan sisi *server* dari protokol RPC.



Gambar II.3 Implementasi RPC

d. Dukungan jaringan

Mekanisme *transport* mengimplementasikan penyampaian pesan jaringan. Implementasi aktual cenderung untuk menggunakan protokol datagram dengan protokol RPC yang khusus. Pendekatan ini dibenarkan dengan

memperhatikan bahwa protokol RPC tidak mensyaratkan sirkuit *virtual* seperti yang disediakan oleh TCP.

RPC berguna untuk proses tersebar pada mesin berbeda tetapi dia tidak mendorong komputasi paralel. Pada implementasi RPC, unjuk kerja pada saat *runtime* selalu menjadi isu utama. Walau begitu, prosedur *remote* digunakan secara luas dalam aplikasi terdistribusi karena dia mengimplementasi mode pemrograman tradisional tanpa perlu membutuhkan pengetahuan yang banyak tentang strategi dan mekanisme distribusi.

2.5.6 Penamaan

Nama dalam sebuah sistem yang terdistribusi adalah sebuah string dari bit atau karakter yang digunakan untuk menunjuk sebuah *Entity*. Untuk mengoperasikan di dalam sebuah *Entity*, diperlukan sesuatu untuk mengaksesnya, yaitu sesuatu yang disebut *Acces Point*. Nama dari sebuah *Acces Point* disebut *Address*. Untuk menambahkan *Address*, perlu adanya tipe nama yang lain mengirimkan sebuah perintah yang khusus, sebagaimana nama unik yang digunakan untuk *identify entity*. *Identifier* yang benar adalah sebuah nama yang mempunyai properti berikut.

1. Sebuah *Identifier* yang menunjukan maksimal satu *Entity*.
2. Tiap *Entity* ditunjukkan oleh maksimal satu *Identifier*.
3. Sebuah *Identifier* selalu menunjuk ke *Entity* yang sama.(tidak pernah digunakan kembali)

Nama dalam sistem terdistribusi biasanya diorganisasikan kedalam sesuatu yang disebut *name space*. Sebuah *name space* dapat diwakili oleh *label*, yaitu

grafik yang diarahkan melalui dua tipe *Node* yaitu *Leaf Node* yang mewakili sebuah nama *Entity* dan mempunyai properti yang tidak mempunyai *outgoing edge* dan *Directory Node* yang mempunyai sejumlah *outgoing edge.*

i. Fault Tolerance

Sebuah sistem dikatakan *fail* apabila sistem tersebut tidak dapat melakukan apa yang sudah dijanjikan. *Error* adalah apabila ada bagian dari sistem yang menujukkan kegagalan. Penyebab dari *error* disebut *fault*. Topik yang paling hangat dari *fault tolerance* adalah bagaimana caranya sistem tersebut dapat menyediakan servis meskipun dalam keadaan *fault*.

Secara umum *fault* diklasifikasikan kedalam *transient*, *intermittent*, atau *permanent*.

2. *Transient fault* terjadi sekali dan kemudian menghilang. Jika pengoperasian diulang, *fault* pun menghilang.

3. *Intermittent fault* terjadi, kemudian lenyap, lalu muncul kembali dan begitu seterusnya.

4. *Permanent fault* adalah satu-satunya *fault* yang terus ada sampai komponen yang mengalami *fault* tersebut diperbaiki.

Terdapat beberapa model *Failure*, diantaranya :

1. *Crash failure* terjadi apabila *server* berhenti secara prematur, tapi pernah berjalan secara benar sampai *server* itu berhenti.
2. *Ommision failure* terjadi ketika *server* gagal untuk merespon sebuah permintaan.

3. *Timing failure* terjadi apabila respon yang terjadi tidak sesuai dengan waktu yang seharusnya.
4. *Respon failure* adalah apabila *respon* dari *server* tidak benar. Ini adalah *failure* yang termasuk kategori serius.
5. *State transition failure* terjadi apabila respon dari *server* tidak sesuai dengan apa yang diharapkan oleh permintaan yang datang.
6. *Arbitrary failure* atau disebut juga *Byzantine failure*, ketika *failure* ini terjadi, dampaknya adalah *client*harus bersiap untuk kemungkinan yang terburuk.

2.6 Client server

ClientServer merupakan salah satu arsitektur dari sistem terdistribusi. Arsitektur ini memiliki beberapa hal yang perlu diketahui, diantaranya :

2.6.1 Komponen dasar *ClientServer*

1. ***Client***

   *Client*merupakan terminal yang digunakan oleh pengguna untuk meminta layanan tertentu yang dibutuhkan. Terminal *client*dapat berupa PC, ponsel, komunikator, robot, televisi dan peralatan lain yang membutuhkan informasi

2. ***Middleware***

   Midleware merupakan komponen perantara yang memungkinkan *client*dan *server* untuk saling terhubung dan berkomunikasi satu sama lain. *Midleware* ini dapat berupa *Transaction Monitor* /TP. *Remote Procedure Call* atau *Object Request Broker*/ORB.

3. ***Server***

   Server merupakan pihak yang menyediakan layanan. *Server* ini akan dapat berupa basis data SQL, Monitor TP, *server groupware*, *server* objek dan web. Secara umum, *server* berperan menerima pesan permintaan layanan dari *client*, memproses permintaan tersebut dan mengirimkan hasil permintaan kepada *client*.

2.6.2 Ciri-ciri *Cilent Server*

Beberapa ciri dari arsitektur sistem terdistribusi *ClientServer* diantaranya :

1. **Berbasis layanan**

   Server memberikan sejumlah layanan yang dibutuhkan dan diminta oleh *client*, antara lain : berbagai pakai berkas, dan peralatan pendukung.

2. **Sumber daya yang digunakan bersama**

   Server mengelola sejumlah sumber daya yang dimiliki agar dapat diakses dan digunakan secara bersama-sama oleh terminal-terminal *client*yang terhubung pada *server*.

3. **Hubungan dan interaksi *Client Server***

   Hubungan yang terjadi antara *server* dan *client* adalah *one-to many*, yang berarti bahwa satu *server* melayani banyak *client*. *Client* selalu memulai transaksi dengan meminta layanan sedangkan *server* menanti permintaan layanan secara pasif.

4. ***Client* tidak perlu mengetahui lokasi fisik *server***

   Server dapat terletak di berbagai tempat yang belum tentu diketahui oleh *client*, Walaupun demikian *client*tetap dapat mengakses *server* untuk mendapatkan layanan sesuai kebutuhannya.

5. **Interoperabilitas perangkat lunak dan perangkat keras**

   Perangkat lunak dan keras yang digunakan oleh masing-masing *client*tidak harus sama dengan yang digunakan pada *server*, namun masih dapat saling terkoneksi antara satu dan yang lain.

6. **Pertukaran berbasis pesan**

   Mekanisme dari *ClientServer* berdasar pada pertukuran pesan. Pesan yang dipertukarkan adalah permintaan layanan dan umpan balik dari permintaan layanan tersebut.

7. **Enkapsulasi layanan**

   *Client*tidak perlu mengetahui Sistem Operasi pengelolaan permintaan yang terjadi dalam *server* sehingga *client* tidak dapat mengontrol Sistem Operasi pengelolaan permintaan.

8. **Skalabilitas**

   Skalabilitas adalah kemampuan untuk diperbesar atau diperkecil. Ukuran sistem *ClientServer* dapat diubah secara horizontal maupun vertikal. Perubahan vertikal berarti berpindah ke *server* lebih besar atau lebih cepat atau mendistribusikan tugas melayani *client*ke beberapa *server*. Pengubahan horizontal berarti menambah atau mengurangi jumlah *client*.

9. **Konsistensi data**

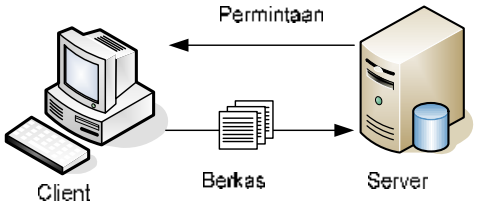
   Data hanya dikelola pada *server* pusat sehingga konsistensi dan data lebih terjamin dan biaya pemeliharaan pun menjadi lebih murah.

2.6.3 Tipe Jaringan *Client Server*

Berdasarkan tipe layanan yang diberikan *server* kepada *client*, jaringan *ClientServer* dapat dibagi menjadi kedalam banyak tipe, tipe-tipe tersebut antara lain :

1. ***Server* Berkas**

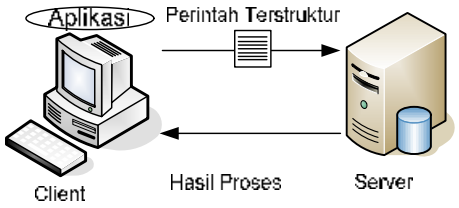
   Sistem jaringan berkas adalah sistem jaringan yang dimana layanan yang diberikan *server* berupa berkas, baik berkas aplikasi seperti aplikasi pengolahan kata, pengolahan angka, pengolahan data, pengolahan gambar dan lain sebagainya, maupun berkas yang dihasilkan oleh aplikasi tersebut, seperti dokumen pengolahan kata, tabel-tabel pengolahan angka, berkas presentasi dan lain sebagainya.



Gambar II.4 *Server* Berkas

2. ***Server* Basis Data**

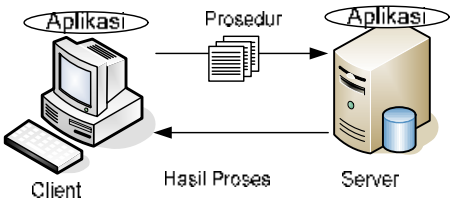
   Sistem jaringan *server* basis data adalah merupakan sistem jaringan dimana layanan yang diberikan oleh *server* berupa pengolahan dan penyajian data berdasarkan perintah terstruktur (*query*) yang diberikan *client*. Pada jaringan ini, *server* menyimpan berbagai macam data yang dapat diakses oleh pengguna melalui terminal-terminal *client*.



Gambar II.5 *Server* Basis Data

3. *Server* Transaksi

Sistem jaringan *server* transaksi adalah sistem jaringan dimana layanan yang diberikan *server* berupa hasil Sistem Operasi dari sekelompok perintah terstruktur yang diberikan *client.* Jaringan ini pada dasarnya hampir sama dengan sistem jaringan basis data sebelumnya. Perbedaan terletak pada *server* transaksi yang memproses sekelompok perintah terstruktur dari *client*, dan sekelompok perintah terstruktur ini disebut prosedur.



Gambar II.6 *Server* Transaksi

4. Groupware *Server*

Sistem jaringan groupware adalah sistem jaringan dimana layanan yang diberikan *server* berupa fasilitas pemakaian bernama informasi semi terstruktur diantara pengguna jaringan. Pada jaringan ini, *server* menyimpan, mengelola dan menyebarkan informasi antar pengguna dalam jaringan, misalnya teks, gambar, surat dan ruang diskusi.

5. *Server* Objek

Sistem jaringan *server* objek adalah sistem jaringan dimana layanan yang diberikan *server* berbentuk objek. Dalam jaringan ini, *client*dan *server* berkomunikasi melalui objek-objek yang miliki *client*dan *server*.

6. Web *Server*

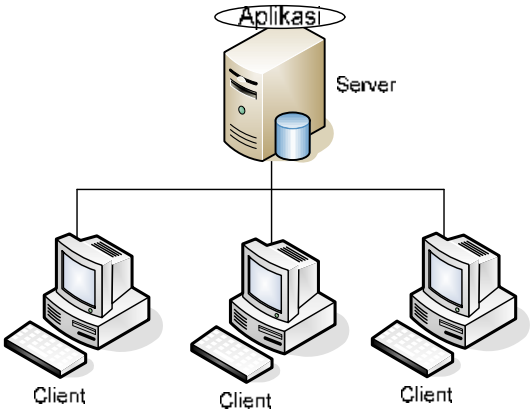
Sistem jaringan web *server* adalah sistem jaringan dimana layanan yang diberikan *server* berupa pengelolaan dan pemakaian bersama dokumen-dokumen yang saling terhubung. Jaringan ini merupakan jaringan yang memungkinkan tiap dokumen dalam jaringan memiliki hubungan ke dokumen lain sehingga dokumen-dokumen dalam jaringan terhubung satu dengan yang lain, semacam jaringan laba-laba.

2.6.4 Arsitektur *Client Server*

1. Two Tier

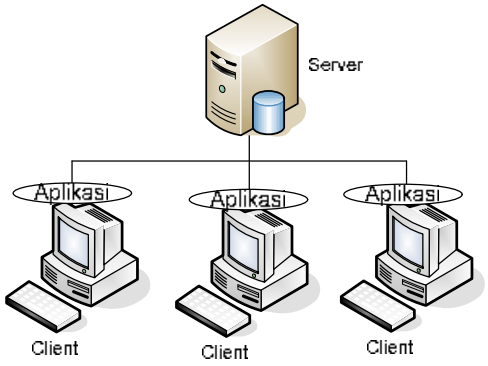
Arsitektur *two tier* merupakan arsitektur yang disebut *ClientServer* dimana terdapat komputer sebagai *client*dan *server* yang berinteraksi melalui protokol dan media komunikasi tertentu Model arsitektur *Two Tier* dikelompokan menjadi dua macam yaitu

a. Thin *Client*– thick *Server*



Gambar II.7 *Thin Client – Thick Server*

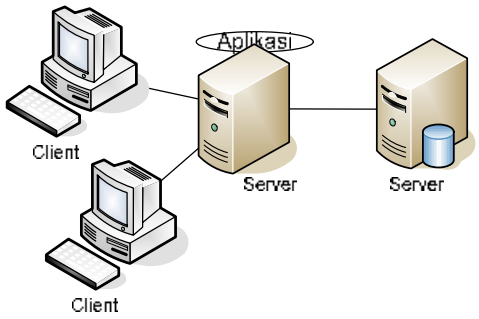
b. Thick Client – Thin Server



Gambar II.8 *Thick Client – Thin Server*

2. Three Tier

Arsitektur *ClientServer* ini memisahkan antara data (*Data Management Tier*), aplikasi (*Middle Tier*) dan penyajian (*Presentation Layer*) sebagaimana ditampilkan pada gambar dibawah ini :



Gambar II.9 *Arsitektur Three Tier*

a. Data Management Tier

Merupakan komputer *server* yang dikhususkan untuk menangani pengolahan basis data.

b. Middle Tier

Merupakan komputer *server* yang dikhususkan untuk menangani aplikasi-aplikasi dimana prosedur-prosedur dan perhitungan-perhitungan yang kompleks dilakukan di komputer.

c. Presentation Layer

Merupakan komputer *client*yang menjadi *interface* bagi pengguna untuk memasukkan data, mengajuan permintaan layanan kepada *server* dan melihat hasilnya..

3. n-Tier

Istilah *n-tier* menunjukan lapisan yang ada dalam sebuah aplikasi. Sebuah aplikasi terdiri dari beberapa komponen utama, yaitu lapisan *Presentation* (*Presentation Layer*), lapisan *application* (*Application Layer*) dan logika bisnis (*bussiness logic layer*) dan lapisan data (*Data layer*).Lapisan *Presentation* menghubungkan antarmuka dengan pengguna aplikasi, dapat berupa model grafis atau berupa teks. Pengguna dapat

berinteraksi dengan aplikasi tersebut menggunakan lapisan *presentation* ini. Lapisan *Application* berisi inti dari aplikasi dan lapisan data yang digunakan oleh aplikasi tersebut. Lapisan data dapat berbentuk satu atau lebih *server* basis data yang lokasinya tersebar dibeberapa tempat.

2.7 Kernel

Kernel yang pada banyak persepsinya disamakan dengan Linux adalah *low level code* yang berfungsi untuk mengatur interaksi perangkat keras komputer yang kemudian lebih dikenal dengan istilah kernel sistem operasi. Kernel ini bertugas menyediakan sumber daya yang akan digunakan oleh aplikasi untuk dapat saling berinteraksi, memungkinkan CPU agar dapat digunakan oleh berbagai aplikasi secara bersamaan dan mengatur penggunaan memori.

Kernel kemudian dilengkapi dengan aplikasi yang menyediakan perangkat fungsional sistem operasi. Perangkat fungsional ini menyediakan antar muka perintah berbasis teks yang berfungsi memberikan instruksi kepada kernel agar dapat berinteraksi dengan perangkat keras komputer. Selain itu, kernel juga dilengkapi dengan *compiler*, sebuah aplikasi yang berfungsi untuk mengubah kode sumber program menjadi aplikasi *binary* yang siap digunakan.

Hingga saat ini, kernel terus dikembangkan agar dapat mengenali dan mengoperasikan berbagai perangkat keras komputer dan elektronik baru yang terus menerus bermunculan. Perkembangan kernel ini kemudian ditandai dengan munculnya versi kernel. Perkembangan kernel terbaru dapat dilihat pada situs http://kernel.org/.

2.8 Linux

Sistem operasi GNU/Linux adalah sebuah sistem operasi yang dibangun dari perpaduan antara kernel Linux yang dibuat oleh Linus Torvalds dan GNU, sebuah sistem yang menyerupai Unix namun terdiri dari berbagai macam program dan komponen sistem operasi yang bersifat terbuka dan bebas yang dikembangkan oleh Free Software Foundation (FSF).

Linux merupakan sistem operasi yang dapat diandalkan, baik sebagai sebuah *workstation/desktop* maupun sebagai *server* yang menyediakan berbagai service dan aplikasi seperti yang dapat dilakukan oleh *server* berbasis sistem operasi Microsoft Windows *Server*, Unix, dan *server* lainnya. Selain itu di Linux, semua program (yang berada dibawah lisensi GNU) dapat diperoleh secara cuma-cuma dan legal dengan cara mengunduhnya langsung dari internet. Hampir seluruh perangkat lunak dan berbagai aplikasi yang biasa digunakan pada sistem operasi windows terdapat padanannya pada sistem operasi Linux.

Migrasi dari sistem operasi Microsoft Windows (atau sebagian diantaranya dari Macintosh) ke sistem operasi Linux adalah isu yang menarik dan sering muncul. Berkaitan dengan kemudahan pemakaian, banyak pengguna komputer yang mengalami kesulitan pada saat pertama kali menginstalasi Linux. Selanjutnya, saat instalasi berjalan mulus, pengguna dihadapkan pada *prompt* dengan perintah-perintah yang asing untuk dapat membuatnya bekerja. Keadaaan semacam ini tampaknya disadari sepenuhnya oleh para pengembang Linux sehingga kemudian muncul berbagai pengembangan modus grafik pada seluruh proses instalasi dan penggunaannya sehingga para pengguna baru dapat menggunakan dan memahami Linux dengan mudah.

2.9 Distribusi Linux

Distribusi Linux (juga disebut GNU / Linux distribusi oleh beberapa *vendor* dan *user*) adalah anggota dari keluarga Unix seperti distribusi software yang dibangun di atas kernel Linux. Seperti distribusi (sering disebut singkat untuk distro-distro) yang besar terdiri dari kumpulan perangkat lunak aplikasi seperti pengolah kata, spreadsheet, pemutar media, dan aplikasi database. Sistem operasi akan terdiri dari kernel Linux, dan biasanya, sejumlah perpustakaan dan utilitas dari GNU proyek, dengan dukungan dari grafis *X Window System*. Distro dioptimalkan untuk ukuran mungkin tidak berisi X, dan cenderung menggunakan lebih kompak alternatif untuk GNU utilitas seperti *busybox*, *uclibc* atau *dietlibc*. Saat ini terdapat lebih dari tiga ratus distribusi Linux. Kebanyakan dari mereka yang aktif dalam pembangunan, yang direvisi dan terus ditingkatkan. Karena kebanyakan dari kernel dan paket yang mendukung beberapa kombinasi dari perangkat lunak bebas dan open source, distribusi Linux telah mengambil berbagai bentuk - fitur penuh dari *desktop* dan *server* untuk sistem operasi minimal lingkungan (biasanya untuk digunakan dalam *embedded system*, atau untuk *booting* dari floppy disk). Selain beberapa perangkat lunak kustom (seperti *installers tool* dan konfigurasi) yang hanya merujuk kepada distribusi tertentu *assortment* aplikasi diinstal di atas satu set perpustakaan disandingkan dengan versi kernel, antara lain bahwa "*out-of-the-box*" kemampuan memenuhi sebagian besar kebutuhan khusus para pengguna yang menginginkan fungsi tertentu. Terdapat juga distibusi Linux juga didukung distribusi komersial, seperti Fedora (Red Hat), openSUSE (Novell), Ubuntu (Canonical Ltd), dan Mandriva Linux dan masyarakat seperti distribusi Debian dan Gentoo, meskipun ada Distro lain yang

tidak didorong oleh korporasi maupun masyarakat, yang salah satu contohnya adalah Slackware.

2.10 Struktur Direktori

Salah satu perbedaan utama antara sistem operasi Linux dengan DOS ataupun windows adalah sistem file. Linux tidak mengunakan notasi drive yang berbeda untuk membedakan partisi yang terdapat pada hard disk karena semuanya disimpan dalam sebuah direktori utama yang dikenal dengan nama *root* (/). Linux mengenali semua partisi yang ada pada sebuah hard disk dengan menempatkan partisi-partisi tersebut kedalam sebuah direktori yang terdapat pada direktori utama.

Keunggulan dari penggunaan metode satu direktori utama ini adalah kemampuannya untuk menciptakan sebuah direktori yang bisa terdiri dari berbagai partisi yang dapat membuat kapasitas penyimpanannya bertambah. Sebagai contoh, jika pada sistem operasi windows kapasitas drive sistem c:\ sudah habis, maka kita tidak dapat menambahkan kapasitas baru dari partisi lain agar kapasitas drive c:\ bertambah, sedangkan pada sistem operasi Linux, jika kapasitas direktori utama root sudah habis, maka kita dapat menambahkan partisi lain cukup dengan menempatkannya pada direktori utama tadi dan secara otomatis maka kapasitasnya pun akan bertambah.

Di dalam direktori utama, terdapat beberapa direktori yang merupakan bagian dari sistem operasi Linux dan memiliki fungsi-fungsi tertentu. Direktori berikut fungsinya tersebut antara lain :

1. /

   Direktori ini adalah direktori utama sistem operasi Linux yang menampung seluruh sub-direktori yang terdapat pada sistem operasi Linux.

2. /bin

   Direktori ini berisi program-program utama yang dapat dieksekusi oleh pengguna untuk berinteraksi dengan sistem operasi. Selain itu, direktori ini berisi *shell* dan juga perintah-perintah sistem file seperti *ls, cp, mkdir,* dll.

3. /boot

   Direktori ini beris program yang digunakan untuk melakukan proses pada saat komputer dinyalakan *(booting)*, program tersebut akan mengaktifkan kernel Linux yang juga terdapat pada direktori ini.

4. /dev

   Pada sistem operasi Linux, semua sistem, berikut dengan konfigurasi dan bahkan juga perangkat kerasnya dikenali kedalam file. Semua perangkat keras seperti *serial port, usb, hard disk,* dll akan dibuatkan sebuah file khusus yang berisi informasi perangkat keras dan metode aksesnya. Direktori ini berisi file-file khusus mengenai perangkat keras yang terdapat pada komputer.

5. /etc

   Direktori ini berisi seluruh file konfigurasi sistem, mulai dari file konfigurasi tampilan grafis, database pengguna, konfigurasi pada saat komputer dinyalakan dan file konfigurasi lain yang digunakan oleh sistem operasi.

6. /home

   Sistem operasi Linux bersifat *multi user* sehingga setiap pengguna diberikan sebuah akses dan direktori kerja masing-masing yang bersifat pribadi dan berbeda antara pengguna yang satu dengan pengguna yang lainnya. Direktori ini berisi direktori-direktori pengguna tersebut.

7. /lib

   Direktori ini berisi seluruh pustaka *(libraries)* yang akan digunakan oleh sistem operasi.

8. /mnt

   Direktori ini berisi akses menuju media penyimpanan data seperti media cakram padat (CD-ROM) ataupun tempat meletakkan partisi-partisi dari media penyimpanan data lain kedalam direktori utama.

9. /opt

   Direktori ini berisi program ataupun aplikasi tambahan yang tidak terdapat pada direktori /bin yang akan digunakan oleh pengguna. Biasanya program/aplikasi tersebut adalah program/aplikasi tambahan (*optional*).

10. /proc

    Direktori ini adalah salah satu direktori spesial yang berisi *virtual filesystem* yang menyediakan akses menuju informasi kernel seperti jenis prosesor yang digunakan, kecepatan akses memori, dan lain sebagainya.

11. /root

    Pada sistem operasi Linux, terdapat pengguna yang memiliki hak akses penuh untuk melakukan pengubahan terhadap sistem. Pengguna ini dikenal dengan nama *root* dan memiliki direktori kerja yang terpisah dari

pengguna lainnya. Direktori ini merupakan direktori kerja untuk pengguna tersebut.

12. /sbin

    Direktori ini berisi program-program yang dapat digunakan oleh pengguna *root* untuk mengubah konfigurasi sistem operasi. Seluruh file dan program yang terdapat pada direktori ini tidak dapat digunakan oleh pengguna biasa.

13. /tmp

    Direktori ini digunakan sebagai media penyimpanan data sementara pada saat proses komputasi berlangsung. Direktori ini dapat digunakan oleh semua pengguna yang terdapat pada sistem operasi.

14. /usr

    Direktori ini merupakan direktori utama untuk berbagai macam keperluan seperti dokumentasi, kode sumber kernel, modul-modul kernel, dan juga program-program seperti Windows Manager untuk tampilan grafis dan juga program lain yang di install dari kode sumber.

15. /var

    Direktori ini berisi *log* dan *cache* dari seluruh proses komputasi yang dilakukan oleh sistem operasi.