BAB II
LANDASAN TEORI

II.1 Jaringan Komputer

Jaringan komputer adalah sebuah sistem yang terdiri atas komputer dan perangkat jaringan lainnya yang bekerja bersama-sama untuk mencapai suatu tujuan yang sama. Tujuan dari jaringan komputer adalah:

a. Membagi sumber daya: contohnya berbagi pemakaian printer, CPU, memori, harddisk
b. Komunikasi: contohnya surat elektronik, *instant messaging*, *chatting*
c. Akses informasi: contohnya *web browsing*

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

Klasifikasi jaringan komputer berdasarkan skala atau ukurannya adalah sebagai berikut :

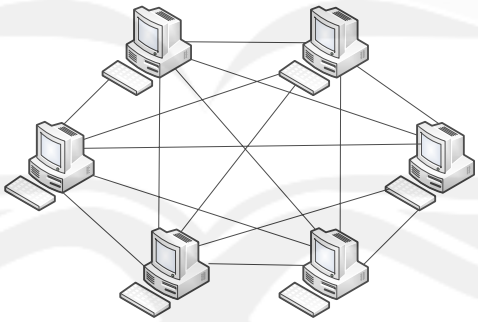
a. Personal Area Network (PAN)
b. Campus Area Network (CAN)
c. Local Area Network (LAN)
d. Metropolitant Area Network (MAN)
e. Wide Area Network (WAN)
f. Global Area Network (GAN)

Berdasarkan fungsinya setiap jaringan komputer ada yang berfungsi sebagai client dan juga server. Tetapi,

ada jaringan yang memiliki komputer yang khusus didedikasikan sebagai server sedangkan yang komputer lain sebagai client. Ada juga jaringan yang tidak memiliki komputer yang khusus berfungsi sebagai server.

II.1.1 Perbandingan Arsitektur Client-server dengan Arsitektur Peer-to-peer

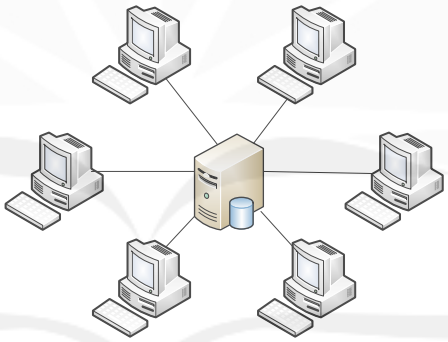
Sistem operasi jaringan sangat menentukan bentuk arsitektur jaringan yang dibangun. Ada dua macam kelompok arsitektur jaringan, yaitu Peer-to-peer (P2P) dan Client-server. Masing-masing arsitektur tersebut memiliki perbedaan dalam derajat koneksitasnya, maupun bentuk hubungan antara Server dengan terminalnya. Namun, semua arsitektur tersebut berfungsi menciptakan hubungan antarterminal yang ada dan menentukan fungsi serta peran masing-masing terminal yang terkoneksi satu sama lain.



Gambar 2.1 Arsitektur Peer-to-Peer

Penerapan arsitektur jaringan ini juga ditentukan oleh skalabilitas dan tingkat penggunaan jaringan. Untuk skala kecil, dengan jumlah komputer yang terhubung hanya berkisar tiga sampai lima buah, maka penerapan arsitektur P2P akan memberikan keuntungan tersediri, di mana semua terminal dapat berfungsi sebagai Client maupun Server.

Namun, untuk sistem jaringan yang lebih besar, baik jumlah terminal yang terhubung maupun skala jangkauan yang lebih luas, apalagi untuk penanganan sistem terdistribusi, penerapan arsitektur P2P tidak efektif lagi. Tentu saja dalam skala besar akan sulit sekali jika tidak diketahui dengan pasti layanan apa yang disediakan oleh komputer-komputer yang ada. Hal itu, akan membuat Client mengalami kesulitan untuk menemukan Server yang menyediakan layanan yang diperlukan. Oleh karena itu, pada sistem jaringan dengan skala yang lebih besar akan lebih tepat jika diterapkan arsitektur Client-server, di mana ada satu komputer yang ditetapkan sebagai Server dan semua layanan dipusatkan pada Server tersebut.



Gambar 2.2 Arsitektur Client-server

II.1.2 Arsitektur Client-server

Model konektivitas pada jaringan yang membedakan fungsi komputer sebagai terminal akses serta pusat pengolahan dan layanan disebut Client-server. Arsitektur ini menempatka nsebuah komputer sebagai server yang bertugas sebagai pusat pengolahan dan layanan bagi terminal-terminal lain (client) yang terhubung dalam sistem jaringan itu. Pada model

arsitektur ini, client tidak dapat berfungsi sebagai server (Dharma Oetomo, 2003, hal 124), tetapi server dapat berfungsi sebagai client (*server non dedicated*), meski sebaiknya dihindari agar tidak berubah menjadi arsitektur P2P dan menurunnya kinerja server mengingat server juga merangkap peran sebagai client. Komputer yang difungsikan sebagai server hanya berperan untuk melayani permintaan layanan dari client.

Model ini dapat menjawab problematika rendahnya kualitas antarmuka pada terminal-terminal akses dalam arsitektur Master-slave karena pada model arsitektur Client-server, komputer client merupakan *intelligent terminal*, yaitu memiliki CPU yang dapat membantu proses dalam penyajian grafis yang tinggi.

Model client-server ini telah dikembangkan untuk membangun sistem jaringan yang menjadi infrastruktur utama SI (Sistem Informasi) dalam perusahaan digital. Jaringan komputer client-server ini memiliki dua model arsitektur, yaitu : *Two Tier* dan *Three Tier* (Ramakrishnan, 2004, hal 183).

II.1.2.1 Two Tier

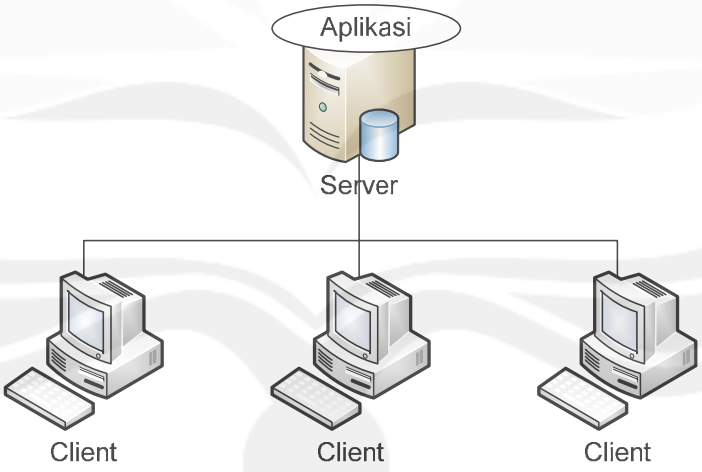
Arsitektur *single tier* memiliki kekurangan penting, yaitu dalam hal grafis antarmuka yang membutuhkan lebih banyak dukungan komputasional dari pada sekedar *dumb terminal* sederhana. Adanya komputer personal yang semakin canggih dengan harga relative murah yang dapat digunakan sebagai client dapat mengarah pada pengembangan arsitektur *Two Tier*.

Arsitektur *Two Tier* merupakan arsitektur yang disebut client-server (Ramakrishnan, 2004, hal 183), di

mana terdapat komputer sebagai client dan server yang berinteraksi melalui protokol dan media komunikasi tertentu. Model arsitektur Two Tier dapat dikelompokkan menjadi dua macam, yaitu *Thin Client-Thick Server* dan *Thick Client-Thin Server*.

II.1.2.1.1 Thin Client-Thick Server

Pada arsitektur ini, client menjalankan satu fungsi, yaitu sebagai penyaji dari tampilan aplikasi dan data yang diakses dari server. Hal ini berarti beban server lebih tinggi dan server menjadi titik kritis dari sistem jaringan tersebut, di mana server harus memberikan layanan penggunaan bersama aplikasi-aplikasi (*Aplication Server*) dan data (*Data atau File Server*) kepada semua client yang terhubung kepadanya.



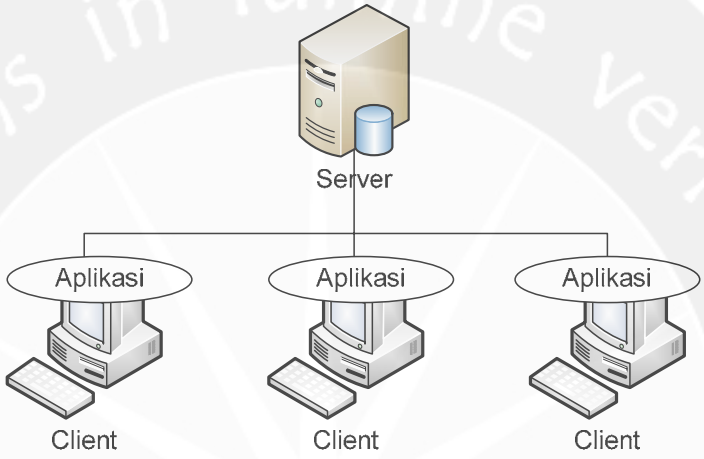
Gambar 2.3 Arsitektur Thin Client-Thick Server

II.1.2.1.2 Thick Client-Thin Server

Arsitektur ini sedikitnya memberi dua peran bagi client, di mana client tidak hanya berperan sebagai penyaji antarmuka saja, melainkan juga berfungsi mengoperasikan aplikasi. Sementara itu, server hanya

bertugas untuk mengelola data saja sehingga beban client menjadi bertambah.

Model ini diterapkan pada sistem layanan Anjungan Tunai Mandiri (ATM), di mana mesin-mesin ATM yang merupakan client berfungsi menangani antarmuka penyajian dan mengoperasikan aplikasi layanan ATM.



Gambar 2.4 Arsitektur Thick Client-Thin Server

Pada sistem ini kinerja server menjadi ringan, namun tetap harus diperhatikan perjalanan data dari client menuju server. Bisa jadi pada sisi client, aplikasi sudah mencatat terjadinya transaksi, sementara data masih dalam perjalanan belum sampai di server.

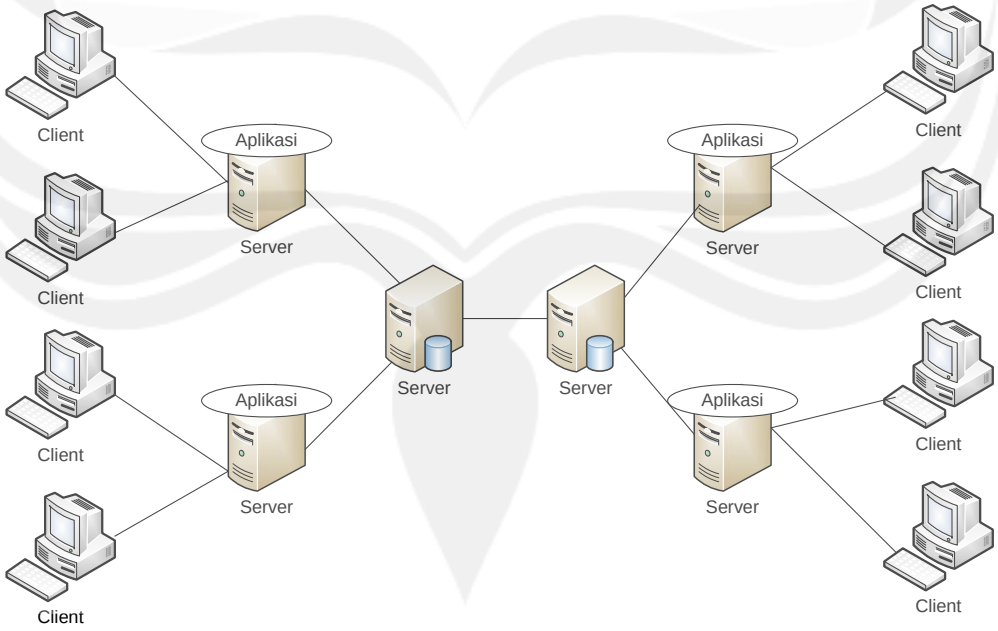
Penerapan model ini juga akan merepotkan tim teknisi untuk melakukan pemeliharaan aplikasi pada masing-masing mesin ATM karena aplikasi terletak pada masing-masing client yang mungkin saja letaknya berjauhan satu sama lain. Model ini tidak menskalakan client dalam jumlah yang besar sehingga dapat menimbulkan masalah dalam pemeliharaan dan perawatannya.

II.1.2.2 Three Tier

Arsitektur client-server ini terus dikembangkan seiring dengan perkembangan perusahaan. Kini banyak perusahaan yang mulai mengoptimalkan penggunaan sistem jaringan Internet, yang memungkinkan untuk menyediakan layanan transaksi selama 24 jam dengan skala pelanggan yang sangat luas hingga mancanegara.

Untuk membangun sistem bisnis berbasis Web yang dikenal dengan istilah e-Business, perlu dilakukan pemisahan peran server yang menangani aplikasi dan basis data agar kinerja server dapat optimal.

Three Tier merupakan arsitektur client-server yang memisahkan antara data (*Data Management Tier*), aplikasi (*Middle Tier*) dan penyajian (*Presentation Tier*) (Ramakrishnan, 2004, hal 184), sebagaimana tampak pada gambar berikut ini.



Gambar 2.5 Arsitektur Three Tier

a. **Data Management Tier** merupakan komputer server yang dikhususkan untuk menangani pengelolaan basis data.
b. **Middle Tier** merupakan server yang dikhususkan untuk menangani aplikasi-aplikasi dimana prosedur-prosedur dan perhitungan-perhitungan yang kompleks dieksekusi di sini.
c. **Presentation Tier** merupakan komputer client yang menjadi antarmuka bagi pengguna untuk memasukkan data, mengajukan permintaan layanan kepada server, dan melihat hasilnya.

Arsitektur ini memiliki sejumlah keuntungan, antara lain masing-masing *tier* akan beroperasi dengan stabilitas yang tinggi karena beban terbagi secara merata. Model ini juga memungkinkan di mana pada masing-masing *tier* diterapkan *platform* yang berbeda (Ramakrishnan, 2004, hal 186). Proses modifikasi pada satu *tier* tidak akan memengaruhi *tier* yang lain.

Sementara itu, pada *Middle Tier* dapat dikembangkan berbagai macam aplikasi yang tersebar dlam beberapa server sesuai kebutuhan yang akan diakses oleh client sehingga tidak terjadi kemacetan.

II.1.2.3 Client

Client dalam ilmu komputer adalah sebuah aplikasi atau sistem yang mengakses sebuah sistem layanan yang berada di sistem atau komputer lain yang dikenal dengan server melalui jaringan komputer. Istilah ini pertama kali diaplikasikan ke perangkat tambahan yang di waktu itu tidak dapat menjalankan programnya sendiri, tetapi

dapat berinteraksi dengan komputer lain melalui jaringan.

II.1.2.4 Server

Server dalam ilmu komputer adalah sebuat aplikasi atau sistem yang menyediakan layanan kepada program komputer yang lain yang berada di dalam komputer lain atau sama. Komputer yang secara fisik menjalankan program server juga sering kali disebut sebagai server.

II.2 Protokol Jaringan

Dalam sistem jaringan, semua peralatan telah saling terhubung secara fisik satu sama lain. Namun, agar peralatan-peralatan tersebut dapat saling berkomunikasi, diperlukan protokol kamunikasi yang merupakan sekumpulan aturan yang mendefinisikan beberapa fungsi seperti pembuatan hubungan, mengirim pesan, data, informasi atau berkas; yang harus dipenuhi oleh pengirim dan penerima agar suatu sesi komunikasi data dapat berlangsung dengan baik dan benar (Dharma Oetomo, 2004).

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

Protokol jaringan perlu diutamakan pada penggunaan standar teknis, untuk menspesifikasi bagaimana membangun komputer atau menghubungkan peralatan

perangkat keras. Protokol jaringan secara umum digunakan pada komunikasi *real-time* dimana standar digunakan untuk mengatur struktur dari informasi untuk penyimpanan jangka panjang.

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

a. Melakukan deteksi adanya koneksi fisik atau ada tidaknya komputer atau mesin lainnya
b. Melakukan metode jabat-tangan (*handshaking*)
c. Negosiasi berbagai macam karakteristik hubungan
d. Bagaimana mengawali dan mengakhiri suatu pesan
e. Bagaimana format pesan yang digunakan
f. Yang harus dilakukan saat terjadi kerusakan pesan atau pesan yang tidak sempurna
g. Mendeteksi hubungan jaringan terputus yang tidak diharapkan dan langkah-langkah yang dilakukan selanjutnya
h. Mengakhiri suatu koneksi

Untuk memudahkan memahami protokol jaringan kita harus mengerti Model OSI. Dalam Model OSI terdapat 7 layer dimana masing-masing layer mempunyai jenis protokol sesuai dengan peruntukannya. Protokol-protokol jaringan yang umum digunakan adalah sebagai berikut :

a. IP (Internet Protocol)
b. UDP (User Datagram Protocol)
c. TCP (Transmission Control Protocol)
d. DHCP (Dynamic Host Configuration Protocol)
e. HTTP (Hypertext Transfer Protocol)

f. FTP (File Transfer Protocol)
g. Telnet (Telnet Remote Protocol)
h. SSH (Secure Shell Remote Protocol)
i. POP3 (Post Office Protocol 3)
j. SMTP (Simple Mail Transfer Protocol)
k. IMAP (Internet Message Access Protocol)
l. SOAP (Simple Object Access Protocol)

II.2.1 TCP/IP

TCP/IP (*Transmission Control Protocol/Internet Protocol*) adalah standar komunikasi data yang digunakan oleh komunitas internet dalam proses tukar-menukar data dari satu komputer ke komputer lain di dalam jaringan Internet. Protokol ini tidaklah dapat berdiri sendiri, karena memang protokol ini berupa kumpulan protokol (*protocol suite*). Protokol ini juga merupakan protokol yang paling banyak digunakan saat ini. Data tersebut diimplementasikan dalam bentuk perangkat lunak (software) di sistem operasi. Istilah yang diberikan kepada perangkat lunak ini adalah TCP/IP stack

Protokol TCP/IP dikembangkan pada akhir dekade 1970-an hingga awal 1980-an sebagai sebuah protokol standar untuk menghubungkan komputer-komputer dan jaringan untuk membentuk sebuah jaringan yang luas (WAN) (Andrew S. Tanenbaum, 2002). TCP/IP merupakan sebuah standar jaringan terbuka yang bersifat independen terhadap mekanisme transport jaringan fisik yang digunakan, sehingga dapat digunakan di mana saja. Protokol ini menggunakan skema pengalamatan yang sederhana yang disebut sebagai alamat IP (*IP Address*) yang mengizinkan hingga beberapa ratus juta komputer

untuk dapat saling berhubungan satu sama lainnya di Internet. Protokol ini juga bersifat *routable* yang berarti protokol ini cocok untuk menghubungkan sistem-sistem berbeda (seperti Microsoft Windows dan keluarga UNIX) untuk membentuk jaringan yang heterogen.

II.2.1.1 Arsitektur

Arsitektur TCP/IP tidaklah berbasis model referensi tujuh lapis OSI, tetapi menggunakan model referensi DARPA. Seperti diperlihatkan dalam diagram, TCP/IP mengimplementasikan arsitektur berlapis yang terdiri atas empat lapis. Empat lapis ini, dapat dipetakan (meski tidak secara langsung) terhadap model referensi OSI. Empat lapis ini, kadang-kadang disebut sebagai *DARPA Model*, *Internet Model*, atau *DoD Model*, mengingat TCP/IP merupakan protokol yang awalnya dikembangkan dari proyek ARPANET yang dimulai oleh Departemen Pertahanan Amerika Serikat (Ronda Hauben, 2007).

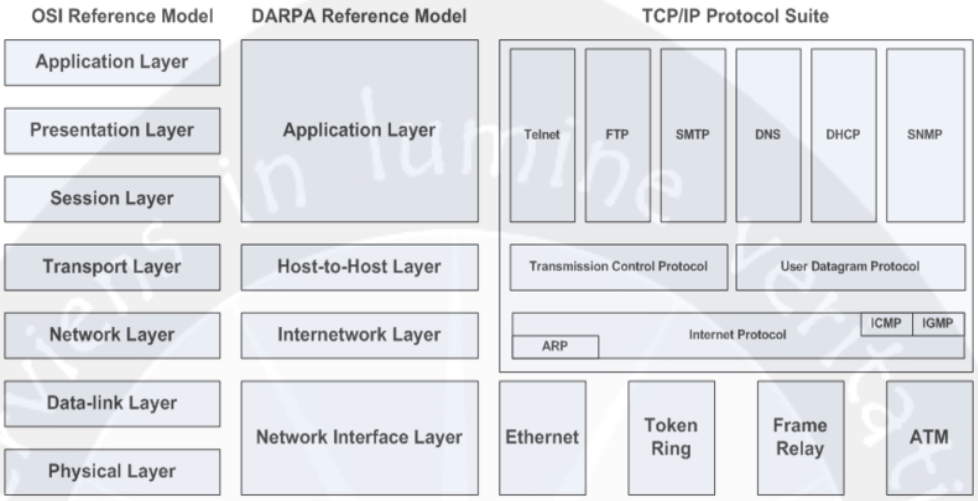
Setiap lapisan yang dimiliki oleh kumpulan protokol (*protocol suite*) TCP/IP diasosiasikan dengan protokolnya masing-masing. Protokol utama dalam protokol TCP/IP adalah sebagai berikut:

a. Protokol lapisan aplikasi: bertanggung jawab untuk menyediakan akses kepada aplikasi terhadap layanan jaringan TCP/IP. Protokol ini mencakup protokol *Dynamic Host Configuration Protocol* (DHCP), *Domain Name System* (DNS), Hypertext Transfer Protocol (HTTP), *File Transfer Protocol* (FTP), *Telnet*, *Simple Mail Transfer Protocol* (SMTP), *Simple Network*

Management Protocol (SNMP), dan masih banyak protokol lainnya. Dalam beberapa implementasi stack protokol, seperti halnya Microsoft TCP/IP, protokol-protokol lapisan aplikasi berinteraksi dengan menggunakan antarmuka Windows Sockets (Winsock) atau NetBIOS over TCP/IP (NetBT).

b. Protokol lapisan antar-*host*: berguna untuk membuat komunikasi menggunakan sesi koneksi yang bersifat *connection-oriented* atau *broadcast* yang bersifat *connectionless*. Protokol dalam lapisan ini adalah Transmission Control Protocol (TCP) dan User Datagram Protocol (UDP).
c. Protokol lapisan *internetwork*: bertanggung jawab untuk melakukan pemetaan (*routing*) dan enkapsulasi paket-paket data jaringan menjadi paket-paket IP. Protokol yang bekerja dalam lapisan ini adalah *Internet Protocol* (IP), *Address Resolution Protocol* (ARP), Internet Control Message Protocol (ICMP), dan Internet Group Management Protocol (IGMP).
d. Protokol lapisan antarmuka jaringan: bertanggung jawab untuk meletakkan frame-frame jaringan di atas media jaringan yang digunakan. TCP/IP dapat bekerja dengan banyak teknologi transport, mulai dari teknologi transport dalam LAN (seperti halnya Ethernet dan Token Ring), MAN dan WAN (seperti halnya dial-up modem yang berjalan di atas Public Switched Telephone Network (PSTN), *Integrated Services Digital*

Network (ISDN), serta *Asynchronous Transfer Mode* (ATM)).



Gambar 2.6 Arsitektur TCP/IP diperbandingkan dengan DARPA Reference Model dan OSI Reference Model

II.2.1.2 Pengalamatan

Protokol TCP/IP menggunakan dua buah skema pengalamatan yang dapat digunakan untuk mengidentifikasikan sebuah komputer dalam sebuah jaringan atau jaringan dalam sebuah internetwork, yakni sebagai berikut:

a. Pengalamatan IP: yang berupa alamat logis yang terdiri atas 32-bit (empat oktet berukuran 8-bit) yang umumnya ditulis dalam format www.xxx.yyy.zzz. Dengan menggunakan *subnet mask* yang diasosiasikan dengannya, sebuah alamat IP pun dapat dibagi menjadi dua bagian, yakni *Network Identifier* (NetID) yang dapat mengidentifikasikan jaringan lokal dalam sebuah *internetwork* dan *Host identifier* (HostID) yang

dapat mengidentifikasikan host dalam jaringan tersebut. Sebagai contoh, alamat 205.116.008.044 dapat dibagi dengan menggunakan subnet mask 255.255.255.000 ke dalam *Network ID* 205.116.008.000 dan *Host ID* 44. Alamat IP merupakan kewajiban yang harus ditetapkan untuk sebuah *host*, yang dapat dilakukan secara manual (statis) atau menggunakan *Dynamic Host Configuration Protocol* (DHCP) (dinamis).

b. *Fully qualified domain name* (FQDN): Alamat ini merupakan alamat yang direpresentasikan dalam nama alfanumerik yang diekspresikan dalam bentuk <*nama_host*>.<*nama_domain*>, di mana <nama_domain> mengindentifikasikan jaringan di mana sebuah komputer berada, dan <nama_host> mengidentifikasikan sebuah komputer dalam jaringan. Pengalamatan FQDN digunakan oleh skema penamaan domain *Domain Name System* (DNS). Sebagai contoh, alamat FQDN id.wikipedia.org merepresentasikan sebuah host dengan nama "id" yang terdapat di dalam domain jaringan "wikipedia.org". Nama domain wikipedia.org merupakan *second-level domain* yang terdaftar di dalam *top-level domain* .org, yang terdaftar dalam root DNS, yang memiliki nama "." (titik). Penggunaan FQDN lebih bersahabat dan lebih mudah diingat ketimbang dengan menggunakan alamat IP. Akan tetapi, dalam TCP/IP, agar komunikasi dapat berjalan, FQDN harus diterjemahkan terlebih dahulu (proses penerjemahan ini disebut sebagai resolusi nama)

ke dalam alamat IP dengan menggunakan *server* yang menjalankan DNS, yang disebut dengan *Name Server* atau dengan menggunakan berkas *hosts* (/etc/hosts atau %systemroot%\system32\drivers\etc\hosts) yang disimpan di dalam mesin yang bersangkutan.

II.2.1.3 Layanan

Berikut ini adalah layanan tradisional yang dapat berjalan di atas protokol TCP/IP (Douglas E. Comer, 2005) :

a. Pengiriman berkas (*file transfer*). *File Transfer Protocol* (FTP) memungkinkan pengguna komputer yang satu untuk dapat mengirim ataupun menerima berkas ke sebuah host di dalam jaringan. Metode otentikasi yang digunakannya adalah penggunaan nama pengguna (*user name*) dan *[[password*]], meskipun banyak juga FTP yang dapat diakses secara anonim (*anonymous*), alias tidak ber*password*. (Keterangan lebih lanjut mengenai FTP dapat dilihat pada RFC 959.)
b. *Remote login*. *Network terminal Protocol* (*telnet*) memungkinkan pengguna komputer dapat melakukan *log in* ke dalam suatu komputer di dalam suatu jaringan secara jarak jauh. Jadi hal ini berarti bahwa pengguna menggunakan komputernya sebagai perpanjangan tangan dari komputer jaringan tersebut. (Keterangan lebih lanjut mengenai Telnet dapat dilihat pada RFC 854 dan RFC 855.)

c. *Computer mail*. Digunakan untuk menerapkan sistem surat elektronik. (Keterangan lebih lanjut mengenai e-mail dapat dilihat pada RFC 821 RFC 822.)
d. *Network File System* (NFS). Pelayanan akses berkas-berkas yang dapat diakses dari jarak jauh yang memungkinkan client-client untuk mengakses berkas pada komputer jaringan, seolah-olah berkas tersebut disimpan secara lokal. (Keterangan lebih lanjut mengenai NFS dapat dilihat RFC 1001 dan RFC 1002.)
e. *Remote execution*. Memungkinkan pengguna komputer untuk menjalankan suatu *program* tertentu di dalam komputer yang berbeda. Biasanya berguna jika pengguna menggunakan komputer yang terbatas, sedangkan ia memerlukan sumber yg banyak dalam suatu sistem komputer. Ada beberapa jenis *remote execution*, ada yang berupa perintah-perintah dasar saja, yaitu yang dapat dijalankan dalam system komputer yang sama dan ada pula yg menggunakan sistem *Remote Procedure Call* (RPC), yang memungkinkan program untuk memanggil subrutin yang akan dijalankan di sistem komputer yg berbeda. (sebagai contoh dalam Berkeley UNIX ada perintah rsh dan rexec.)
f. *Name server* yang berguna sebagai penyimpanan basis data nama *host* yang digunakan pada Internet (Keterangan lebih lanjut dapat dilihat pada RFC 822 dan RFC 823 yang menjelaskan mengenai penggunaan protokol *name server* yang

bertujuan untuk menentukan nama *host* di Internet.

II.3 Socket Programming

Di dalam ilmu komputer, pemrograman jaringan (*network programming*) sangat identik dengan pemrograman socket (*socket programming)* atau pemograman client-server, yang melibatkan penulisan program komputer yang berkomunikasi dengan program lain lewat jaringan komputer. Program atau proses yang memulai komunikasi disebut proses client, dan program yang menunggu komunikasi dimulai disebut proses server. Proses client dan server bersama-sama membentuk sebuah sistem terdistribusi. Komunikasi antara client dan server dapat menggunakan salah satu cara yaitu : *connection-oriented* (seperti sirkuit maya TCP atau *session* yang dibangun), atau *connectionless* (berdasarkan pada datagram UDP).

Sebuah program dapat bertindak sebagai client ataupun server jika berbasis komunikasi *peer to peer*.

Socket biasanya diimplementasikan oleh sebuat library API seperti Berkeley sockets yang pertama kali dikenalkan pada tahun 1983. Kebanyakan implementasi-implementasi berdasarkan Berkeley sockets, contohnya Winsock yang dikenalkan pada tahun 1991. Implementasi socket API lain yang ada adalah STREAMS based Transport Layer Interface (TLI).

Dibawah ini adalah contoh dari fungsi atau cara khas yang disediakan oleh library API (http://en.wikipedia.org/) :

a. socket() ,untuk membuat socket baru dengan tipe socket tertentu, diidentifikasi dengan sebuah nilai integer dan mengalokasikan resource sistem untuknya.
b. bind() ,biasanya digunakan pada sisi server dan mengasosiasikan sebuah socket dengan struktur alamat socket, contohnya menspesifikasikan nomor port local dan alamat IP.
c. listen() ,digunakan pada sisi server, menyebabkan socket TCP yang ada memasuki kondisi *listening*.
d. connect() ,digunakan pada sisi client, dan memberikan nomor port lokal yang dalam kondisi *free*. Dalam kasus socket TCP, perintah ini menghasilkan sebuah usaha untuk membangun sebuah koneksi TCP yang baru.
e. accept() ,digunakan pada sisi server. Perintah ini menerima sebuah usaha yang diterima untuk membuat sebuah koneksi TCP yang baru dari *remote* client dan membuat socket baru yang terasosiasi dengan pasangan alamat socket dari koneksi ini.
f. send() dan recv(), atau write() dan read(), atau recvfrom() dan sendto() digunakan untuk mengirim dan menerima data kepada atau dari sebuah *remote* socket.
g. close() ,perintah ini akan membuat sistem melepaskan *resource* yang teralokasi kepada sebuah socket. Dalam kasus TCP, koneksi akan dihentikan.

II.4 Windows API

Windows API (*Windows Application Programming Interface*), yang sering disebut sebagai WinAPI atau Windows API adalah sekumpulan antarmuka pemrograman aplikasi yang dibuat oleh Microsoft dalam inti sistem operasi Microsoft Windows buatannya. Semua program Windows, kecuali program konsol, harus berinteraksi dengan Windows API tanpa melihat dengan bahasa apa ia dibuat. Akses terhadap elemen sistem operasi yang lebih rendah, seperti halnya yang dibutuhkan oleh *device driver*, tidak disediakan oleh Windows API, tapi disediakan oleh *Windows Driver Foundation* atau *Native API* dalam versi-versi baru Windows.

II.4.1 Versi-versi Windows API

Hampir pada setiap peluncuran versi baru Windows, Microsoft memperkenalkan API baru terhadap Windows API. Meskipun demikian, nama dari panggilan API tersebut tetap dipertahankan dan konsisten antara satu versi dengan versi yang lainnya, dan perubahan nama pun mungkin dilakukan jika memang terjadi di sana perubahan besar-besaran pada platform Windows itu sendiri. Microsoft pun kemudian akan mengubah nama keluarga Win32 API yang digunakan saat ini menjadi Windows API, dan membuatnya dapat digunakan oleh semua versi API sistem operasi Windows (http://msdn.microsoft.com/).

Win16 API

Windows 16 API atau Win16 API merupakan API yang digunakan pertama kali pada versi Windows 16-bit. Pada awalnya, Win16 API disebut dengan Windows API, tapi

kemudian diubah menjadi Win16 dalam usaha Microsoft untuk membedakannya dengan versi Windows API yang lebih baru yang berjalan pada Windows 32-bit, Win32 API. Fungsi-fungsi Win16 API umumnya terdapat di dalam berkas-berkas inti sistem operasi: kernel.exe (atau krnl286.exe pada Windows for 286 atau krnl386.exe pada Windows yang berjalan pada Enhanced 386), user.exe, dan gdi.exe. Meskipun memiliki ekstensi EXE, sebenarnya mereka bukanlah berkas yang dapat dieksekusi (*executable*), melainkan mereka adalah DLL (*Dynamic Linking Library*).

Win32 API

Win32 API merupakan antarmuka pemrograman yang terdapat di dalam sistem operasi Windows 32-bit modern. Seperti halnya Win16 API, Win32 API juga sama mengimplementasikan fungsi-fungsi di dalam DLL sistem operasi. DLL inti yang dimiliki oleh Win32 API antara lain kernel32.dll, user32.dll, dan gdi32.dll. Win32 pertama kali muncul pada tahun 1993, saat Windows NT diluncurkan. Windows 95 juga menggunakan Win32 API, dan pada awalnya dikenal dengan sebutan Win32c, di mana huruf "c" di sana merujuk kepada "*compatibility*", tapi istilah ini akhirnya ditinggalkan oleh Microsoft demi konsistensi nama "Win32".

Dalam Windows NT 4.0 dan para penerusnya (termasuk di antaranya versi-versi terbaru Windows), panggilan-panggilan Win32 dieksekusi oleh dua modul, yakni csrss.exe (Client/Server Runtime SubSystem) di dalam modus pengguna dan Win32K.sys di dalam modus kernel miliknya.

Win32s API

Win32s API merupakan sebuah ekstensi untuk keluarga Windows 3.1x yang mengimplementasikan sekumpulan kecil dari Win32 API untuk sistem-sistem tersebut, yang merupakan sistem operasi 16-bit. Huruf "s" di sana merupakan singkatan dari "subset."

Win32 for 64-bit Windows

Win32 for 64-bit Windows, yang sebelumnya dikenal dengan sebutan Win64, merupakan sebuah versi Windows API yang ditargetkan untuk digunakan oleh Windows versi 64-bit, yakni Windows XP Professional x64 Edition dan Windows Server 2003 x64 Edition (untuk prosesor-prosesor x86-64) dan Windows XP 64-bit Edition dan Windows Server 2003 for Itanium-series (untuk prosesor-prosesor IA-64). Dengan kemunculan Win64, Windows NT pun akhirnya masuk ke dalam pasar komputasi 64-bit, dan kompatibilitas aplikasi 32-bit pun masih terjaga. Akan tetapi, memang semua *pointer* memori dialamatkan dengan menggunakan alamat 64-bit, sehingga kode sumber program harus dicek ulang untuk melihat apakah ada masalah kompatibilitas dengan *pointer* aritmetika 64-bit dan jika perlu ditulis ulang. Tidak ada penambahan fungsi-fungsi baru yang spesifik ditambahkan ke dalam Windows versi 64-bit.

II.4.2 Fungsionalitas yang Ditawarkan

Fungsionalitas yang ditawarkan oleh Windows API dapat digolongkan ke dalam delapan kategori (http://msdn.microsoft.com/) :

a. ***Base Services*** : Menyediakan akses terhadap beberapa sumber daya fundamental yang tersedia

di dalam sebuah sistem yang menjalankan sistem operasi Microsoft Windows. Fungsi-fungsi ini terdapat dalam kernel.exe, krnl286.exe, atau krnl386.exe (dalam Windows 16-bit) dan kernel32.dll serta advapi32.dll dalam Windows 32-bit. Subkomponen yang terdapat pada golongan ini antara lain:

1. *File system* (sistem berkas): FAT12, FAT16, FAT32, CDFS, UDFS dan NTFS
2. *Devices*
3. *Process and threads*
4. *Error handling*.

b. ***Advanced Services*** *:* Menyediakan akses terhadap fungsionalitas yang berada di luar kernel sistem operasi Windows. Fungsionalitas yang termasuk di dalam kategori ini adalah akses *registry* Windows, *shutdown* (*turn-off*, *restart*, *hibernate*, atau *standby*) Windows, manajemen *Windows service*, hingga manajemen akun pengguna. Fungsi-fungsi ini terdapat di dalam advapi32.dll, dan hanya terdapat pada versi Windows 32-bit.

c. ***Graphics Device Interface*** **(GDI)** : Menyediakan fungsionalitas untuk mengeluarkan tampilan grafik ke monitor, *printer* dan beberapa perangkat keluaran lainnya. Dalam Windows 16-bit, GDI ditangani oleh GDI.EXE, atau GDI32.DLL dalam Windows 32-bit.

d. ***User Interface*** : Menyediakan fungsionalitas untuk membuat dan mengatur layar jendela dan sebagian besar kontrol dasar, seperti tombol

(*button*), baris geser (*scroll bar*), hingga menerima *input* dari pengguna melalui *keyboard* atau *mouse*. Dalam Windows 16-bit, fungsionalitas *user interface* ditangani oleh USER.EXE, atau User32.DLL dalam Windows 32-bit. Saat Windows XP dirilis oleh Microsoft, kontrol-kontrol dasar dari Windows digabungkan ke dalam bagian registry ComCtl32.DLL, bersama-sama dengan *Common Control Library* (CCL) yang dimilikinya.

e. ***Common Dialog Box Library*** : Menyediakan fungsionalitas untuk membuat kotak dialog untuk membuka dan menutup berkas, memilih warna, huruf (*font*) dan lain sebagainya. Antarmuka ini terdapat pada sebuah berkas dengan nama Commdlg.dll pada Windows 16-bit, atau Comdlg32.dll pada Windows 32-bit. Meskipun demikian, fungsionalitas ini dimasukkan dalam kategori *User Interface*.

f. ***Common Control Library*** : Mengizinkan aplikasi agar dapat mengakses beberapa kontrol tingkat lanjut yang disediakan oleh sistem operasi, seperti halnya baris status (*status bar*), baris kemajuan (*progress bar*), baris peralatan (*toolbar*), dan juga tabulasi (*tab*). Pustaka untuk ini disediakan oleh sebuah DLL yang disebut dengan CommCtrl.DLL dalam sistem operasi Windows 16-bit, atau ComCtl32.DLL dalam Windows 32-bit. Ini juga dikelompokkan ke dalam kategori User Interface di dalam API.

g. ***Windows Shell*** : Komponen Windows API yang mengizinkan aplikasi untuk mengakses fungsionalitas yang disediakan oleh *shell* sistem operasi, dan juga mengubah atau bahkan meningkatkannya. Komponen ini terdapat di dalam DLL Shell.DLL dalam Windows 16-bit, sementara dalam Windows 32-bit terdapat di dalam Shell32.DLL. Windows 95 juga menawarkan sebuah DLL yang disebut sebagai ShlWApi.DLL. Ini juga dikelompokkan ke dalam kategori *User Interface* di dalam API.

h. ***Network Services*** : Mengizinkan aplikasi agar dapat mengakses kemampuan jaringan di dalam sistem operasi. Komponen ini memiliki beberapa subkomponen, yakni:

   1. *Network Basic Input Output System* (NetBIOS)
   2. *Windows Socket* (Winsock)
   3. *Network Dynamic Data Exchange* (NetDDE)
   4. *Remote Procedure Call* (RPC)
   5. dan masih banyak yang lainnya.

II.5 Basis Data

Saat ini peranan basis data sangat penting didalam pengembangan suatu sistem informasi. Pemrosesan basis data menjadi perangkat andal yang sangat diperlukan oleh berbagai instansi atau perusahaan. Basis data akan mempercepat proses perolehan informasi, dan juga dapat meningkatkan pelayanan dari badan yang terkait. Data merupakan fakta mengenai objek, orang dan lain-lain.

Data dinyatakan dengan nilai tertentu, berbentuk angka, maupun simbol-simbol.

Basis data adalah suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data dengan cara-cara tertentu sehingga mudah untuk digunakan atau ditampilkan kembali; dapat digunakan oleh satu atau lebih program aplikasi secara optimal; data disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya; data disimpan sedemikian rupa sehingga penambahan, pengambilan dan modifikasi data dapat dilakukan dengan mudah dan terkontrol.

Secara tradisional, data diorganisasikan kedalam suatu hirarki yang terdiri atas:

1. Elemen Data

   Elemen data adalah satuan terkecil yang tidak dapat dipecahkan lagi sebagai unit lain yang bermakna.

2. Rekaman

   Rekaman adalah gabungan sejumlah elemen data yang saling terkait.

3. Berkas (File)

   Himpunan seluruh rekaman yang bertipe sama membentuk sebuah berkas.

Perkembangan teknologi basis data sendiri tidak terlepas dari perkembangan perangkat keras dan perangkat lunak komputer. Perkembangan teknologi jaringan komputer dan komunikasi data adalah salah satu penyumbang kemajuan penerapan basis data, yang kemudian melahirkan sistem basis data yang terdistribusi.

II.5.1 Database Management System (DBMS)

DBMS merupakan perangkat lunak yang memungkinkan user mendefinisikan, menciptakan dan memanajemen basis data. Fungsi utama dari DBMS adalah:

1. Mendefinisikan basis data dengan cara mendefinisikan tipe data, struktur dan constraint.
2. Membangun sebuah basis data yaitu proses untuk menyimpan data itu sendiri ke media penyimpan.
3. Memanipulasi basis data yaitu suatu proses untuk melakukan query terhadap data tertentu di dalam basis data dan memperbaharui basis data.

Dalam perkembangan selanjutnya untuk membuat derajat kebebasan data yang tinggi sehingga program aplikasi tidak harus terpengaruh oleh perubahan representasi data internal dan menyediakan landasan yang kokoh yang berhubungan dengan masalah yang berkaitan semantik data, konsistensi data dan redundansi data serta untuk memungkinkan pengembangan bahasa manipulasi data yang bersifat *set-oriented* dikembangkan Relational Database Management System (RDBMS). Di mana model relasional ini berdasarkan konsep relasi dalam matematika yang secara fisik direpresentaikan dalam bentuk tabel. Sebuah relasi adalah sebuah tabel dengan kolom dan baris. Informasi/data disimpan dalam tabel dua dimensi berupa: baris data (*row/record*) dan kolom (*column/field*)

Salah satu RDBMS yang terkenal dan digunakan banyak orang adalah MySQL (http://www.mysql.com/). MySQL didistribusikan secara gratis dibawah lisensi General Public License (GPL). Dimana setiap orang bebas

untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat closed source atau komersial.

II.6 Microsoft .NET Framework

Microsoft .NET Framework adalah sebuah komponen yang dapat ditambahkan ke sistem operasi Microsoft Windows atau telah terintegrasi ke dalam Windows (mulai dari Windows Server 2003 dan versi-versi Windows terbaru) (http://msdn2.microsoft.com/). Kerangka kerja ini menyediakan sejumlah besar solusi-solusi program untuk memenuhi kebutuhan-kebutuhan umum suatu program baru, dan mengatur eksekusi program-program yang ditulis secara khusus untuk framework ini. .NET Framework adalah kunci penawaran utama dari Microsoft, dan dimaksudkan untuk digunakan oleh sebagian besar aplikasi-aplikasi baru yang dibuat untuk platform Windows. Pada dasarnya, .NET Framework memiliki 2 komponen utama : CLR dan .NET Framework Class Library.

Program-program yang ditulis untuk .NET Framework dijalankan pada suatu lingkungan software yang mengatur persyaratan-persyaratan *runtime* program. *Runtime environment* ini, yang juga merupakan suatu bagian dari .NET Framework, dikenal sebagai *Common Language Runtime* (CLR). CLR menyediakan penampilan dari *application virtual machine*, sehingga para programmer tidak perlu mengetahui kemampuan CPU tertentu yang akan menjalankan program. CLR juga menyediakan layanan-layanan penting lainnya seperti jaminan keamanan, pengaturan memori, *garbage collection* dan *exception handling* / penanganan kesalahan pada saat *runtime*.

Class library dan CLR ini merupakan komponen inti dari .NET Framework. Kerangka kerja itu pun dibuat sedemikian rupa agar para programmer dapat mengembangkan program komputer dengan jauh lebih mudah, dan juga untuk mengurangi kerawanan aplikasi dan juga komputer dari beberapa ancaman keamanan.

CLR adalah turunan dari CLI (Common Language Infrastructure) yang saat ini merupakan standar ECMA. Untuk keterangan lebih lanjut, silakan mengunjungi situs ECMA atau kunjungi sumber pranala di bawah artikel ini.

Solusi-solusi program pembentuk *class library* dari .NET Framework mengcover area yang luas dari kebutuhan program pada bidang user interface, pengaksesan data, koneksi basis data, kriptografi, pembuatan aplikasi berbasis web, algoritma numerik, dan komunikasi jaringan. Fungsi-fungsi yang ada dalam *class library* dapat digabungkan oleh *programmer* dengan kodenya sendiri untuk membuat suatu program aplikasi baru.

Pada berbagai literatur dan referensi di Internet, .NET Framework seringkali disingkat menjadi .NET saja.

.NET seringkali juga dapat diartikan sebagai platform, yang merupakan suatu lingkungan terpadu untuk pengembangan dan eksekusi untuk berbagai macam bahasa pemrograman dan kumpulan library untuk bekerja sama membuat dan menjalankan aplikasi berbasis Windows yang lebih mudah untuk dibuat, diatur, didistribusikan, dan diintegrasikan dengan sistem jaringan lain.

Dalam perkembangannya, .NET seringkali dikaitkan pula dengan versi Visual Studio yang sesuai dengan dukungan versi yang bersangkutan untuk pengembangan

aplikasi. Berikut ini versi .NET dan versi Visual Studio yang terkait:

a. .NET 1.0 dan Visual Studio .NET (atau seringkali disebut juga dengan Visual Studio .NET 2002)
b. .NET 1.1 dan Visual Studio .NET 2003
c. .NET 2.0 dan Visual Studio 2005
d. .NET 3.0 dan Visual Studio 2005 dengan tambahan addin untuk WPF, WCF dan WF
e. .NET 3.5 dan Visual Studio 2008

.NET 2.0, 3.0 dan 3.5 memiliki CLR yang sama. Dengan demikian, struktur IL juga sama. Adapun fasilitas penambahan kata kunci pemrograman seperti pada LINQ yang sebenarnya lebih mengarah sebagai fitur bahasa pemrograman (*programming language feature*) sehingga bukan merupakan fitur CLR.