BAB 2
LANDASAN TEORI

2.1 Jaringan komputer

Jaringan komputer adalah sekumpulan komputer serta perangkat-perangkat lain pendukung komputer yang saling terhubung dalam suatu kesatuan. Media jaringan komputer dapat melalui kabel-kabel atau tanpa kabel sehingga memungkinkan pengguna jaringan komputer dapat saling melakukan pertukaran informasi, seperti dokumen dan data, dapat juga melakukan pencetakan pada printer yang sama dan bersama-sama memakai perangkat keras dan perangkat lunak yang terhubung dengan jaringan. Setiap komputer, ataupun perangkat-perangkat yang terhubung dalam suatu jaringan disebut dengan *node*. Dalam sebuah jaringan komputer dapat mempunyai dua, puluhan, ribuan atau bahkan jutaan *node*.

2.1.1. Tipe Jaringan Komputer

Jaringan komputer berdasarkan ruang lingkup dan jangkauan dibagi menjadi tiga kelompok:

1. Jaringan LAN (*Local Area Network*)

   Jaringan LAN (*Local Area Network*) adalah jaringan yang menghubungkan beberapa komputer dalam satu lokal area saja (biasanya dalam satu gedung atau antargedung). Biasanya digunakan di dalam rumah, perkantoran, perindustrian, universitas atau akademik, rumah sakit dan daerah yang sejenis.

Pada jaringan LAN (*Local Area Network***)**, kecepatan transmisi data akan mencapai satu sampai 100 megabit per detik. Berbagai tipe LAN *(Local Area Network)* telah dibangun dan diinstalasi, namun beberapa tipe menjadi lebih dominan dari yang lain. Sistem LAN (*Local Area Network*) yang sering digunakan adalah sistem *Ethernet* yang dikembangkan oleh perusahaan Xerox. Penggunaan *nodeintermediete* (seperti repeater, bridge, dan switch) memungkinkan LAN *(Local Area Network)* terkoneksi membentuk jaringan yang lebih luas. LAN *(Local Area Network)* juga dapat terkoneksi ke LAN *(Local Area Network)*, WAN (*Wide Area Network*), atau MAN (*Metropolitan Area Network*) lain menggunakan router. Secara garis besar, LAN (*Local Area Network*)adalah sebuah jaringan komunikasi yang memiliki ciri sebagai berikut :

1. Bersifat lokal (misal, satu gedung atau antargedung).
2. Dikontrol oleh satu kekuasaan administratif.
3. Pengguna dalam sebuah LAN (*Local Area Network*) dianggap dapat dipercaya.
4. Mempunyai kecepatan yang tinggi dan data dalam semua komputer selalu di-*sharing.*

2. Jaringan MAN (*Metropolitan Area Network*)

MAN (*Metropolitan Area Network*) merupakan jaringan yang menghubungkan beberapa jaringan komputer dalam wilayah yang lebih luas. MAN (*Metropolitan Area Network*) biasanya digunakan oleh sebuah

korporasi jaringan komputer dalam satu kota, antarkampus, dalam suatu universitas, dan lain-lain. Tiga ciri khusus yang membedakan *LAN (Local Area Network)*, MAN (*Metropolitan Area Network*) dan WAN (*Wide Area Network*), yaitu:

1. Luas Jaringan MAN (*Metropolitan Area Network*) merupakan pertengahan antara luas jaringan *LAN (Local Area Network)* dengan jaringan WAN (*Wide Area Network*). MAN (*Metropolitan Area Network*) biasanya mempunyai luas jaringan antara 5 sampai 10 km dan mengkoneksikan komputer dalam suatu kota.
2. MAN (*Metropolitan Area Network*) seperti juga WAN (*Wide Area Network*) biasanya dimiliki oleh beberapa organisasi. Pada MAN (*Metropolitan Area Network*), jaringan komunikasi dan peralatan biasanya dimiliki oleh *network provider* yang menjual servis kepada pengguna. Servis yang ada tergantung kepada operator MAN (*Metropolitan Area Network*) dan *performance* yang ada biasanya telah ditentukan.
3. MAN (*Metropolitan Area Network*) merupakan jaringan dengan kecepatan tinggi yang memungkinkan *sharing* data pada suatu area yang luas.
4. MAN (*Metropolitan Area Network*) dapat dikoneksikan ke jaringan lain untuk membentuk WAN (*Wide Area Network*).

Pada Internet, organisasi yang beroperasi memakai sistem jaringan ini adalah ISP *(Internet Service Provider)*. ISP *(Internet Service Provider)*

mengatur siklus komunikasi ke pengguna perorangan. Mereka juga memastikan dapat memonitor penggunaan pengguna dan pelayanan aksesnya. ISP *(Internet Service Provider)* mendapat keuntungan dari pelanggan-pelanggan jaringannya dan bertanggung jawab memenuhi kebutuhan konsumen yang telah membayarnya.

ISP *(Internet Service Provider)* sendiri membayar kepada penyedia WAN (*Wide Area Network*) atau NSP *(Network Services Provider)* untuk konektivitas yang diperlukan. Banyak ISP *(Internet Service Provider)* terkoneksi ke ISP *(Internet Service Provider)* lain atau dengan beberapa NSP *(Network Services Provider)* yang lain. Koneksi-koneksi tambahan ini terbukti jarang mengalami masalah.

3. Jaringan WAN (*Wide Area Network*)

Jaringan WAN (*Wide Area Network*) menghubungkan beberapa WAN (*Wide Area Network*) dari beberapa kota atau negara yang berbeda. WAN (*Wide Area Network*) biasanya terhubung via satelit. WAN (*Wide Area Network*) mempunyai daerah yang sangat luas dan menggunakan siklus komunikasi yang menghubungkan *node-node intermediate*. Kecepatan transmisinya beragam dari 2 Mbps, 34 Mbps, 45 Mbps, 155 Mbps, sampai dengan 625 Mbps (atau kadang-kadang lebih). Faktor khusus yang mempengaruhi desain dan *performance*-nya terletak pada siklus komunikasi, seperti jaringan telepon, satelit atau komunikasi pembawa lain yang disewanya.

Ciri dari jaringan WAN (*Wide Area Network*) adanya penekanan pada fasilitas transmisi sehingga komunikasi dapat berjalan efisien. Sangatlah penting untuk mengontrol jumlah lalu lintas data dan mencegah delay yang berlebihan. Karena topologi WAN (*Wide Area Network*) lebih komplek dari MAN (*Metropolitan Area Network*) maka algoritma rute aliran data menjadi perhatian.

Banyak jaringan WAN (*Wide Area Network*) telah dibangun seperti jaringan publik, jaringan korporasi yang besar, jaringan militer, jaringan perbankan, jaringan perdagangan *online* dan jaringan pemesanan jasa angkutan. WAN (*Wide Area Network*) menjadi sangat luas, membentang keseluruh dunia.Organisasi yang mendukung WAN (*Wide Area Network*) adalah penggunaan protokol Internet yang dikenal sebagai NSP *(Network Service Provider)*. NSP *(Network Service Provider)* merupakan bagian inti dari Internet.

WAN (*Wide AreaNetwork*) yang dikoneksikan melalui NSP *(Network Service Provider)* bersama-sama terkoneksi membentuk suatu jaringan Internet yang bersifat global .NSP *(Network Service Provider)* yang ada bervariasi dalam kecepatan transmisi data, *transit delay*, dan konektivitas yang diberikan. Keuntungan penggunaan WAN (*Wide Area Network*) adalah :

1. Luas jaringan WAN (*Wide Area Network*) dapat mencapai seluruh belahan dunia.
2. Transfer file pada tempat berjauhan dapat dikirim secara cepat melalui *e-mail* dan FTP (*File Transfer Protocol*).

2.1.2. Topologi Jaringan

Topologi jaringan adalah metode yang digunakan untuk menghubungkan komputer yang satu dengan komputer lainnya sehingga membentuk jaringan. Dalam suatu jaringan komputer jenis topologi akan mempengaruhi kecepatan komunikasi. Untuk itu, sebelum membangun sebuah jaringan komputer, perlu diketahui karakteristik dari masing-masing topologi jaringan. Ada tiga tipe jaringan komputer berdasarkan topologinya, yaitu:

1. Topologi *Bus*
2. Topologi *Star* (bintang)
3. Topologi *Ring* (cincin)

Topologi jaringan yang sering digunakan dari ketiga topologi jaringan di atas adalah tipe *star*. Selain ketiga topologi jaringan tersebut, ada juga topologi jaringan *hybrid*.Seperti *hybrid-star-bus*, jaringan *star-ring*, dan jaringan *mesh* yang mengkoneksikan berbagai macam komputer dalam jaringan.

1. Topologi *Bus*

   Pada topologi *bus*, kedua ujung jaringan harus diakhiri dengan sebuah terminator.*Barel* konektor dapat digunakan untuk memperluasnya.Jaringan hanya terdiri dari satu saluran kabel menggunakan kabel BNC (*Barel Network Cable*). Komputer yang ingin terhubung ke jaringan dapat, mengaitkan dirinya dengan men-*tap Ethernet* sepanjang kabel.

   Instalasi jaringan *Bus* sangat sederhana, murah dan maksimal terdiri atas 5-7 komputer. Kesulitan yang sering dialami adalah kemungkinan terjadi

tabrakan data karena mekanisme jaringan relatif sederhana dan jika salah satu *node* terputus maka akan mengganggu kinerja dan trafik seluruh jaringan.

2. Topologi *Star*

Pada topologi s*tar* (bintang), semua komputer mengelilingi hub atau switch pusat yang mengontrol komunikasi jaringan dan dapat berkomunikasi dengan hub atau switch yang lain. Batas jarak komputer dengan hub sekitar 100 meter. Setiap *node* pada jaringan akan berkomunikasi melalui *node* pusat atau konsentrator terlebih dahulu sebelum menuju *server*. Jaringan lebih fleksibel dan luas dibandingkan dua topologi yang lain.

Keunggulan tipe *star* adalah jika salah satu *node* putus maka tidak akan mengganggu kinerja jaringan lainnya. Kabel yang biasa yang digunakan adalah kabel UTP (*Unshielded Twisted Pair*).

3. Topologi Ring

Pada topologi *ring* (cincin), jaringan membentuk lingkaran tertutup sehingga mengesankan cincin tanpa ujung. Komputer terkoneksi satu dengan yang lain seperti sebuah cincin. Sinyal akan mengalir satu arah sehingga dapat menghindari terjadinya tabrakan paket. Namun, salah satu komputer yang putus akan tetap mempengaruhi keseluruhan jaringan.

2.1.3. IP Address

Alamat IP atau *Internet Protokol Address* adalah deretan angka biner antar 32 bit sampai 128 bit yang dipakai sebagai alamat identifikasi untuk tiap komputer *host* dalam jaringan Internet. Panjang dari angka ini adalah 32 bit

(untuk *IPv*4 atau IP versi 4), dan 12 bit (untuk IP*v*6 atau IP versi 6) yang menunjukkan alamat dari komputer tersebut pada jaringan Internet berbasis TCP *(Transport Control Protocol)/*IP*(Internet Protocol)*. Alamat IP adalah Kombinasi angka unik yang ditetapkan untuk mengindentifikasi / mengenali suatu *host* di suatu jaringan / Internet agar bisa berkomunikasi satu sama lain. Angka ini dibutuhkan oleh *software-software* Internet untuk mengakses informasi dari dan ke *host* tersebut. Setiap komputer yang ingin terhubung ke jaringan / Internet maka komputer tersebut harus memiliki alamat yang jelas agar bisa berkomunikasi satu sama lain. Pengguna Internet umumnya hanya mengenal nama*domain* dari alamat *website* yang akan dikunjungi, seperti: detik.com, ilmukomputer.com, e-dukasi.net, yahoo.com, gmail.com, dll. Tapi sebenarnya alamat-alamat tersebut merupakan konversi dari alamat komputer/jaringan berbasiskan angka yang unik. Konsep dasarnya pengalamatan pada komputer menggunakan sekumpulan angka sebanyak 32 bit (IPv4) atau 128 bit (IPv6) yang dikenal sebagai IP *Address.*

Adanya IP *Address* merupakan konsekuensi dari penerapan Internet Protokol untuk mengintegrasikan jaringan komputer Internet di dunia. Seluruh *host* (komputer) yang terhubung ke Internet dan ingin berkomunikasi memakai *TCP/*IP harus memiliki IP*Address* sebagai alat pengenal *host* pada *network*. Secara logika, Internet merupakan suatu *network* besar yang terdiri dari berbagai sub-*network* yang terintegrasi.Oleh karena itu, suatu IP*Address* harus bersifat unik. Tidak boleh ada satu IP*Address* yang sama dipakai oleh dua *host* yang berbeda. Contoh

IP*Address* : 66.249.89.99 (www.google.com) , 203.130.198.131 (www.e-dukasi.net) , 192.168.0.1.

Untuk itu, penggunaan IP *Address* di seluruh dunia dikoordinasi oleh lembaga sentral Internet yang di kenal dengan IANA (*Internet Assigned Numbers Authority*). Badan-badan yang mengawal IP di bawah *IANA* adalah:

1. APNIC (*Asia Pacific Network Information Centre*) kawasan Asia/ Pacific
2. ARIN (*American Registry for Internet Numbers*) kawasan Amerika Utara dan Afrika
3. LACNIC (*Regional Latin-American and Caribbean IP Address Registry*) kawasan Amerika Latin dan beberapa kepulauan Karibia
4. RIPE NCC (*Rseaux IP Europens*) kawasan Eropa, Asia tengah, and Afrika utara.

Sistem pengalamatan IP ini terbagi menjadi dua, yakni:

1) IP versi 4 (IPv4)
2) IP versi 6 (IPv6)

2.1.3.1 IP versi 4

Alamat IP versi 4 (sering disebut dengan Alamat IP*v4*) adalah sebuah jenis pengalamatan jaringan yang digunakan di dalam protokol jaringan *TCP(Transport Control Protocol)*/IP *(Internet Protocol)* yang menggunakan protokol IP versi 4. Panjang totalnya adalah 32-bit, dan secara teoritis dapat mengalamati hingga 4 miliar *host* komputer atau lebih tepatnya 4.294.967.296 *host* di seluruh dunia,

jumlah *host* tersebut didapatkan dari 256 (didapatkan dari 8 bit) dipangkat 4 (karena terdapat 4 oktet) sehingga nilai maksimal dari alamat IP versi 4 tersebut adalah 255.255.255.255 dimana nilai dihitung dari nol sehingga nilai *host* yang dapat ditampung adalah 256x256x256x256=4.294.967.296 *host*. Sehingga bila *host* yang ada diseluruh dunia melebihi kuota tersebut maka dibuatlah IP versi 6 atau IP*v6*.

2.1.3.1.1 Representasi Alamat IPv4

Alamat IP versi 4 umumnya diekspresikan dalam notasi desimal bertitik (*dotted decimal notation*), yang dibagi ke dalam empat buah oktet berukuran 8 bit. Dalam beberapa buku referensi, format bentuknya adalah **w.x.y.z**. Karena setiap oktet berukuran 8 bit, maka nilainya berkisar antara **0** hingga **255** meskipun begitu, terdapat beberapa pengecualian nilai.

Alamat IP yang dimiliki oleh sebuah *host* dapat dibagi dengan menggunakan *subnet mask* jaringan ke dalam dua buah bagian, yakni:

1) *Network IDentifier* (*NetID)* atau *Network Address* (alamat jaringan) yang digunakan khusus untuk mengidentifikasikan alamat jaringan dimana *host* berada. Dalam banyak kasus, sebuah alamat *network identifier* adalah sama dengan segmen jaringan fisik dengan batasan yang dibuat dan didefinisikan oleh routerIP. Meskipun demikian, ada beberapa kasus dimana beberapa jaringan logis terdapat di dalam sebuah segmen jaringan fisik yang sama dengan menggunakan sebuah praktek yang disebut sebagai *multinetting*. Semua sistem di dalam sebuah jaringan fisik yang

sama harus memiliki alamat *network identifier* yang sama. *Network identifier* juga harus bersifat unik dalam sebuah *Internetwork*. Jika semua *node* di dalam jaringan logis yang sama tidak dikonfigurasikan dengan menggunakan *networkidentifier* yang sama, maka terjadilah masalah yang disebut dengan *routing error*. Alamat *network identifier* tidak boleh bernilai 0 atau 255.

2) *Host Identifier*/*HostID* atau *Host Address* (alamat *host*) yang digunakan khusus untuk mengidentifikasikan alamat *host* (dapat berupa workstation, *server* atau sistem lainnya yang berbasis teknologi *TCP* (*Transport Control Protocol*)/IP(*Internet Protocol*) di dalam jaringan. Nilai *host identifier* tidak boleh bernilai 0 atau 255 dan harus bersifat unik di dalam *network identifier* atau segmen jaringan dimana ia berada.

2.1.3.1.2 Jenis-Jenis Alamat IPv4

Alamat IPv4 terbagi menjadi beberapa jenis, yakni sebagai berikut:

1) Alamat *Unicast*, merupakan alamat IPv4 yang ditentukan untuk sebuah antarmuka jaringan yang dihubungkan ke sebuah *Internetwork*. Alamat *unicast* digunakan dalam komunikasi *point-to-point* atau *one-to-one*.
2) Alamat *Broadcast*, merupakan alamat IPv4 yang didesain agar diproses oleh setiap *node* IP dalam segmen jaringan yang sama. Alamat *broadcast* digunakan dalam komunikasi *one-to-everyone*.

3) Alamat *Multicast*, merupakan alamat IPv4 yang didesain agar diproses oleh satu atau beberapa *node* dalam segmen jaringan yang sama atau berbeda. Alamat *multicast* digunakan dalam komunikasi *one-to-many*.

2.1.3.1.3 Kelas-kelas alamat IPv4

Dalam RFC 791, alamat IP versi 4 dibagi ke dalam beberapa kelas, dilihat dari oktet pertamanya, seperti terlihat pada tabel. Sebenarnya yang menjadi pembeda kelas IP versi 4 adalah pola biner yang terdapat dalam oktet pertama (utamanya adalah bit-bit awal/*high-order bit*), tapi untuk lebih mudah mengingatnya, akan lebih cepat diingat dengan menggunakan representasi desimal.

Tabel 2.1 Kelas IP*Address*

Kelas Alamat IP	Oktet Pertama (Desimal)	Oktet Pertama (Biner)	Digunakan Oleh
Kelas A	1-126	0xxx xxxx	Alamat *unicast* untuk jaringan skala besar
Kelas B	128-191	10xx xxxx	Alamat *unicast* untuk jaringan skala menengah hingga skala besar
Kelas C	192-223	110x xxxx	Alamat *unicast* untuk jaringan skala kecil
Kelas D	224-239	1110 xxxx	Alamat *multicast* (bukan alamat *unicast*)
Kelas E	240-255	1111 xxxx	Direservasikan; umumnya digunakan sebagai alamat percobaan (eksperimen); (bukan alamat *unicast*)

1. Kelas A

Kelas A hanya menggunakan *octet* pertama untuk menunjukkan ID jaringan dan menggunakan tiga *octet* yang lain untuk menunjukkan ID *host*. Bit *high order* (bit pertama dari *octet* pertama) pada kelas ini selalu diset menjadi 0 (nol). Karena, bit high-order selalu diset menjadi 0 (nol), maka tujuh bit selanjutnya menunjukkan ID jaringan. Tujuh bit ini memungkinkan adanya 127 alamat jaringan. ID jaringan 127 disediakan khusus untuk fungsi umpan balik adapter jaringan sehingga kelas A mempunyai 126 alamat yang tersedia.24 bit sisanya disediakan untuk penggunaan ID *host* dari alamat. Tersedia 16.777.214 atau (224) *host* per jaringan. Karena kelas *address* ini menyediakan banyak ID *host* perjaringan, maka penggunaan kelas A diperuntukkan bagi perusahaan yang membutuhkan penyediaan akses *host* dalam jumlah sangat besar. Nomor pada kelas A sudah tinggal sedikit karena telah digunakan untuk jaringan universitas-universitas dan militer beberapa tahun lalu.

2. Kelas B

Kelas B menggunakan *octet* pertama dan kedua untuk menentukan ID jaringan serta dua *octet* berikutnya untuk ID *host*. Bit *high order*(dua bit pertama dari *octet* pertama) dari alamat kelas ini selalu diset menjadi 10 (satu-nol). Karena bit hig-order diset menjadi 1 0, maka 14 bit sisanya menunjukkan ID jaringan. 14 bit sisanya menyediakan 16.384 alamat jaringan. 16 bit sisanya digunakan untuk menyediakan ID *host*. Kelas B

menyediakan 65.534 (216) – 2 *host* per jaringan. Kelas B disediakan untuk jaringan berskala menengah sampai besar.

3. Kelas C

Kelas C menggunakan tiga *octet* pertama untuk menentukan ID jaringan, sedangkan satu *octet* sisanya untuk ID *host*.Bit *high-order* (3 bit pertama dari oktet pertama) dari alamat kelas ini selalu diset menjadi 110 (satu-satu-nol). Karena bit high-order diset menjadi 1 1 0, maka 21 bit sisanya menunjukkan ID jaringan. 21 bit menyediakan 2.097.152 alamat jaringan. 8 bit sisanya disediakan untuk penggunaan ID *host* dari alamat. Tersedia 254 (28) – 2 *host* perjaringan. Kelas *address* diperuntukkan bagi jaringan kecil yang hanya memerlukan nomor *host* dalam jumlah terbatas.

4. Kelas D

Kelas D digunakan untuk *multicasting*.*Multicasting* digunakan untuk mengirim informasi ke nomor *host register*. *Host-host* dikelompokkan dengan meregistrasi dirinya kepada router *local* menggunakan alamat *multicast* dari*range* alamat kelas D. Bit *hig-order* untuk alamat kelas D di-set menjadi 1 1 1 0 (satu- satu- satu-nol). Bit sisanya digunakan untuk grup *host* di jaringan.

5. Kelas E

Kelas E adalah kelas eksperiment yang dipersiapkan untuk penggunaan dimasa yang akan datang. Alamat-alamat di kelas ini di identifikasikan dengan bit *high-order* nya yang diset menjadi 1 1 1 1 (satu- satu- satu- satu).

2.1.3.2 IP versi 6 (IPv6)

Alamat IP versi 6 (sering disebut sebagai alamat IPv6) adalah sebuah jenis pengalamatan jaringan yang digunakan di dalam protokol jaringan TCP/IP yang menggunakan protokol IP versi 6. Panjang totalnya adalah 128-bit, dan secara teoritis dapat mengalamati hingga 2128=3,4 x 1038 *host* komputer di seluruh dunia. Contoh alamat IP versi 6 adalah

1DA:00D3:0000:2F3B:02AA:00FF:FE28:9C5A.

Berbeda dengan IPv4 yang hanya memiliki panjang 32-bit (jumlah total alamat yang dapat dicapainya mencapai 4,294,967,296 alamat), alamat IPv6 memiliki panjang 128-bit. IPv4, meskipun total alamatnya mencapai 4 miliar, pada kenyataannya tidak sampai 4 miliar alamat, karena ada beberapa limitasi, sehingga implementasinya saat ini hanya mencapai beberapa ratus juta saja. IPv6, yang memiliki panjang 128-bit, memiliki total alamat yang mungkin hingga 2128=3,4 x 1038 alamat. Total alamat yang sangat besar ini bertujuan untuk menyediakan ruang alamat yang tidak akan habis (hingga beberapa masa ke depan), dan membentuk infrastruktur *routing* yang disusun secara hierarkis, sehingga mengurangi kompleksitas proses *routing* dan tabel *routing*. Sama seperti halnya IPv4, IPv6 juga mengizinkan adanya DHCP *server* sebagai pengatur alamat otomatis. Jika dalam IPv4 terdapat *dynamic Address* dan *static Address*, maka dalam IPv6, konfigurasi alamat dengan menggunakan DHCP *Server* dinamakan dengan *stateful Address configuration*, sementara jika konfigurasi alamat IPv6 tanpa DHCP *Server* dinamakan dengan *stateless Address configuration*.

Seperti halnya IPv4 yang menggunakan bit-bit pada tingkat tinggi (*high order bit*) sebagai alamat jaringan sementara bit-bit pada tingkat rendah (low-order bit) sebagai alamat *host*, dalam IPv6 juga terjadi hal serupa. Dalam IPv6, bit-bit pada tingkat tinggi akan digunakan sebagai tanda pengenal jenis alamat IPv6, yang disebut dengan *Format Prefix (FP)*. Dalam IPv6, tidak ada *subnet mask*, yang ada hanyalah *Format Prefix*. Pengalamatan IPv6 didefinisikan dalam RFC 2373.

2.1.3.2.1 Format Alamat

Dalam IPv6, alamat 128-bit akan dibagi ke dalam 8 blok berukuran 16-bit, yang dapat dikonversikan ke dalam bilangan heksadesimal berukuran 4-digit. Setiap blok bilangan heksadesimal tersebut akan dipisahkan dengan tanda titik dua (:). Karenanya, format notasi yang digunakan oleh IPv6 juga sering disebut dengan *colon-hexadecimal format*, berbeda dengan IPv4 yang menggunakan *dotted-decimal format*. Berikut ini adalah contoh alamat IPv6 dalam bentuk bilangan biner:

```
00100001110110100000000011010011000000000000000000
01011110011101100000010101010100000000
11111111111111100010100010011100010110 10
```

Gambar 2.1 Alamat IPv6 (*biner*)

Untuk menerjemahkannya ke dalam bentuk notasi *colon hexadecimal format*, angka-angka biner di atas harus dibagi ke dalam 8 buah blok berukuran 16-bit:

```
0010000111011010 0000000011010011 0000000000000000
0010111100111011 0000001010101010
0000000011111111 1111111000101000 1001110001011010
```

Gambar 2.2 Alamat IPv6 (8 blok)

Lalu, setiap blok berukuran 16-bit tersebut harus dikonversikan ke dalam bilangan heksadesimal dan setiap bilangan heksadesimal tersebut dipisahkan dengan menggunakan tanda titik dua. Hasil konversinya adalah sebagai berikut:

21DA:00D3:0000:2F3B:02AA:00FF:FE28:9C5A

Gambar 2.3 Hasil Konversi IPv6

Alamat di atas juga dapat disederhanakan lagi dengan membuang angka 0 pada awal setiap blok yang berukuran 16-bit di atas, dengan menyisakan satu digit terakhir. Dengan membuang angka 0, alamat di atas disederhanakan menjadi:

21DA:D3:0:2F3B:2AA:FF:FE28:9C5A

Gambar 2.4 Hasil Penyederhanaan IPv6

Konvensi pengalamatan IPv6 juga mengizinkan penyederhanaan alamat lebih jauh lagi, yakni dengan membuang banyak karakter 0, pada sebuah alamat yang banyak angka 0-nya. Jika sebuah alamat IPv6 yang direpresentasikan dalam notasi *colon-hexadecimal* format mengandung beberapa blok 16 bit dengan angka 0, maka alamat tersebut dapat disederhanakan dengan menggunakan tanda dua buah titik dua (**:**). Untuk menghindari kebingungan, penyederhanaan alamat IPv6 dengan cara ini sebaiknya hanya digunakan sekali saja di dalam satu alamat, karena kemungkinan nantinya pengguna tidak dapat menentukan berapa banyak bit 0 yang direpresentasikan oleh setiap tanda dua titik dua (**:**) yang terdapat dalam alamat tersebut. Tabel berikut mengilustrasikan cara penggunaan hal ini.

Tabel 2.2 Alamat IPv6

Alamat Asli	**Alamat Asli yang disederhanakan**	**Alamat yang telah dikompress**
FE80:0000:0000:0000:02AA:00FF:FE9A:4CA2	FE80:0:0:0:2AA:FF:FE9A:4CA2	FE80::2AA:FF:FE9A:4CA2
FF02:0000:0000:0000:0000:0000:0000:0002	FF02:0:0:0:0:0:0:2	FF02::2

Untuk menentukan berapa banyak bit bernilai 0 yang dibuang (dan digantikan dengan tanda dua titik dua) dalam sebuah alamat IPv6, dapat dilakukan dengan menghitung berapa banyak blok yang tersedia dalam alamat tersebut, yang kemudian dikurangkan dengan angka 8, dan angka tersebut dikalikan dengan 16. Sebagai contoh, alamat FF02::2 hanya mengandung dua blok alamat (blok FF02 dan blok 2). Maka, jumlah bit yang dibuang adalah (8-2) x 16 = 96 buah bit.

2.1.3.3 Address Khusus

Selain *Address* yang dipergunakan untuk pengenal *host*, ada beberapa jenis *Address* yang digunakan untuk keperluan khusus dan tidak boleh digunakan untuk pengenal *host address* tersebut adalah :

1. *Network Address*

   Address ini digunakan untuk mengenali suatu *network* pada jaringan Internet. Misalkan untuk *host* dengan IP *Address* kelas B 167.205.9.35. Tanpa memakai subnet, *network address* dari *host* ini adalah 167.205.0.0. *address* ini didapat dengan membuat seluruh bit *host* pada 2 segmen terakhir menjadi 0. Tujuannya adalah untuk menyederhanakan informasi *routing*

pada Internet. Router cukup melihat *network Address* (167.205) untuk menentukan kemana paket tersebut harus dikirimkan. Contoh untuk kelas C, *network address* untuk *IP Address* 202.152.1.250 adalah 202.152.1.0.

Analogi yang baik untuk menjelaskan fungsi *network address* ini adalah dalam pengolahan surat pada kantor pos. Petugas penyortir surat pada kantor pos cukup melihat kota tujuan pada alamat surat (tidak perlu membaca seluruh alamat) untuk menentukan jalur mana yang harus ditempuh surat tersebut. Pekerjaan *routing* surat-surat menjadi lebih cepat.Demikian juga halnya dengan router di Internet pada saat melakukan *routing* atas paket-paket data.

2. *Broadcast Address*

Address ini digunakan untuk mengirim/menerima informasi yang harus diketahui oleh seluruh *host* yang ada pada suatu *network*. Seperti diketahui, setiap paket IP memiliki header alamat tujuan berupa IP *Address* dari *host* yang akan dituju oleh paket tersebut. Dengan adanya alamat ini, maka hanya *host* tujuan saja yang memproses paket tersebut, sedangkan *host* lain akan mengabaikannya.

Tidak efisien jika ia harus membuat replikasi paket sebanyak jumlah *host* tujuan. Pemakaian *bandwidth* akan meningkat dan beban kerja *host* pengirim bertambah, padahal isi paket-paket tersebut sama. Oleh karena itu, dibuat konsep *broadcast address*. *Host* cukup mengirim ke alamat *broadcast*, maka seluruh *host* yang ada pada *network* akan menerima paket tersebut. Konsekuensinya, seluruh *host* pada *network* yang sama harus

memiliki *addressbroadcast* yang sama dan *address* tersebut tidak boleh digunakan sebagai IP *Address* untuk *host* tertentu. Jadi, sebenarnya setiap *host* memiliki 2 *Address* untuk menerima paket : pertama adalah IP *Address*-nya yang bersifat unik dan kedua adalah *broadcast Address* pada *network* tempat *host* tersebut berada. *Address broadcast* diperoleh dengan membuat seluruh bit *host* pada IP *Address* menjadi 1. Jadi, untuk *host* dengan *IPAddress* 167.205.9.35 atau 167.205.240.2, *broadcast Address*-nya adalah 167.205.255.255 (2 segmen terakhir dari IP *Address* tersebut dibuat berharga 11111111.11111111, sehingga secara desimal terbaca 255.255). Jenis informasi yang di-*broadcast* biasanya adalah informasi *routing*.

3. Netmask

Adalah *address* yang digunakan untuk melakukan masking / filter pada proses pembentukan *routing* supaya kita cukup memperhatikan beberapa bit saja dari total 32 bit IP *Address*. Artinya dengan menggunakan netmask tidak perlu kita memperhatikan seluruh (32 bit) IP *Address* untuk menentukan *routing*, akan tetapi cukup beberapa buah saja dari IP *Address* yg kita perlu perhatikan untuk menentukan kemana *packet* tersebut dikirim.

Kaitan antara *host address*, *network address*, *broadcast address* & *network mask* sangat erat sekali - semua dapat dihitung dengan mudah. Jika ingin secara serius mengoperasikan sebuah jaringan komputer menggunakan teknologi TCP/IP & Internet, adalah mutlak untuk menguasai konsep IP *Address* tersebut. Konsep IP *Address* sangat penting artinya bagi *routing* jaringan Internet. Kemampuan untuk membagi jaringan dalam subnet IP

Address penting artinya untuk memperoleh *routing* yang sangat effisien & tidak membebani router-router yang ada di Internet.

2.1.4. Model Referensi

2.1.4.1 Model OSI

Model referensi jaringan terbuka OSI atau OSI *Reference Model for open networking* adalah sebuah model arsitektural jaringan yang dikembangkan oleh badan *International Organization for Standardization* (ISO) di Eropa pada tahun 1977. OSI sendiri merupakan singkatan dari *Open System Interconnection*. Model ini disebut juga dengan model "Model tujuh lapis OSI" (*OSI seven layer model*).

Sebelum munculnya model referensi OSI, sistem jaringan komputer sangat tergantung kepada pemasok (*vendor*). OSI berupaya membentuk standar umum jaringan komputer untuk menunjang interoperatibilitas antar pemasok yang berbeda. Dalam suatu jaringan yang besar biasanya terdapat banyak protokol jaringan yang berbeda. Tidak adanya suatu protokol yang sama, membuat banyak perangkat tidak bisa saling berkomunikasi.

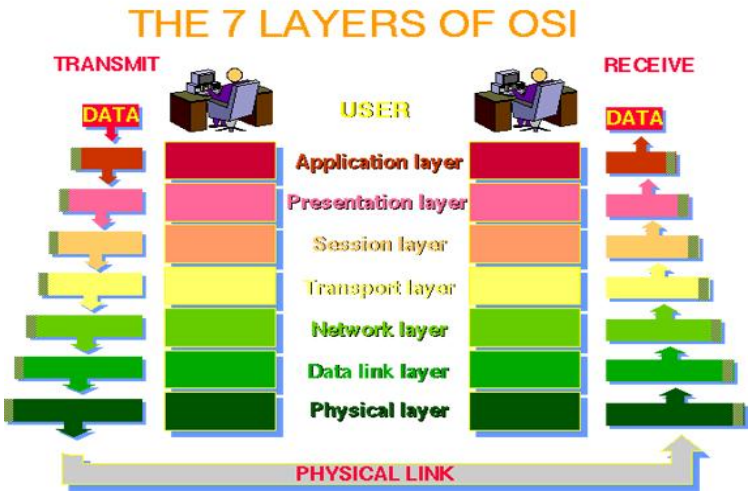
Model referensi ini pada awalnya ditujukan sebagai basis untuk mengembangkan protokol-protokol jaringan, meski pada kenyataannya inisatif ini mengalami kegagalan. Kegagalan itu disebabkan oleh beberapa faktor berikut:

- Standar model referensi ini, jika dibandingkan dengan model referensi DARPA (Model Internet) yang dikembangkan oleh IETF (*Internet Engineering Task Force*) , sangat berdekatan. Model DARPA adalah model basis protokol TCP/IP yang populer digunakan.

- Model referensi ini dianggap sangat kompleks. Beberapa fungsi (seperti halnya metode komunikasi *connectionless*) dianggap kurang bagus, sementara fungsi lainnya (seperti *flow control* dan koreksi kesalahan) diulang-ulang pada beberapa lapisan.
- Pertumbuhan Internet dan protokol TCP/IP (sebuah protokol jaringan dunia nyata) membuat OSI *Reference Model* menjadi kurang diminati.

Pemerintah Amerika Serikat mencoba untuk mendukung protokol OSI *Reference* Model dalam solusi jaringan pemerintah pada tahun 1980-an, dengan mengimplementasikan beberapa standar yang disebut dengan *Government Open Systems Interconnection Profile* (GOSIP). Meski demikian.usaha ini akhirnya ditinggalkan pada tahun 1995, dan implementasi jaringan yang menggunakan *OSI Reference model* jarang dijumpai di luar Eropa.

OSI Reference Model pun akhirnya dilihat sebagai sebuah model ideal dari koneksi logis yang harus terjadi agar komunikasi data dalam jaringan dapat berlangsung.Beberapa protokol yang digunakan dalam dunia nyata, semacam TCP/IP, DECnet dan IBM *Systems Network Architecture* (SNA) memetakan tumpukan protokol (*protokol stack*) mereka ke *OSI Reference Model*.*OSI Reference Model* pun digunakan sebagai titik awal untuk mempelajari bagaimana beberapa protokol jaringan di dalam sebuah kumpulan protokol dapat berfungsi dan berinteraksi.



Gambar 2.5 Layer OSI

Struktur tujuh lapis model OSI, bersamaan dengan *protokol data unit* pada setiap lapisan OSI Reference Model memiliki tujuh lapis, yakni sebagai berikut :

Tabel 2.3 Layer OSI

Lapisan ke-	Nama lapisan	Keterangan,
7	*Application layer*	Berfungsi sebagai antarmuka dengan aplikasi dengan fungsionalitas jaringan mengatur bagaimana aplikasi dapat mengakses jaringan,dan kemudian membuat pesan-pesan kesalahan. Protokol yang berada dalam lapisan ini adalah HTTP, FTP, SMTP, dan NFS.
6	*Presentation layer*	Berfungsi untuk mentranslasikan data yang hendak ditransmisikan oleh aplikasi ke dalam format yang dapat ditransmisikan melalui

		jaringan. Protokol yang berada dalam level ini adalah perangkat lunak redirektor (*redirector software*), seperti layanan *Workstation* (dalam Windows NT) dan juga *Network* shell (semacam *Virtual Network Computing* (VNC) atau *Remote Desktop Protocol* (RDP).
5	*Session layer*	Berfungsi untuk mendefinisikan bagaimana koneksi dapat dibuat, dipelihara, atau dihancurkan. Selain itu, di level ini juga dilakukan resolusi nama.
4	*Transport layer*	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 terhadp paket-paket yang hilang di tengah jalan.
3	*Network layer*	Berfungsi untuk mendefinisikan alamat-alamat IP, membuat *header* untuk paket-paket, dan kemudian

		melakukan *routing* melalui *Internetworking* dengan menggunakan *router* dan *switch layer-3*.
2	*Data-link layer*	Berfungsi untuk menentukan bagaimana bit-bit data dikelompokkan menjadi format yang disebut sebagai *frame*. Selain itu, pada level ini terjadi koreksi kesalahan, *flow control*, pengalamatan perangkat keras (seperti halnya *Media Access Control Address (MAC Address))*, dan menetukan bagaimana perangkat-perangkat jaringan seperti *hub*, *bridge*, *repeater*, dan *switch layer 2* beroperasi. Spesifikasi IEEE 802, membagi *level* ini menjadi dua level anak, yaitu lapisan *Logical Link Control* (LLC) dan lapisan *Media Access Control* (MAC).

1	*Physical layer*	Berfungsi untuk mendefinisikan media transmisi jaringan, metode pensinyalan, sinkronisasi bit, arsitektur jaringan (seperti halnya *Ethernet* atau Token Ring), topologi jaringan dan pengabelan. Selain itu, level ini juga mendefinisikan bagaimana *Network Interface Card* (NIC) dapat berinteraksi dengan media kabel atau radio.

2.1.4.2 Model TCP/IP

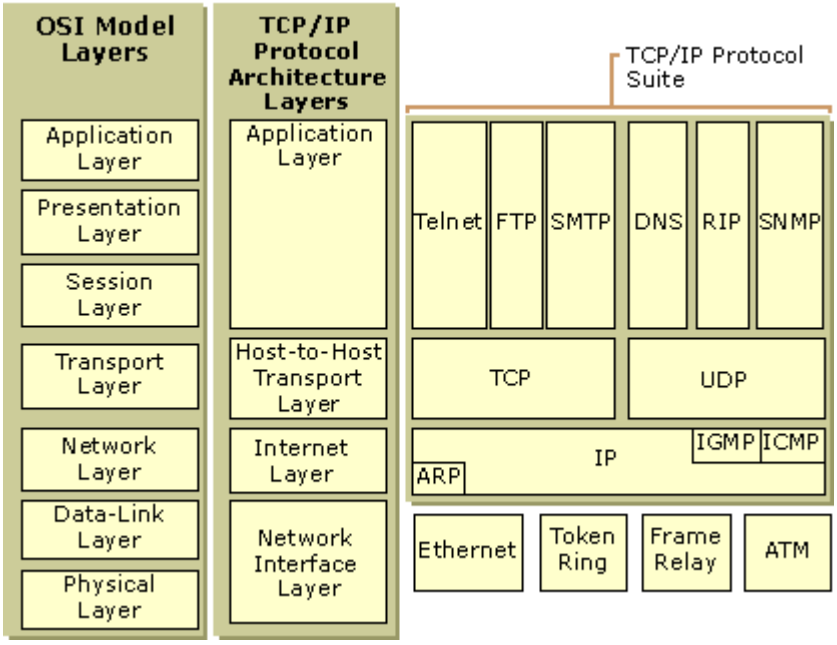
TCP/IP (singkatan dari *Transmission Control Protocol*/*Internet Protocol*) adalah standar komunikasidata yang digunakan oleh komunitas Internet dalam proses tukar-menukar data dari satu komputer ke komputer lain di dalam jaringan Internet. Model ini tidaklah dapat berdiri sendiri, karena memang model ini berupa kumpulan protokol (*protocol suite*). Protokol adalah kumpulan peraturan yang mendefinisikan bagaimana cara informasi ditransmisikan melalui jaringan. Model ini juga merupakan model 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

Model 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). TCP/IP merupakan sebuah standar jaringan terbuka yang bersifat independen terhadap mekanisme transport jaringan fisik yang digunakan, sehingga dapat digunakan dimana saja. Model 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. Model 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.

Model TCP/IP selalu berevolusi seiring dengan waktu, mengingat semakin banyaknya kebutuhan terhadap jaringan komputer dan Internet. Pengembangan ini dilakukan oleh beberapa badan, seperti halnya *Internet Society* (ISOC), *Internet Architecture Board* (IAB), dan *Internet Engineering Task Force* (IETF). Macam-macam protokol yang berjalan di atas TCP/IP, skema pengalamatan, dan konsep TCP/IP didefinisikan dalam dokumen yang disebut sebagai *Request for Comments* (RFC) yang dikeluarkan oleh IETF.

Arsitektur TCP/IP tidaklah berbasis model referensi tujuh lapis OSI, tetapi menggunakan model referensi DARPA. Seperti diperlihatkan dalam gambar, 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.



Gambar 2.6 Layer TCP/IP

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

1) 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 Protokol* (FTP), *Telnet*, *Simple Mail Transfer Protocol* (SMTP), *Simple Network Management Protokol* (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.

2) 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).
3) 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 Protokol* (IP), *Address Resolution Protocol* (ARP), *Internet Control Message Protocol* (ICMP), dan *Internet Group Management Protocol* (IGMP).
4) 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)).

2.1.5. Perangkat Keras Jaringan

2.1.5.1. PC Dekstop

PC *desktop* adalah komputer personal yang digunakan untuk menjalankan aplikasi *desktop* grafis dalam keperluan sehari-hari.Aplikasi yang dijalankan umumnya merupakan aplikasi yang membutuhkan kualoitas grafis yang cukup tinggi. PC *desktop* memiliki perangkat keras yang lebih lengkap bila dibandingkan dengan laptop ataupun tablet PC. PC *desktop* memiliki ruangan yang jauh lebih luas untuk dilengkapi dengan drive-drive seperti CD-rom CD-*rewriter*. Namun, PC dekstop memiliki kekurangan, yaitu tidak bersifat portabel sehingga kurang praktis untuk dibawa bepergian.

PC *desktop* membutuhkan system operasi grafis karena aplikasi yang dijalankan umumnya menuntut antarmuka grafis. Aplikasi Internet saat ini pun umumnya membutuhkan antarmuka grafis yang cukup tinggi. Selain itu, PC *desktop* saat ini juga dituntut untuk mampu menjalankan aplikasi multimedia sehingga memberikan kenyamanan yang optimal dalam penggunaannya.

PC *desktop* umumnya dilengkapi dengan harddisk tersendiri dimana sistem operasi dan data tersimpan.Namun, ada juga PC *desktop* yang tidak membutuhkan hardisk atau sering disebut dengan jaringan *diskless workstation*. *Diskless workstation* membutukan jaringan dengan sistem operasi Unix atau Windows NT.

2.1.5.2. Network Inteface Card (NIC)

Network Inteface Card (NIC) atau LAN Card adalah sebuah kartu yang terpasang pada sebuah komputer sehingga komputer tersebut mampu terkoneksi pada jaringan.NIC menyediakan antarmuka ke media. NIC bias menggunakan *transceiver eksternal* atau melalui *transceiver internal* terintegrasi yang dipasang pada *Network Interface Card* PCB. NIC biasanya juga dilengkapi protokol *control firmware* dan *Ethernet Controller* yang dibutuhkan untuk mendukung protokol data-link *Medium Access Control* (MAC) yang digunakan *Ethernet*. Setiap NIC mendapatkan *Ethernet* source *address* dari perusahaan pembuat NIC (biasanya disimpan di ROM pada NIC). *Address* tersebut bersifat unik secara global. Ada sekitar 16 blok (atau 8 juta) *address* yang disediakan oleh perusahaan pembuat NIC berdasarkan struktur *address*ing *flat*. Hal ini untuk memastikan bahwa tidak ada dua *Ethernet Network Interface* (NIC) yang akan mempunyai source *Address* yang sama.

Pada komputer personal dan workstation dalam suatu *Local Area Network* (LAN) biasanya terpasang *Network Interface Card* yang di desain khusus untuk transmisi data pada LAN.Teknologi pada LAN *card* yang umumnya digunakan adalah *Ethernet* atau Token Ring. Kebanyakan LAN card yang ada dipasaran menggunakan teknologi *Ethernet* sehingga banyak orang yang menyebut *Network Inteface Card* (NIC) sebagai *Ethernet Card*. Teknologi *Ethernet* dikembangkan oleh ternama Xerox.

Standard IEEE 802.3 menentukan *Ethernet* berdasarkan fisik dan lapisan data link pada model OSI. Sebagian besar system *Ethernet* menggunakan :

1. *Carrier-sense multiple-access with collision detection* (CSMA/CD) untuk mengontrol akses ke media jaringan.
2. *Broadcast base band*
3. Metode pemaketan data menjadi paket data yang disebut frame.
4. Kecepatan transmisinya 10 Mbps, 100 Mbps dan 1 Gbps.

Tipe-tipe *Ethernet* :

1. **10Base5,** tipe kartu *Ethernet* menggunakan kabel koaksial dengan diameter 0,5 inci dan berwarna kuning. Topologi yang digunakan menggunakan Topologi Bus. Jangkauan jarak maksimum 500 meter. Apabila ditambahkan repeater (penguat) akan dapat mencapai jarak ±2,5 km. Kecepatan transmisi data 10 Mbps.
2. **10Base**2, tipe kartu *Ethernet* menggunakan kabel koaksial dengan diameter 5 mm. Topologi yang digunakan berbentuk bus. Jangkauan jarak maksimum 185 m dan kecepatan transmisi data 10 Mbps.
3. **10BaseT**, tipe kartu *Ethernet* menggunakan kabel twisted pair. Topologi yang digunakan adalah Star. Jangkauan jarak maksimum 100 m dan kecepatan transmisi data 100 Mbps.
4. **10BaseF**, tipe *Ethernet* yang menggunakan kabel serat optik (*fiber optic*). Topologi yang digunakan adalah star. Jangkauan jarak sampai dengan 2000 m. Untuk transmisi output dan input menggunakan kabel yang berbeda. Kecepatan transmisi data mencapai 100 Mbps.

5. **100BaseT series**, tipe *Ethernet* menggunakan kabel twisted pair. Tipe ini memiliki beragam metode akses. Kecepatan transmisi data mencapai 20-200 Mbps.

2.1.5.3. Hub

Hub adalah alat ynag berfungsi sebagai tempat untuk menerima data dan meneruskannya menuju tempat yang lain. Hub terdiri dari port-port RJ-45 *female* sehingga kabel-kabel twisted pair yang sudah terpasang konektor RJ-45 pada ujung-ujungnya dapat dengan mudah ditancapkan pada hub. Hub digunakan pada topologi star dan dapat diparalel hingga maksimum tiga buah hub. Hub dipasaran beredar dengan jumlah 5 port, 10 port, 12 port, 16 port, 24 port, 32 port dan yang terbesar 64 port. Untuk jaringanyang menggunakan hub yang diparallel lebih dari tiga buah, diperlukan sebuah router untuk menghubungkan hub ynag keempat.

2.1.5.4. Switch

Switch atau switch hub pada dasarnya adalah hub juga. Switch memiliki kemampuan untuk mentransfer data hanya pada por yang dituju saja, berbeda dengan hub yang mem-*broadcast* data yang dating ke semua *port* yang dimiliki termasuk *port* tempat data tersebut berasal. Perbedaan ini membuat switch lebih cepat dalam mentransferkan data walaupun memiliki kecepatan yang sama dengan hub. Dengan sistem teknologi *Ethernet* dalam sebuah LAN, switch akan membedakan alamat komputer tempat data tersebut berasal dan alamat komputer (MAC *Address* atau *Media Access Control Address*) tempat data tersebut dituju.

Pada jaringan yang luas atau Internet, switch akan membedakan IP asal dan IP tujuan ke data tersebut.

Pada suatu LAN yang menggunakan model OSI, switch melakukan fungsi komunikasi layer 2 atau *data-link layer*. Namun, pada jaringan luas atau Internet, switch membutuhkan sebuah router untuk dapat meneruskan paket datanya. Beberapa switch model terbaru mampu melakukan komunikasi layer 3 atau melakukan fungsi *routing* sehingga disebut IP switch. Kehadiran switch menggeser keberadaan hub sehingga di pasaran saat ini lebih banyak ditemui switch karena dengan selisih harga yang sedikit, menawarkan kinerja yang lebih baik dibandingkan hub.

2.1.5.5. Router

Router adalah sebuah alat jaringan komputer yang mengirimkan paket data melalui sebuah jaringan atau Internet menuju tujuannya, melalui sebuah proses yang dikenal sebagai *routing*. Proses *routing* terjadi pada lapisan 3 (Lapisan jaringan seperti Internet Protokol) dari stack protokol tujuh-lapis OSI. Router berfungsi sebagai penghubung antar dua atau lebih jaringan untuk meneruskan data dari satu jaringan ke jaringan lainnya. Router berbeda dengan switch. Switch merupakan penghubung beberapa alat untuk membentuk suatu *Local Area Network (LAN).*

Sebagai ilustrasi perbedaan fungsi dari router dan switch adalah switch merupakan suatu jalanan, dan router merupakan penghubung antar jalan. Masing-masing rumah berada pada jalan yang memiliki alamat dalam suatu urutan tertentu. Dengan cara yang sama, switch menghubungkan berbagai

macam alat, dimana masing-masing alat memiliki alamat IP sendiri pada sebuah LAN.

Router sangat banyak digunakan dalam jaringan berbasis teknologi protokol TCP/IP, dan router jenis itu disebut juga dengan IP Router. Selain IP Router, ada lagi AppleTalk Router, dan masih ada beberapa jenis router lainnya. Internet merupakan contoh utama dari sebuah jaringan yang memiliki banyak router IP.Router dapat digunakan untuk menghubungkan banyak jaringan kecil ke sebuah jaringan yang lebih besar, yang disebut dengan Internet-*work*, atau untuk membagi sebuah jaringan besar ke dalam beberapa sub-*network* untuk meningkatkan kinerja dan juga mempermudah manajemennya. Router juga kadang digunakan untuk mengoneksikan dua buah jaringan yang menggunakan media yang berbeda atau berbeda arsitektur jaringan, seperti halnya dari *Ethernet* ke *Token Ring*.

Router juga dapat digunakan untuk menghubungkan *LAN (Local Area Network)* ke sebuah layanan telekomunikasi seperti halnya telekomunikasi *leased line* atau *DSL* (*Digital Subscriber Line)*. Router yang digunakan untuk menghubungkan *LAN (Local Area Network)* ke sebuah koneksi *leased line* seperti T1, atau T3, sering disebut sebagai *access server*. Sementara itu, router yang digunakan untuk menghubungkan jaringan lokal ke sebuah koneksi *DSL (Digital Subscriber Line)* disebut juga dengan *DSL* (*Digital Subscriber Line)* router. Router-router jenis tersebut umumnya memiliki fungsi *firewall* untuk melakukan penapisan paket berdasarkan alamat sumber dan alamat tujuan paket tersebut, meski beberapa router tidak memilikinya. Router yang memiliki fitur

penapisan paket disebut juga dengan *packet filtering* router. Router umumnya memblokir lalu lintas data yang dipancarkan secara *broadcast* sehingga dapat mencegah adanya *broadcaststorm* yang mampu memperlambat kinerja jaringan.

2.1.5.6. Bridge

Pengertian dari sebuah bridge adalah bekerja pada *data link layer* pada OSI. bridge adalah alat yang digunakan pada suatu jaringan yang berfungsi untuk memisahkan sebuah jaringan yang luas menjadi segment yang lebih kecil. bridge membaca alamat MAC (*media access control*) dari setiap paket data yang diterima yang kemudian akan mempelajari *bridging table* untuk memutuskan apa yang akan dikerjakan bridge selanjutnya pada paket data tersebut, apakah diteruskan atau diabaikan. jika switch menpunyai *domain collision* sendiri-sendiri di setiap portnya, begitu juga dengan bridge memiliki *domain collision* tetapi ia juga dapat membaginya dari sebuah *domain collision* yang besar menjadi yang lebih kecil, dan bridge hanya akan melewatkan paket data antar segment - segment jika hanya segment itu sangat diperlukan terdapat tiga jenis bridge jaringan yang umum dijumpai :

1. Bridge Lokal : sebuah bridge yang dapat menghubungkan segmen-segmen jaringan lokal.
2. Bridge Remote : dapat digunakan untuk membuat sebuah sambungan (link) antara *LAN (Local Area Network)* untuk membuat sebuah *Wide Area Network*.
3. Bridge Nirkabel : sebuah bridge yang dapat menggabungkan jaringan *LAN (Local Area Network)* berkabel dan jaringan *LAN nirkabel*

2.1.5.7. Repeater

Repeater adalah sebuah *device* yang meregenerasi/menghasilkan kembali sinyal yang ditransmisikan pada kabel.Repeater mengijinkan sinyal untuk mengalir diluar batas keterbatasan panjang kabel. Sebuah repeater tidak melakukan translasi atau filterisasi paket. Repeater mempunyai beberapa karakteristik antara lain :

1. Digunakan untuk meregenerasi sinyal *baseband* yang ada
2. Digunakan terutama pada topologi *bus koaksial* (linear) segmen yang dihubungkan oleh sebuah repeater
3. Harus menggunakan metode *media access control* (MAC) yang sama (misalnya, sebuah repeater tidak bisa melewatkan *traffic* antara *Ethernet* dan token ring) Repeater bisa melewatkan *traffic* antara beberapa media yang berbeda (misalnya, coax ke *fiber optic*)
4. Jika *Interface* yang sesuai tersedia repeater tidak melakukan akselerasi atau mengubah sinyal, ia hanya meregenerasinya saja.
5. Repeater tidak melalukan *filter packet* atau batasan kongesti/kemacetan.
6. Repeater akan melewatkan paket *broadcast*
7. Repeater beroperasi di *physical layer* pada model OSI fungsi dasar dari repeater (regenerasi sinyal) bisa dilakukan oleh *device-device* penghubung (misalnya hub atau bridge).

Karena repeater tidak melakukan apa-apa terhadap filterisasi atau pembatasan *traffic*, ia seharusnya dilihat terutama sebagai penghubung workstasion yang jauh, tidak sebagai penambah workstasion tambahan. Dengan

kata lain, repeater seharusnya digunakan untuk menambahkan jarak dari *network*, bukan kepadatannya.

2.1.5.8. Access Point

Digunakan untuk melakukan pengaturan lalu lintas jaringan dari *mobile* radio ke jaringan kabel atau dari *backbone* jaringan *wireless client*/*server*.Biasanya berbentuk kotak kecil dengan 1 atau 2 antena kecil. Peralatan ini merupakan radio based, berupa *receiver* dan transmiter yang akan terkoneksi dengan *LAN (Local Area Network)* kabel atau *broadband Ethernet*. Saat ini beredar di pasaran adalah access point yang telah dilengkapi dengan router di dalamnya yang biasa disebut *wireless* router.

2.1.5.9. Modem

Modem berasal dari singkatan *Modulator Demodulator*. *Modulator* merupakan bagian yang mengubah sinyal informasi kedalam sinyal pembawa (*carrier*) dan siap untuk dikirimkan, sedangkan *Demodulator* adalah bagian yang memisahkan sinyal informasi (yang berisi data atau pesan) dari sinyal pembawa yang diterima sehingga informasi tersebut dapat diterima dengan baik. Modem merupakan penggabungan kedua-duanya, artinya modem adalah alat komunikasi dua arah. Setiap perangkat komunikasi jarak jauh dua-arah umumnya menggunakan bagian yang disebut "modem", seperti VSAT, Microwave Radio, dan lain sebagainya, namun umumnya istilah modem lebih dikenal sebagai Perangkat keras yang sering digunakan untuk komunikasi pada komputer. Data dari komputer yang berbentuk sinyal digital diberikan kepada modem untuk diubah menjadi sinyal analog. Sinyal analog tersebut dapat dikirimkan melalui

beberapa media telekomunikasi seperti telepon dan radio. Setibanya di modem tujuan, sinyal analog tersebut diubah menjadi sinyal digital kembali dan dikirimkan kepada komputer. Terdapat dua jenis modem secara fisiknya, yaitu modem eksternal dan modem internal. Terdapat beberapa jenis modem yaitu :

1. Modem 3GP
2. Modem GSM
3. Modem analog yaitu modem yang mengubah sinyal analog menjadi sinyal digital
4. Modem ADSL (*Asymetric Digital Subscribe Line*) yaitu modem yang memungkinkan berselancar Internet dan menggunakan telepon analog secara berbarengan. Caranya sangat mudah, untuk ADSL diberikan sebuah alat yang disebut sebagai Splitter atau pembagi line. Posisi Splitter ditempatkan di depan ketika line telepon masuk. Artinya anda tidak boleh mencabangkan line modem untuk ADSL dengan suara secara langsung. Alat Splitter berguna untuk menghilangkan gangguan ketika anda sedang menggunakan ADSL modem. Dengan Splitter keduanya dapat berjalanbersamaan, sehingga pengguna dapat menjawab dan menelpon seseorang dengan telepon biasa. Di sisi lain, pengguna tetap dapat terkoneksi dengan Internet melalui ADSL modem.
5. Modem kabel yaitu modem yang menerima data langsung dari penyedia layanan lewat TV Kabel
6. Modem CDMA

2.1.6. Media Transmisi

Media transmisi adalah media yang dapat digunakan untuk mengirimkan informasi dari suatu tempat ke tempat lain. Dalam jaringan, semua media yang dapat menyalurkan gelombang listrik atau elektromagnetik atau cahaya dapat dipakal sebagai media pengirim, baik untuk pengiriman dan penerimaan data.Pilihan media transmisi (pengirim) untuk keperluan komunikasi data tergantung pada beberapa faktor, seperti harga, *Performance* jaringan yang dikehendaki, ada atau ada tidaknya medium tersebut. Beberapa media transmisi dalam jaringan komputer yaitu :

2.1.6.1. Copper Media

Copper media merupakan semua media transmisi data yang terbuat dari bahan tembaga. Orang biasanya menyebut dengan nama kabel. Data yang dikirim melalui kabel, bentuknya adalah sinyal listrik (tegangan atau arus) digital.

Jenis-jenis kabel yang dipakai sebagai transmisi data pada jaringan :

1. Koaksial
2. STP
3. UTP

2.1.6.1.1. Kabel Koaksial

Kabel ini sering digunakan sebagai kabel antena TV. Disebut juga sebagai kabel BNC (*Bayonet Naur Connector*). Kabel ini merupakan kabel yang paling banyak digunakan pada LAN *(Local Area Network),* karena memiliki perlindungan terhadap derau yang lebih tinggi, murah, dan mampu mengirimkan data dengan kecepatan standar . Ada 2 jenis yaitu RG-58 (10Base2) dan RG-8 (10Base5). Ada 3 jenis konektor pada kabel Coaxial, yaitu T konektor, I konektor (socket) dan BNC konektor.Keuntungan menggunakan kabel koaksial adalah lebih murah dari pada kabel fiber optic dan jarak jangkauannya cukup jauh dari kabel jenis UTP/STP yang menggunakan repeater sebagai penguatnya. Kekurangannya adalah susah pada saat instalasi, baik installasi konektor maupun kabel. Untuk saat ini kabel koaksial sudah tidak direkomendasikan lagi intuk instalasi jaringan.

2.1.6.1.2. Twisted Pair

Twisted Pair terdiri dari 2 jenis :

1. *Unshielded Twisted Pair* (UTP)
2. *Shielded Twisted Pair* (STP)

Kabel ini terdiri dari 4 pasang kabel yang dipilin (*twisted pair*), instalasinya mudah, harganya relatif murah dan cukup handal.

1. Shielded Twisted Pair

   Keuntungan menggunakan kabel STP adalah lebih tahan terhadap interferensi gelombang elektromagnetik baik dari dari dalam maupun

dari luar. Kekurangannya adalah mahal, susah pada saat instalasi (terutama masalah grounding), dan jarak jangkauannya hanya 100m

2. Unshielded Twisted Pair

Keuntungan menggunakan kabel UTP adalah murah dan mudah diinstalasi.Kekurangannya adalah rentan terhadap interferensi gelombang elektromagnetik, dan jarak jangkauannya hanya 100m. Ada beberapa kategori untuk kabel *Twisted Pair*, yaitu :

1. Kategori 1 (Cat-1).

   Umumnya menggunakan konduktor padat standar AWG sebanyak 22 atau 24 pin dengan range impedansi yang lebar. Digunakan pada koneksi telepon dan tidak direkomendasikan untuk transmisi data.

2. Kategori 2 (Cat-2).

   Range impedansi yang lebar, sering digunakan pada sistem PBX dan sistem alarm. Transmisi data ISDN menggunakan kabel kategori 2, dengan *bandwidth* maksimum 1 MBps.

3. Kategori 3 (Cat-3).

   Sering disebut kabel voice grade, menggunakan konduktor padat sebanyak 22 atau 24 pin dengan impedansi 100 Ω dan berfungsi hingga 16 MBps. Dapat digunakan untuk jaringan 10BaseT dan Token Ring dengan *bandwidth* 4 Mbps.

4. Kategori 4 (Cat-4).

   Seperti kategori 3 dengan *bandwidth* 20 MBps, diterapkan pada jaringan Token Ring dengan *bandwidth* 16 Mbps.

5. Kategori 5 (Cat-5).

   Merupakan kabel Twisted Pair terbaik (data *grade*) dengan *bandwidth* 100 Mbps dan jangkauan transmisi maksimum 100m.

2.1.6.2. Fiber optik

Pada fiber optik terdapat tiga jenis kabel yang biasanya digunakan, yaitu single mode, multi mode dan *plastic optical* fiber yang berfungsi sebagai petunjuk cahaya dari ujung kabel ke ujung kabel lainnya.Dari transmitter receiver, yang mengubah pulsa elektronik ke cahaya dan sebaliknya, dalam bentuk *light-emitting diode* ataupun laser. Kabel fiber optic single mode merupakan fiber glass tunggal dengan diameter 8.3 sampai 10 mikrometer, memiliki satu jenis transmisi yang dapat mengantarkan data berkapasitas besar dengan kecepatan tinggi untuk jarak jauh, dan membutuhkan sumber cahaya dengan lebar spektrum yang lebih kecil.

Kemampuan kabel jenis single mode dalam mengantarkan transmisi adalah 50 kali lebih cepat dari kabel jenis multimode, karena memiliki core yang lebih kecil sehingga dapat menghilangsetiap distorsi dan pulsa cahaya yang tumpang tindih. Kabel fiber optic multimode terbuat dari *fiberglass* dengan diameter lebih besar, yaitu 50 sampai dengan 100 mikrometer yang dapat mengantarkan data berkapasitas besar dengan kecepatan tinggi untuk jarak

menengah. Apabila jarak yang ditempuh lebih dari 3000 kaki, akan terjadi distorsi sinyal pada sisi penerima yang mengakibatkan transmisi data menjadi tidak akurat. Sedang *plastic optical fiber* adalah kabel berbasis plastik terbaru yang menjamin tingkat performa yang sama dengan *fiber glass* dalam jarak pendek dengan biaya yang jauh lebih murah. Saat ini, *fiber optic* telah digunakan sebagai standar kabel data dalam biding *physical layer* telekomunikasi atau jaringan, seperti perangkat TV kabel, juga sistem keamanan yang menggunakan *Closed Circuit Television* (CCTV), dan lain sebagainya Bahan dasar dari optical media adalah kaca dengan ukuran yang sangat kecil (skala mikron). Biasanya dikenal dengan nama *fibre optic* (serat optik). Data yang dilewatkan pada medium ini dalam bentuk cahaya (laser atau inframerah).

Satu buah kabel *fibre optic* terdiri atas dua fiber,satu berfungsi untuk *Transmit* (Tx) dan satunya untuk *Receive* (Rx) sehingga komunikasi dengan *fiber optik* bisa terjadi dua arah secara bersama-sama (*full duplex*).

2.1.6.3. Wireless Network

Saat ini sudah banyak digunakan jaringan tanpa kabel (*wireless network*), transmisi data menggunakan sinar infra merah atau gelombang mikro untuk menghantarkan data. Walaupun kedengarannya praktis, namun kendala yang dihadapi disini adalah masalah jarak, *bandwidth*, dan mahalnya biaya. Namun demikian untuk kebutuhan *LAN (Local Area Network)* di dalam gedung, saat ini sudah dikembangkan teknologi *wireless* untuk *Active Hub* (*Wireless Access Point*) dan *Wireless LAN Card*, sehingga bisa mengurangi kompleksitas kabel transmisi

data pada jaringan komputer. *Wireless Access Point* juga bisa digabungkan (up-*link*) dengan Active Hub dari jaringan yang sudah ada.

Media transmisi wireless menggunakan gelombang radio frekuensi tinggi. Biasanya gelombang elektromagnetik dengan frekuensi 2.4 Ghz dan 5 Ghz. Data-data digital yang dikirim melalui *wireless* ini akan dimodulasikan ke dalam gelombang elektromagnetik ini.

2.2 VoIP

Voice over Internet Protokol (VoIP) dedefinisikan sebagai suatu sistem yang menggunakan jaringan Internet untuk mengirimkan paket data suara dari suatu tempat ke tempat lainnya menggunakan perantara protokol IP. VoIP mentransmisikan sinyal suara dengan mengubahnya ke dalam bentuk digital, dan dikelompokkan menjadi paket–paket data yang dikirim dengan menggunakan *platform* IP (*Internet Protocol*).

Ada berapa komponen yang dibutuhkan dalam membuat jaringan VoIP, yaitu:

1. User Agent

   User Agent adalah komponen VoIP yang langsung berhubungan dengan user. User dapat menggunakan dan mengoperasikan untuk melakukan komunikasi dengan user yang lain dan juga melakukan setting untuk berkomunkasi dengan phone analog.

2. Proxy

   Merupakan software yang digunakan sebagai *server* VoIP yang menangani proses registrasi dan *autentikasi user*. Proxy melakukan banyak proses yang membuat panggilandapat terlaksana diantara user dan menampilkan proses panggilanyang terlihat dalam log di *server*.

3. Protokol *Signaling*

   Signaling dilakukan sebelum melakukan panggilan.Jalur dibentuk *point to point* menggunakan protokol signaling yang berbasis TCP. Setelah jalur terbentuk, RTP melakukan transfer sinyal suara yang telah dipaketisasi melui jalur UDP.

4. CODEC (Coder-Decoder)

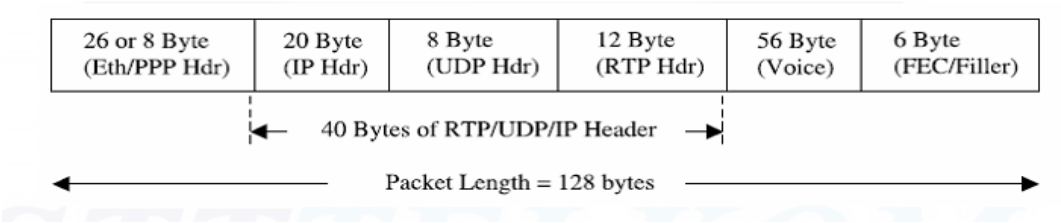
   Digunakan untuk mengubah informasi sinyal suara analog menjadi sinyal digital yang dapat ditransmisikan melalui jaringan IP dengan bandwidth tertentu dan mendapatkan informasinya kembali.Masing masing codec punyai spesifikasi *bandwidth* yang dibutuhkan yang berbeda.Misalkan G711 64 Kbps.

2.2.1. Paket VoIP

Tiap paket VoIP terdiri atas dua bagian, yakni header dan payload (beban). Header terdiri atas :

1. IP *header*,
2. *Real-time Transport Protokol (RTP) header*,
3. *User Datagram Protokol (UDP) header*,

4. *Ethernet header*



Gambar 2.7 Contoh Protokol Stack VoIP

1. IP *header*

IP *header* bertugas menyimpan informasi *routing* untuk mengirimkan paket-paket ke tujuan. Pada setiap *header IP* disertakan tipe layanan atau *Type of Service (ToS)* yang memungkinkan paket tertentu seperti paket suara diperlakukan berbeda dengan paket yang non real-time.

2. RTP (*Real*-time *Transport Protokol*) *header*

RTP *header* adalah header yang dapat dimanfaatkan untuk melakukan framing dan segmentasi data real time. RTP tidak mendukung realibilitas paket untuk sampai tujuan. RTP menggunakan protokol kendali yang disebut RTCP (*Real Time Control Protocol*) yang mengendalikan QoS dan sinkroniasi media *stream* yang berbeda

3. UDP (*User Datagram Protokol*) *header*

UDP *header* memiliki ciri tertentu yaitu tidak menjamin paket akan mencapai tujuan sehingga UDP cocok digunakan pada aplikasi *voice real time* yang sangat peka terhadap *delay* dan *latency*.

4. *Ethernet header*

Sebuah *Ethernet header* selalu berisi informasi tentang siapa yang mengirimkan frame dan kemana frame itu pergi. *Header* juga mungkin mengandung informasi lain, seperti berapa banyak *byte* yang terdapat dalam frame tersebut. Informasi ini disebut sebagai *field length*, atau *field* panjang dan digunakan sebagai pengecekan *error*. Jika stasion penerima mengukur frame dengan hasil yang berbeda dibandingkan dengan ukuran yang ditunjukkan dalam *field length*. maka stasion penerima akan meminta stasion pengirim untuk mengirimkan sebuah frame baru. Jika *field length* tidak digunakan, maka *header* tersebut mungkin mengandung sebuah *field type*, yang menerangkan apa tipe *frame Ethernet* tersebut.

2.2.2. Kualitas VoIP

VoIP merupakan salah satu jenis layanan realtime yang membutuhkan QoS (*Quality of Service*). Untuk menjamin terjaganya kualitas VoIP maka ada beberapa faktor yang mempengaruhi diantaranya :

2.2.2.1 Codec

Sebuah codec (yang merupakan kepanjangan dari *compressor*/*decompressor* atau *coder*/*decoder* adalah suatu *hardware* atau *software* yang melakukan sampling terhadap sinyal suara analog, kemudian

mengkonversi ke dalam bit – bit digital dan mengeluarkannya. Beberapa jenis codec melakukan kompresi agar dapat menghemat *bandwidth*.

ITU-T (*International Telecommunication Union – Telecommunication Sector*) membuat beberapa standar untuk *voice coding* yang direkomendasikan untuk implementasi VoIP. Beberapa standar yang sering dikenal antara lain:

1. G.711

   Sebelum mengetahui lebih jauh apa itu G.711 sebelumnya diberikan sedikit gambaran singkat fungsi dari kompresi. Sebuah kanal video yang baik tanpa dikompresi akan mengambil bandwidth sekitar 9Mbps. Sebuah kanal suara (*audio*) yang baik tanpa di kompresi akan mengambil *bandwidth* sekitar 64Kbps. Dengan adanya teknik kompresi, kita dapat menghemat sebuah kanal video menjadi sekitar 30Kbps dan kanal suara menjadi 6Kbps (*half-duplex*), artinya sebuah saluran Internet yang tidak terlalu cepat sebetulnya dapat digunakan untuk menyalurkan video dan audio sekaligus. Tentunya untuk kebutuhkan konferensi dua arah dibutuhkan *double bandwidth*, artinya minimal sekali kita harus menggunakan kanal 64Kbps ke Internet. Dengan begitu suara (*audio*) akan memakan *bandwidth* jauh lebih sedikit dibanding pengiriman gambar atau video.

   G.711 adalah suatu standar Internasional untuk kompresi audio dengan menggunakan teknik PCM (*Pulse Code Modulation*) dalam pengiriman suara. Standar ini banyak digunakan oleh operator

Telekomunikasi termasuk PT. Telkom sebagai penyedia jaringan telepon terbesar di Indonesia.

PCM (*Pulse Code Modulation*) mengkonversikan sinyal analog ke bentuk digital dengan melakukan *sampling* sinyal analog tersebut 8000 kali/detik dan dikodekan dalam kode angka.Jarak antar sampel adalah 125 µ detik.Sinyal analog pada suatu percakapan diasumsikan berfrekuensi 300 Hz – 3400 Hz. Sinyal tersampel lalu dikonversikan ke bentuk diskrit.Sinyal diskrit ini direpresentasikan dengan kode yang disesuaikan dengan amplitudo dari sinyal sampel. Format PCM menggunakan 8 bit untuk pengkodeannya. Laju transmisi diperoleh dengan mengkalikan 8000 sampel /detik dengan 8 bit/sampel, menghasilkan 64.000 bit/detik . Bit rate 64 kbps ini merupakan standar transmisi untuk satu kanal telepon digital.

Percakapan berupa sinyal analog yang melalui jaringan PSTN mengalami kompresi dan pengkodean menjadi sinyal digital oleh PCM G.711 sebelum memasuki VoIP *gateway* . Pada VoIP *gateway*, dibagian terminal, terdapat audio codec melakukan proses *framing* (pembentukan frame datagram IP yang dikompresi) dari sinyal suara terdigitasi (hasil PCM G.711) dan juga melakukan rekonstruksi pada sisi *receiver*. Frame-frame yang merupakan paket-paket informasi ini lalu di transmisikan melalui jaringan IP dengan suatu standar komunikasi jaringan *packet–based*. Standar G.711 merupakan teknik kompresi yang tidak efisien, karena akan memakan *bandwidth* 64Kbps untuk kanal pembicaraan. Agar

bandwidth yang digunakan tidak besar dan tidak mengesampingkan kualitas suara, maka solusi yang digunakan untuk pengkompresi diguanakan standar G.723.1.

2. G.723.1

Pengkode sinyal suara G.723.1 adalah jenis pengkode suara yang direkomendasikan untuk terminal multimedia dengan *bit rate* rendah. G.723.1 memiliki *dual rate speech coder* yang dapat di-*switch* pada batas 5.3 kbit/s dan 6.3 kbit/s. Dengan memiliki *dual rate speech coder* ini maka G.723.1 memiliki fleksibilitas dalam beradaptasi terhadap informasi yang dikandung oleh sinyal suara. G.723.1 dilengkapi dengan fasilitas untuk memperbagus sinyal suara hasil sintesis.

Pada bagian *encoder* G.723.1 dilengkapi dengan *formant perceptual weighting filter* dan *harmonic noise shaping filter* sementara di bagian decoder-nya G.723.1 memiliki *pitch post filter* dan *formant postfilter* sehingga sinyal suara hasil rekonstruksi menjadi sangat mirip dengan aslinya. Sinyal eksitasi untuk bit *rate* rendah dikodekan dengan *Algebraic Code Excited Linier Prediction* (*ACELP*) sedangkan untuk rate tinggi dikodekan dengan menggunakan *Multipulse Maximum Likelihood Quantization* (*MP-MLQ*). *Rate* yang lebih tinggi menghasilkan kualitas yang lebih baik. Masukan bagi G.723.1 adalah sinyal suara digital yang di-sampling dengan frekuensi *sampling* 8.000 Hz dan dikuantisasi dengan

PCM 16 bit. *Delay* algoritmik dari G.723.1 adalah 37.5 msec (panjang *frame* ditambah *lookahead*).

Delay pemrosesannya sangat ditentukan oleh prosesor yang mengerjakan perhitungan-perhitungan pada algoritma G.723.1. Dengan menggunakan DSP prosesor maka *delay* pemrosesan dapat diperkecil.

Selain itu kompresi data suara yang direkomendasikan ITU adalah G.726, merupakan teknik pengkodean suara ADPCM dengan hasil pengkodean pada 40, 32, 24, dan 16 kbps. Biasanya juga digunakan pada pengiriman paket data pada telepon publik maupun peralatan PBX yang mendukung ADPCM.G.728, merupakan teknik pengkodean suara CELP dengan hasil pengkodean 16 kbps.G.729 merupakan pengkodean suara jenis CELP dengan hasil kompresi pada 8 kbps.

Berikut ini adalah gambar perbandingan beberapa teknik kompresi standar ITU-T.

Teknik Kompresi	Bit Rate (Kbps)	Sample size (ms)	MOS
G.711 PCM	64	0,125	4,1
G.726 ADPCM	32	0,125	3,85
G.728 LD-CELP	16	0,625	3,61
G.729 CS-ACELP	8	10	3,92
G.723.1 MP-MLQ	6,3	30	3,9
G.723.1 ACELP	5,3	30	3,65

Gambar 2.8 Teknik Kompresi Standar ITU-T

2.2.2.2 Frame Loss

Dalam jaringan Internet, setiap paket yang dikirim ke penerima belum tentu sampai, hal ini disebabkan karena sifat jaringan Internet yaitu *best*

effort, dimana jaringan hanya berusaha menjaga agar paket sampai di penerima. Jika paket lama berputar di jaringan, oleh router paket itu akan dibuang. Begitu pula dengan QoS. Jaringan Internet tidak menyediakan QoS bagi penggunanya. Hal itu yang menyebabkan VoIP menggunakan protokol RTP. Salah satu kelemahan protokol ini adalah tidak adanya transmisi ulang jika paket yang dikirim hilang.

2.2.2.3 Delay

Dalam perancangan jaringn VoIP, delay merupakan suatu permasalahan yang harus diperhitungkan karena kualitas suara bagus tidaknya tergantung dari waktu *delay*. Besarnya *delay* maksimum yang direkomendasikan oleh ITU untuk aplikasi suara adalah 150 ms, sedangkan delay maksimum dengan kualitas suara yang masih dapat diterima pengguna adalah 250 ms. *Delay end to end* adalah jumlah *delay* konversi suara analog–digital, *delay* waktu paketisasi atau bisa disebut juga *delay* panjang paket dan *delay* jaringan pada saat t (waktu) beberapa *delay* yang dapat mengganggu kualitas suara dalam perancangan jaringan VoIP dapat dikelompokkan menjadi :

1. *Propagation delay* (*delay* yang terjadi akibat transmisi melalui jarak antar pengirim dan penerima)
2. *Serialization delay* (*delay* pada saat proses peletakan bit ke dalam sirkuit)
3. *Processing delay* (*delay* yang terjadi saat proses *coding*, *compression*, *decompression* dan *decoding*)

4. *Packetization delay* (*delay* yang terjadi saat proses paketisasi *digital voice sample*)
5. *Queuing delay* (*delay* akibat waktu tunggu paket sampai dilayani)
6. *Jitter buffer* (*delay* akibat adanya *buffer* untuk mengatasi *jitter*)

Selain itu parameter – parameter lain yang mempengaruhi adalah QoS *(Quality of Service)*, agar didapatkan hasil suara sama dengan menggunakan telepon tradisional PSTN (*Public Switch Telephony Network*). Beberapa parameter yang mempengaruhi QoS antara lain :

1. Pemenuhan kebutuhan *bandwidth*
2. Keterlambatan data (*latency*)
3. *Packet loss* dan *desequencing*
4. Jenis kompresi data
5. Interopabilitas peralatan (vendor yang berbeda)
6. Jenis standar multimedia yang digunakan (H.323/SIP/MGCP)

Untuk berkomunikasi dengan menggunakan teknologi VoIP yang harus *real time* adalah *jitter*, *echo* dan *loss pa*cket.

Jitter merupakan variasi *delay* yang terjadi akibat adanya selisih waktu atau interval antar kedatangan paket di penerima. Untuk mengatasi *jitter* maka paket data yang datang dikumpulkan dulu dalam *jitter buffer* selama waktu yang telah ditentukan sampai paket dapat diterima pada sisi penerima dengan urutan yang benar.

Echo disebabkan perbedaan impedansi dari jaringan yang menggunakan *four-wire* dengan *two-wire*. Efek *echo* adalah suatu efek yang dialami

mendengar suara sendiri ketika sedang melakukan percakapan. Mendengar suara sendiri pada waktu lebih dari 25 ms dapat menyebabkan terhentinya pembicaraan.

Loss packet (kehilangan paket) ketika terjadi *peak load* dan *congestion* (kemacetan transmisi paket akibat padatnya *traffic* yang harus dilayani) dalam batas waktu tertentu, maka *frame* (gabungan data *payload* dan *header* yang di transmisikan) suara akan dibuang sebagaimana perlakuan terhadap frame data lainnya pada jaringan berbasis IP.

Salah satu alternatif solusi permasalahan di atas adalah membangun *link* antar *node* pada jaringan VoIP dengan spesifikasi dan dimensi dengan QoS yang baik dan dapat mengantisipasi perubahan lonjakan trafik hingga pada suatu batas tertentu.

2.2.2.4 Bandwith

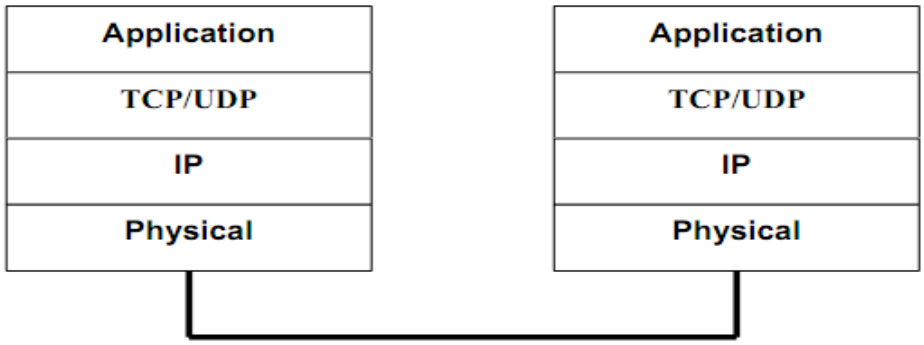
Bandwidth adalah kecepatan maksimum yang dapat digunakan untuk melakukan transmisi data antar komputer pada jaringan IP atau Internet. Dalam perancangan VoIP, *bandwidth* merupakan suatu yang harus diperhitungkan agar dapat memenuhi kebutuhan user yang dapat digunakan menjadi parameter untuk menghitung jumlah peralatan yang dibutuhkan dalam suatu jaringan. Perhitungan ini juga sangat diperlukan dalam efisiensi jaringan dan biaya serta sebagai acuan pemenuhan kebutuhan untuk pengembangan di masa mendatang. *Packet lo*ss (kehilangan paket data pada proses transmisi) dan *desequencing* merupakan masalah yang berhubungan

dengan kebutuhan *bandwidth*, namun lebih dipengaruhi oleh stabilitas *rute* yang dilewati data pada jaringan, metode antrian yang efisien, pengaturan pada router, dan penggunaan kontrol terhadap kongesti (kelebihan beban data) pada jaringan. *Packet loss* terjadi ketika terdapat penumpukan data pada jalur yang dilewati dan menyebabkan terjadinya *overflow buffer* pada router.

2.2.3. Protokol-Protokol Penunjang VoIP

2.2.3.1 Protokol TCP/IP

TCP/IP (*Transfer Control Protocol/Internet Protocol*) merupakan sebuah protokol yang digunakan pada jaringan Internet. Protokol ini terdiri dari dua bagian besar, yaitu TCP dan IP. Ilustrasi pemrosesan data untuk dikirimkan dengan menggunakan protokol TCP/IP diberikan pada gambar 2.8.



Gambar 2.9 Ilustrasi Pemrosesan pada TCP/IP

1. *Application layer*

   Fungsi utama lapisan ini adalah pemindahan file. Perpindahan file dari sebuah sistem ke sistem lainnya yang berbeda memerlukan suatu sistem pengendalian untuk menangatasi adanya ketidakkompatibelan

sistem file yang berbeda – beda. Protokol ini berhubungan dengan aplikasi. Salah satu contoh aplikasi yang telah dikenal misalnya HTTP (*Hypertext Transfer Protokol*) untuk web, FTP (*File Transfer Protokol*) untuk perpindahan file, dan TELNET untuk terminal maya jarak jauh.

2. TCP (*Transmission Control Protocol*)

Dalam mentransmisikan data pada layer *transport* ada dua protokol yang berperan yaitu TCP dan UDP. TCP merupakan protokol yang *connection-oriented* yang artinya menjaga reliabilitas hubungan komunikadasi *end-to-end*. Konsep dasar cara kerja TCP adalah mengirm dan menerima segment – segment informasi dengan panjang data bervariasi pada suatu datagram Internet. TCP menjamin realibilitas hubungan komunikasi karena melakukan perbaikan terhadap data yang rusak, hilang atau kesalahan kirim. Hal ini dilakukan dengan memberikan nomor urut pada setiap oktet yang dikirimkan dan membutuhkan sinyal jawaban positif dari penerima berupa sinyal ACK (*acknoledgment*). Jika sinyal ACK ini tidak diterima pada interval pada waktu tertentu, maka data akan dikirimkan kembali. Pada sisi penerima, nomor urut tadi berguna untuk mencegah kesalahan urutan data dan duplikasi data. TCP juga memiliki mekanisme *flow control* dengan cara mencantumkan informasi dalam sinyal ACK mengenai batas jumlah oktet data yang masih boleh ditransmisikan pada setiap segment yang diterima dengan sukses.

Dalam hubungan VoIP, TCP digunakan pada saat *signaling*, TCP digunakan untuk menjamin setup suatu *call* pada sesi signaling. TCP tidak digunakan dalam pengiriman data suara pada VoIP karena pada suatu komunikasi data VoIP penanganan data yang mengalami keterlambatan lebih penting dari pada penanganan paket yang hilang.

3. *User Datagram Protocol* (UDP)

UDP yang merupakan salah satu protokol utama di atas IP merupakan *transport* protokol yang lebih sederhana dibandingkan dengan TCP. UDP digunakan untuk situasi yang tidak mementingkan mekanisme reliabilitas. Header UDP hanya berisi empat *field* yaitu *source port*, *destination port*, *length* dan UDP *checksum* dimana fungsinya hampir sama dengan TCP, namun fasilitas *checksum* pada UDP bersifat opsional.

UDP pada VoIP digunakan untuk mengirimkan audio *stream* yang dikrimkan secara terus menerus. UDP digunakan pada VoIP karena pada pengiriman *audio streaming* yang berlangsung terus menerus lebih mementingkan kecepatan pengiriman data agar tiba di tujuan tanpa memperhatikan adanya paket yang hilang walaupun mencapai 50% dari jumlah paket yang dikirimkan. Karena UDP mampu mengirimkan data *streaming* dengan cepat, maka dalam teknologi VoIP UDP merupakan salah satu protokol penting yang digunakan sebagai *header* pada pengiriman data selain RTP dan IP.Untuk mengurangi

jumlah paket yang hilang saat pengiriman data (karena tidak terdapat mekanisme pengiriman ulang) maka pada teknolgi VoIP pengiriman data banyak dilakukan pada *private network*.

4. Internet Protokol (IP)

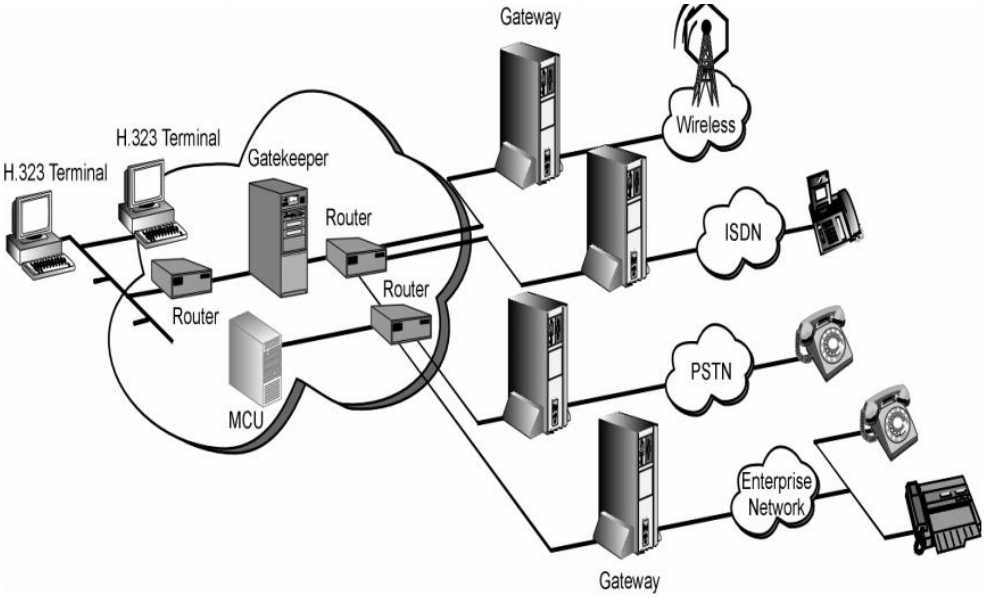
Internet Protokol didesain untuk interkoneksi sistem komunikasi komputer pada jaringan paket *switched*. Pada jaringan TCP/IP, sebuah komputer diidentifikasi dengan alamat IP. Tiap-tiap komputer memiliki alamat IP yang unik, masing-masing berbeda satu sama lainnya. Hal ini dilakukan untuk mencegah kesalahan pada transfer data. Terakhir, protokol data akses berhubungan langsung dengan media fisik. Secara umum protokol ini bertugas untuk menangani pendeteksian kesalahan pada saat transfer data. Untuk komunikasi datanya, Internet Protokol mengimplementasikan dua fungsi dasar yaitu *addressing* dan fragmentasi.

Salah satu hal penting dalam IP dalam pengiriman informasi adalah metode pengalamatan pengirim dan penerima.Saat ini terdapat standar pengalamatan yang sudah digunakan yaitu IPv4 dengan alamat terdiri dari 32 bit. Jumlah alamat yang diciptakan dengan IPv4 diperkirakan tidak dapat mencukupi kebutuhan pengalamatan IP sehingga dalam beberapa tahun mendatang akan diimplementasikan sistim pengalamatan yang baru yaitu IPv6 yang menggunakan sistem pengalamatan 128 bit.

2.2.3.2 Protokol VoIP

2.2.3.2.1 H.323

H.323 merupakan protokol standar yang direkomendasikan oleh ITU-T yang mendefinisikan komunikasi multimedia *real-time* dan konferensi melalui jaringan *packet-based* yang tidak menyediakan *guaranteed* QoS seperti LAN (*Local Area Network)* dan Internet.



Gambar 2.10 Alur Kerja Protokol H.323

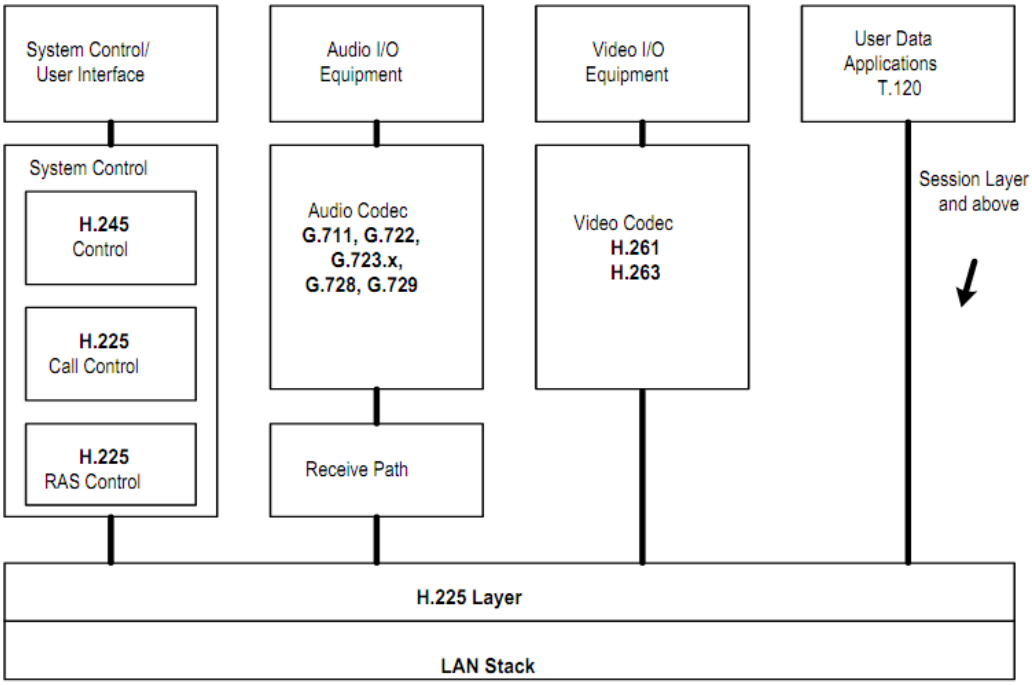
H.323 merupakan protokol yang memayungi beberapa protokol lain yang terlibat dalam proses transmisi multimedia. Komponen dari protokol H.323, yaitu :

1. Terminal

   Terminal adalah klien *endpoint* dalam suatu LAN *(Local Area Network)* yang menyediakan komunikasi dua arah dan *real-time*. H.323 menspesifikasikan mode operasi yang dibutuhkan untuk audio, video, dan/atau terminal data untuk dapat bekerja sama-sama.

2. Gateway**.**

Komponen gateway menghubungkan jaringan H.323 dengan jaringan berbeda. Fungsi dasar entitas ini menyambungkan terminal H.323 dengan terminal non H.323. Gateway adalah komponen H.323 yang bertugas melakukan translasi yang tepat antara format transmisi (cotohnya H.225.0 ke/dari H.221) dan antara prosedur komunikasi (contohnya H.245 ke/dari H.242).



Gambar 2.11 Arsitektur Protokol H.323

Berikut adalah penjelasan mengenai arsitektur protokol H.323.

1. G.7xx codecs (*Audio Codec*)

Rekomendasi - rekomendasi ini mendefinisikan mengenai *coding* dan *decoding* sinyal suara analog ke format digital beserta dengan format kompresinya. Contohnya : G.711, G.729, G.722, dan lain-lain.

2. H.26x codecs (Video Codec)

   Rekomendasi mengenai proses digitalisasi sinyal video analog. Contohnya : H.261 dan H.263

3. H.225.0

   Jika *gatekeeper* terdapat dalam suatu *network* maka H.225.0 mengatur proses registrasi terminal ke *gatekeeper* tersebut dan mengatur pula proses admisi di jaringan tersebut. Jika *gatekeeper* tidak ada maka H.225 digunakan untuk proses *setup* dan *clear down* panggi*lan*, bekerja sama dengan protokol Q.931.

4. H.245

   Protokol ini berfungsi untuk membangun kanal logikal (*logical channel*) yang akan menjadi kanal transmisi media. Setelah proses setup hubungan antara dua endpoint berhasil dilakukan menggunakan H.225.0 dan Q.931

5. Q.931

   Q.931 digunakan bersama H.225.0 untuk membangun hubungan H.323.H.225.0 di sisipkan dalam pesan UUIE (*User to User Information Element*) dari Q.931 untuk menyediakan informasi tambahan yang tidak tersedia dalam format Q.931 misalnya informasi mengenai IP *Address*.

6. RTP (*Real Time Transport Protocol*)

   RTP merupakan protokol yang digunakan untuk mendapatkan transmisi multimedia (suara dan video) secara real time.Pada saat ditransmisikan melalui jaringan IP, RTP menempati layer bawah UDP.

7. RTCP (*RTP Control Protocol*)

   RTCP mirip dengan RTP. Protokol ini mendefinisikan mekanisme pengawasan terhadap kualitas penerimaan media yang ditransmisikan menggunakan RTP dalam suatu sesi *real time*.

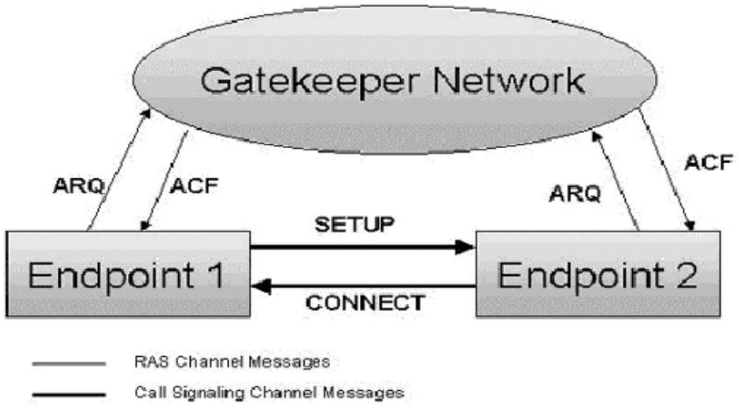
8. T.120

   Protokol untuk mengatur pertukaran data pada saat terjadi panggilan multimedia.

3. Gatekeeper**.**

   Gatekeeper merupakan komponen yang paling penting dalam sistem H.323.Entitas ini merupakan komponen opsional. Gatekeeper menyediakan layanan *call control,* bekerja sama dengan terminal, MCU, *Gateway* atau MC. Komponen ini juga dapat melakukan fungsi opsional seperti *Call Control Signalling, Call Authorization, Bandwidth Management* dan *Call Management*

   .



Gambar 2.12 Cara Kerja Gatekeeper

4. *Multipoint Control Unit* (MCU)

Entitas MCU adalah sebuah *endpoint* pada LAN *(Local Area Network)*, mendukung konferensi *point-to-point* dan multipoint.Terdiri dari *Multipoint Controller* atau MC dan opsional *Multipoint Processor* atau MP. *Multipoint Controller* bertanggung jawab atas determinasi kapabilitas umum untuk *audio* dan *video processing* antara semua terminal, menyediakan fungsi kontrol untuk *endpoint* dalam sebuah konferensi *multipoint* dan membawa pargantian kapabilitas, serta mengatur mode operasi umum untuk transmisi *stream* multimedia antara *endpoint*. Tujuan desain dan pengembangan H.323 adalah untuk memungkinkan interoperabilitas dengan tipe terminal multimedia lainnya. MC tidak berurusan langsung dengan beberapa *stream* media. Hal ini ditangani oleh MP, bagian yang mencampur, memindahkan, dan memproses audio, video dan/atau bit-bit data.

2.2.3.2.2 SIP (Session Initiation Protokol)

Session Initiation Protocol atau SIP merupakan standar IETF untuk suara atau layanan multimedia melalui jaringan Internet. SIP (RFC 2543) diajukan pada tahun 1999. Pencipta standar ini adalah Henning Schulzrinne. SIP merupakan protokol layer aplikasi yang digunakan untuk manajemen pengaturan panggilandan pemutusan panggilan SIP digunakan bersamaan dengan protokol IETF lain seperti SAP,SDP,MGCP (MEGACO) untk menyediakan layanan VoIP yang lebih luas. Arsitektur SIP mirip dengan arsitektur HTTP. Arsitekturnya terdiri dari *request* yang dikirim dari user SIP ke *server* SIP. *Server* itu memproses *request* yang masuk dan memberikan respon kepada *client*. Permintaan *request* itu, bersama dengan komponen respon pesan yang lain membuat suatu komunikasi SIP.

1. Komponen SIP

   Arsitektur SIP terdiri dari dua buah komponen seperti tertera di bawah ini.

   1. User Agent

      SIP *User Agent* merupakan akhir dari sistem (terminal akhir) yang bertindak berdasarkan kehendak dari pemakai. Terdiri dari dua bagian yaitu:

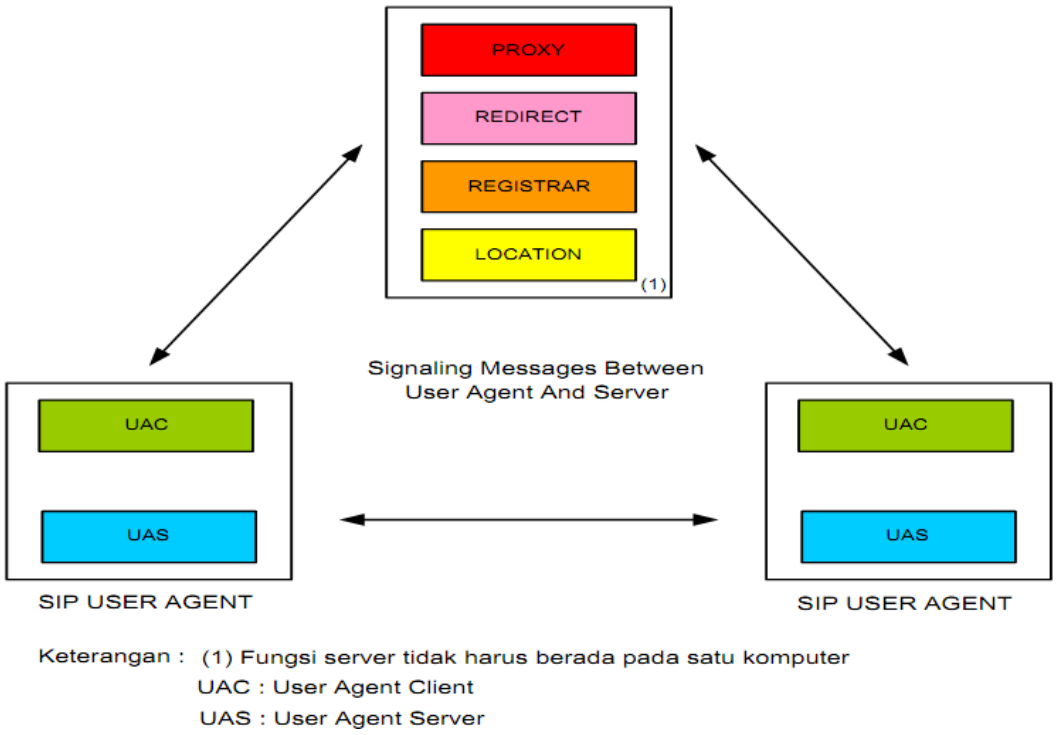
      - *User Agent Client* (UAC) : bagian ini terdapat pada pemakai (*Client*), yang digunakan untuk melakukan inisiasi request dari *server* SIP ke UAS

- *User Agent Server* (UAS) : bagian ini berfungsi untuk Mendengar dan merespon terhadap request SIP.

2. SIP *Server*

Arsitektur SIP sendiri menjelaskan jenis – jenis *server* pada jaringan untuk membantu layanan dan pengaturan panggilan SIP.

1. *Registration Server* : *server* ini menerima *request* registrasi dari user SIP dan melakukan update terhadap lokasi user dengan *server* ini.
2. *Proxy Server* : *Server* ini menerima *request* SIP dan meneruskan ke *server* yang dituju yang memiliki informasi tentang user yang dipanggil.
3. *Redirect Server* : *Server* ini setelah menerima *request* SIP, menentukan *server* yang dituju selanjutnya dan mengembalikan alamat *server* yang dituju selanjutnya kepada *client* daripada meneruskan *request* ke *server* yang dituju tersebut.



Gambar 2.13 Alur SIP *server*

SIP menggabungkan beberapa macam protokol baik dari standar yang dikeluarkan oleh IETF maupun ITU – T.

1. IETF *Session Description Protokol (*SDP*)* yang mendefinisikan suatu metode standar dalam menggambarkan karakteristik dari suatu sesi multimedia.
2. IETF *Session Announcement Protokol (*SAP*)* yang berjalan setiap periode waktu tertentu untuk mengumumkan parameter dari suatu sesi konferensi.
3. IETF *Realtime Transport Protokol (*RTP*)* dan *Realtime Transport Control Protokol (*RTCP*)* yang menyediakan informasi tentang manajemen *transport* dan *session*

4. Algoritma pengkodean. Yang direkomendasikan ITU – T seperti G.723.1, G.729,G728.

Messages yang terdapat pada SIP didefinisikan dalam dua format :

1. *Request*, dikirim dari *client* ke *server*, yang berisi tentang operasi yang diminta oleh *client* tersebut.
2. Responses, dikirim dari *server* ke *client*, yang berisi informasi mengenai status dari apa yang diminta oleh *client*.

Ada enam tipe dari *request messages* :

1. *INVITE* : menunjukan bahwa pengguna atau servis sedang diundang untuk bergabung dalam *session*. Isi dari pesan ini akan memasukan suatu uraian menyangkut *session* untuk pengguna yang diundang.
2. *ACK* : mengkonfirmasi bahwa *client* telah menerima suatu final *response* untuk suatu *INVITE request*, dan hanya digunakan di *INVITE request*.
3. *OPTION* : digunakan untuk *query* suatu *server* tentang kemampuan yang dimilikinya.
4. *BYE* : dikirim oleh *user agent client* untuk menunjukan pada *server* bahwa percakapan ingin segera diakhiri.
5. *CANCEL* : digunakan untuk membatalkan suatu request yang sedang menunggu keputusan.
6. *REGISTER* : digunakan oleh *client* untuk mendaftarkan informasi kontak.

Protokol SIP didukung oleh beberapa protokol, antara lain :

1. *Session Description Protocol (*SDP*)*
2. *Session Announcement Protocol (*SAP*)*
3. *Real-Time Transport Protocol (*RTP*)*
4. *Real-Time Control Protocol* (RTCP)
5. *ITU-T Codec*

2.2.4 Jenis – Jenis Layanan VoIP

Jenis – jenis layanan VoIP meliputi

1. ATA

ATA adalah metode paling umum untuk menggunakan layanan VoIP yaitu menggunakan alat yang bernama ATA yang memungkinkan kita menyambungkan telepon konvensional ke PC *(Personal Computer)* atau Internet untuk melakukan VoIP

2. IP PHONES

IP PHONES yaitu telepon yang sudah memiliki port RJ-45 untuk langsung di sambungkan ke router guna melakukan panggilan VoIP.

3. PC to PC

PC to PC yaitu panggilan VOIP yang dilakukan menggunakan PC dengan perlengkapan *microphone*, *speaker*, dan *software* yang di sediakan para *developer* komunikasi VOIP ini contoh : Skype dan InterVoIP. *User* tidak membayar satu sen pun dalam melakukan panggilan antar pengguna sesama layanan.

2.3. Sistem Terdistribusi

Sistem terdistribusi adalah sekumpulan komputer otonom yang terhubung ke suatu jaringan, dimana bagi pengguna sistem terlihat sebagai satu komputer. Maksud komputer otonomi adalah walaupun komputer tidak terhubung ke jaringan, komputer tersebut tetap data berjalan.

Dengan menjalankan sistem terdistribusi, komputer dapat melakukan :

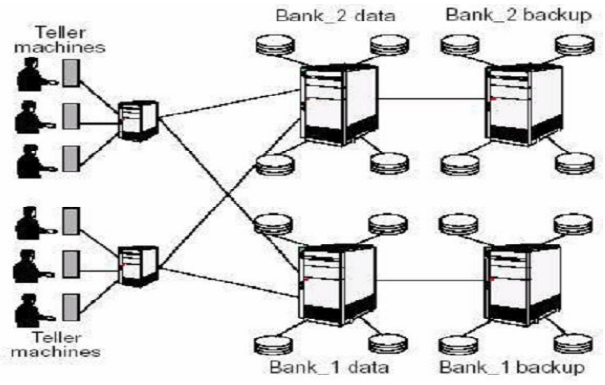
1. Koordinasi Aktifitas
2. Berbagi sumber daya : *hardware*, *software* dan data

Dengan definisi tersebut maka Internet sesungguhnya bukanlah suatu sistem terdistribusi, melainkan infrastruktur dimana sistem terdistribusi dapat di aplikasikan pada jaringan tersebut.

2.3.1 Contoh Sistem Terdistribusi

Contoh aplikatif dari system terdistribusi antara lain :

1. Sistem Telepon : ISDN, PSTN.
2. Manajemen Jaringan : Adminstrasi sesumber jaringan.
3. *Automatic Banking* (*teller machine*) *System.*



Gambar 2.14 Contoh Sistem Terdistribusi

4. *Network File System* (NFS) : Arsitektur untuk mengakses sistem file melalui jaringan.
5. WWW : Arsitektur *client*/*server* yang diterapkan infrastruktur *Internet Shared Resource* (melalui URL).

2.3.2 Keuntungan dan Permasalahan Sistem Terditribusi

2.3.2.1 Keuntungan Sistem Terdistribusi

Keuntungan yang didapatkan dalam menerapkan sistem terdistribusi, antara lain :

1. *Performance*

   Kumpulandari beberapa prosesor akan memberikan kinerja yang lebih baik dari pada komputer yang terpusat. Begitu juga kalau dilihat dari sisi biaya.
2. *Distribution*
3. *Reliability* (*Fault Tolerance*)

   Apabila salah satu komponen terjadi kerusakan, sistem tetap dapat berjalan
4. *Incremental Growth*

   Mudah dalam melakukan penambahan komputer/komponen
5. *Sharing Data/Resources*

   Berbagi data adalah salah satu hal yang pokok pada kebanyakan aplikasi.

2.3.2.2 Permasalahan dalam Sistem Terdistribusi

Kelemahan pada sistem terdistribusi adalah :

1. Kesulitan dalam membangun perangkat lunak

   Kesulitan yang akan dihadapi antara lain :bahasa pemrogramman yang harus dipakai, sistem operasi, penjadwalan dll.

2. Masalah Jaringan

   Karena sistem terdistribusi di implementasikan dalam jaringan komputer, maka isu yang berkaitan dengan jaringan komputer akan menjadi pertimbangan utama dalam merancang dan mengimplementasikan sistem.

3. Masalah Keamanan

   Karena pada sistem terdistribusi berbagi data/sumber daya merupakan hal yang mutlak maka muncul masalah yang berkaitan dengan keamanan data.

2.3.3 Karakteristik Sistem Terdistribusi

Ada beberapa hal yang harus d*IP*erhatikan dalam membangun sistem terdistribusi, yaitu :

1. *Transparency* (Kejelasan)
2. *Communication* (Komunikasi)
3. *Performance* & *Scalability* (Kinerja dan Ruang Lingkup)
4. *Heterogenity* (Keanekaragaman)
5. *Opennes* (Keterbukaan)
6. *Reliability* & *Fault Tolerancy* (Kehandalandan Toleransi Kegagalan)
7. *Security* (Kemanan)

2.3.3.1 Transparency

1. *Access transparency*

   Sumber daya lokal dan remote di akses dengan menggunakan operasi yang sama.

2. *Location transparency*

   Pengguna sistem tidak tahu mengetahui keberadaan hardware dan software (CPU, File dan data).

3. *Migration (Mobility) transparency*

   Sumber daya (baik berupa *hardware* dan/atau *software*) dapat bebas berpindah tanpa mengubah sistem penamaan.

4. *Replication transparency*

   Sistem bebas untuk menambah file atau sumber daya tanpa diketahui oleh *user* (dalam rangka meningkatkan kinerja)

5. *Concurency transparency*

   User tidak akan mengetahui keberadaan user lain dalam sistem, walaupun user tersebut menggunakan sumber daya yang sama.

6. *Failure transparency*

   Aplikasi harus dapat menyelesaikan prosesnya walaupun terdapat kegagalanpada beberapa pada komponen sistem.

7. *Performance transparency*

   Beban kerja yang bervariasi tidak akan menyebabkan turunnya kinerja sistem, hal ini dapat dicapai dengan melakukan automatisasi konfigurasi terhadap perubahan beban.

2.3.3.2 Communication

Komponen pada sistem terdistribusi harus melakukan komunikasi dalam suatu urutan. Sebagai berikut :

1. Infrastruktur jaringan (interkoneksi dan *software* jaringan)
2. Metode komunikasi : *Send*, *Receive*, *Remote Procedure Call.*
3. Model Komunikasi :
   - *Client – server communication* : Pertukaran pesan antara dua proses dimana satu proses (*client*) menggunakan / meminta layanan pada *server* dan *server* menyediakan hasil dari proses tersebut
   - *Group mulitcast* : Target dari pesan yang dikirimkan adalah gabungan dari proses, yang berasal dari suatu grup.

2.3.3.3 Performance and Scalability

2.3.3.3.1 Performance

Ada beberapa faktor yang mempengaruhi kinerja (*performance*) dari pada sistem terdistribusi :

1. Kinerja dari pada *personal workstations*.
2. Kecepatan infrastruktur komunikasi
3. Fleksibilitas dalam membagi beban kerja : contoh, apabila terdapat *prosesor* (*workstation*) yang idle maka dapat di alokasikan secara otomatis untuk mengerjakan tugas-tugas *user*.

2.3.3.3.2 Scalability

Sistem tetap harus memperhatikan efesiensi walaupun terdapat penambahan secara signifikan user atau sumber daya yang terhubung :

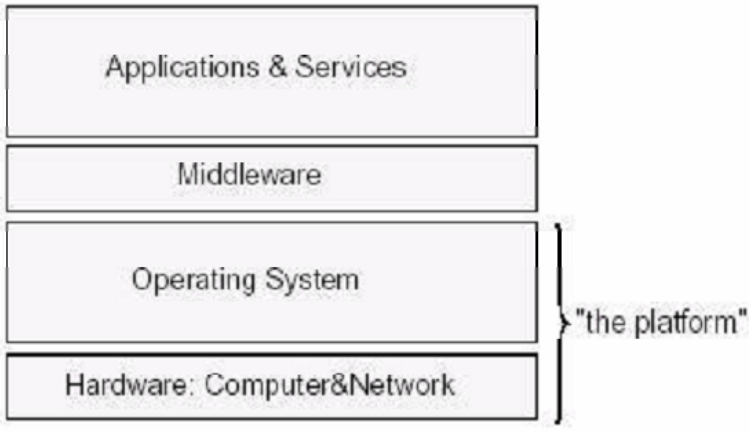
1. *Cost* (biaya) penambahan sumber daya (*resources*) harus reasonable.
2. Penurunan kinerja (*performance*) diakibatkan oleh penambahan user atau sumber daya harus terkontrol.

2.3.3.4 Heterogenity

Aplikasi yang terdistribusi biasa berjalandalam keberagaman :

1. *Hardware* : mainframes, workstations, PC's, *server*.
2. *Software* : UNIX, MS Windows, IMB OS/2, LINUX.
3. *Devices* : *teller machine*, robot, sistem *manufacturing*.
4. *Network* dan *Protokol* : *Ethernet*, FDDI, ATM, TCP/*IP*.

Melihat keanekaragaman di atas maka salah satu solusi yang bisa diterapkan adalah *middleware* yang berfungsi sebagai jembatan untuk komunikasi dan proses.



Gambar 2.15 Arsitektur *Software* pada Sistem Terdistribusi

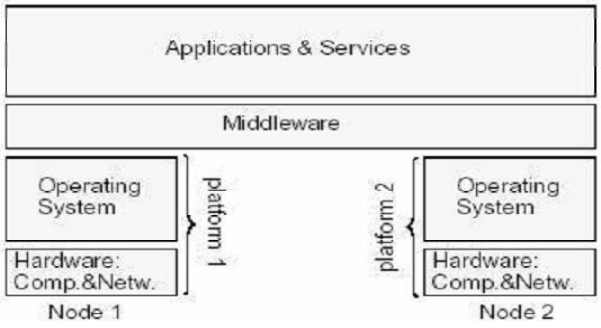
2.3.3.5 Opennes

Salah satu hal terpenting yang harus dimiliki oleh sistem terdistribusi adalah *opennes* (keterbukaan) dan *flexibility* (fleksibilitas) :

1. Setiap layanan (*services*) harus dapat di akses oleh semua user.
2. Mudah dalam implementasi, install dan *debug services*.
3. User dapat membuat dan menginstall service yang telah dibuat oleh si user tersebut.

Aspek kunci pada *opennes* :

1. *Interface* dan Protokol yang standard (seperti protokol komunikasi di Internet)
2. *Support* terhadap keanekaragaman. (dengan membuat *midleware* seperti CORBA)



Gambar 2.16 Sistem Terdistribusi pada Dua Titik

2.3.3.6 Reliability dan Fault Tolerance

Salah satu tujuan dalam membangun sistem terdistribusi adalah memungkinkan untuk melakukan improvisasi terhadap kehandalan*)* sistem.

Availability : kalau mesin mati (*down*), sistem tetap harus berjalandengan jumlah layananan yang tersisa.

1. Dalam sistem terdistribusi componen yang sangat vital (*critical resources*) berjumlah seminimal mungkin. *Critical resources* adalah komponen yang harus ada untuk menjalankan sistem terdistribusi.
2. Masing - masing *software* dan *hardware* harus direplikasi kalau terjadi kegagalan/*error* maka yang lain akan menangani.

Data dalam sistem tidak boleh hilang, *copy* dari file tersebut disimpan pada secara redundan pada *server* lain, tapi tetap harus dijaga konsistensi datanya.

Fault Tolerance sistem harus bisa mendeteksi kegagalan dan melakukan tindakan dengan dasar sebagai berikut :

1. *Mask the fault* (menutupi kegagalan) : tugas harus dapat dilanjutkan dengan menurunkan kinerja tapi tanpa terjadi kehilangan data atau informasi.
2. *Fail Gracefully* : membuat suatu antisipasi terhadap suatu kegagalan ke suatu prosedur yang telah di rencanakan dan memungkinkan untukmenghentikan proses dalam waktu yang singkat tanpa menghilangkan informasi atau data.

2.3.3.7 Security

Jenis keamanan pada sistem tersistribusi meliputi :

1. *Confidentiality*, keamanan terhadap data yang di akses oleh user yang tidak di perbolehkan (*unauthorizes user*).
2. *Integrity*, keamanan terhadap kelengkapan dan autentikasi data.

3. *Availability*, Menjaga agar resource dapat selalu di akses.

   Sistem terdistribusi harus memperbolehkan komunikasi antara program pada *computer* yang berbeda, maka resiko keamanan akan muncul apabila memberlakukan *free access*. Dan ada hal lain juga yang harus dijamin dalam sistem terdistribusi, yaitu : penggunaan *rerources* yang tepat oleh user yang berlainan.

2.3.4 Model dalam Sistem Terdistribusi

Model dalam sistem terdistribusi :

1. Model Arsitektur (*Architectural Models*)
2. Model Interaksi (*Interaction Models*)
3. Model Kegagalan (*Failure Models*)

Resources dalam sistem terdistribusi dipakai secara bersama oleh *users*. Biasanya di bungkus (*encapsulated*) dalam suatu komputer dan dapat diakses oleh komputer lain dengan komunikasi

Setiap *resource* diatur oleh program yang disebut dengan *resource manager*. *Resource manager* memberikan kemungkinan komunikasi *interface* antar *resource*. *Resource managers* dapat digeneralisasi sebagai proses, kalau sistem di *design* dengan sudut pandang *object* (*Object Oriented*), *resource* dibungkus dalam suatu objek.

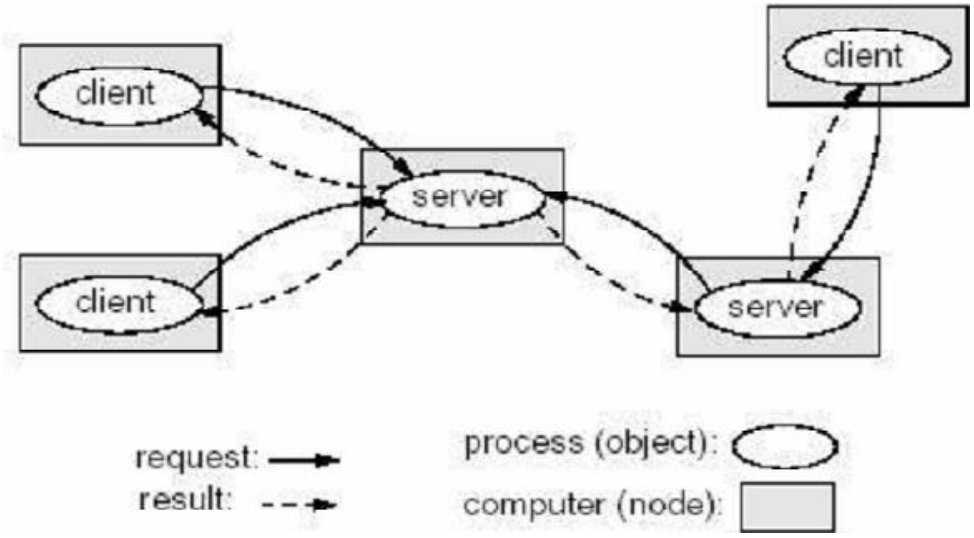
2.3.4.1 Architectural Model

Architectural models adalah cara kerja sistem terdistribusi antara komponen - komponen sistem dan bagaimana komponen tersebut berada pada sistem terdistribusi :

1. *Client - Server Model*

   Sistem yang terdiri dari kumpulan proses disebut dengan *server*, dan memberikan layanan kepada user yang disebut dengan *Client*. Model *Client - server* biasanya berbasiskan protokol *request/reply*. Contoh implementasinya, atara lain: RPC (*Remote Procedure Calling*) dan RMI (*Remote Method Invocation*). Cara kerja model *client-server* adalah:

   1. *Client* mengirimkan *request* berupa pesan ke *server* untuk mengakses suatu *service*.
   2. *Server* menerima pesan tersebut dan mengeksekusi request *client* dan me-*reply* hasil ke *client*



Gambar 2.17 Model Arsitektur *Client - Server*

2. *Proxy Server*

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

Proxy server dapat diletakkan pada setiap *client* atau dapat di pakai bersama oleh beberapa *client*. Tujuannya adalah meningkatkan *performance* dan *availibity* dengan mencegah frekwensi akses ke *server*.

3. *Peer Process*

Semua proses (*object*) mempunyai peran yang sama. Proses berinteraksi tanpa ada nya perbedaan antara *client* dan *server*. Pola komunikasi yang digunakan berdasarkan aplikasi yang digunakan. Merupakan model yang paling general dan *fleksible*.

2.3.5 Interaction Models

Untuk interaksi nya sistem terdistribusi dibagi menjadi dua bagian :

1. *Synchronous Distributed System*

Batas atas dan batas bawah waktu pengeksekusian dapat diset.Pesan yang dikirim di terima dalam waktu yang sudah di tentukan Fluktuasi ukuran antara waktu lokal berada dalam suatu batasan.Beberapa hal yang penting untuk di perhatikan.

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.

2. *Asynchronous Distributed System*

Banyak sistem terdistribusi yang menggunakan model interaksi ini (termasuk Internet). Ciri khas dari system ini :

1. Tidak ada batasan dalam waktu pengeksekusian.
2. Tidak ada batasan dalam *delay transmission* (penundaan pengiriman)
3. Tidak ada batasan terhadap fluktuasi waktu lokal.

Asynchronous system pada prakteknya lebih banyak digunakan.

2.3.6 Failure Models

Mendefiniskan kegagalanapa saja yang dapat terjadi dan bagaimana efek yang ditimbulkan. Jenis kegagalantersebut antara lain

1. *Omission Faluires*
2. *Arbitary Failures*
3. *Timing Failures*

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 kagagalan yang mungkin terjadi.

1. *Ommision Failures*

Ommision Failures adalah ketika prosesor dan kanal komunikasi mengalami kegagalan untuk melakukan hal yang seharusnya dilakukan. Dikatakan tidak mempunyai *Ommision Failures* apabila :

1. Terjadi keterlambatan (*delayed*) tetapi akhirnya tetap tereksekusi.
2. Sebuah aksi dieksekusi walaupun terdapat kesalahan pada hasil.

Dengan *synchronous system*, *omission 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.

2. *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.

3. *Timing Failures*

Timing Failures dapat terjadi pada *synchronous system*, dimana batas waktu di atur untuk eksekusi proses, komunikasi dan fluktuasi waktu. *Timing failures* terjadi apabila waktu yang telah ditentukan terlampaui.

2.4 Proses

2.4.1 KonsepProses

Jika membahas mengenai CPU, maka akan timbul sebuah pertanyaan yaitu mengenai istilah apa yang tepat untuk menyebut semua kegiatan yang dilakukan oleh CPU. Sistem *batch* mengeksekusi *jobs* sebagaimana suatu sistem *time-share* menggunakan program pengguna (user programs) atau *tasks*. Bahkan pada sistem dengan pengguna tunggal pun, seperti pada Microsoft Windows dan Macintosh OS, seorang pengguna mampu menjalankan beberapa program pada saat yang sama, contohnya *Word Processor*, *Web Browser*, dan paket *e-mail*. Bahkan jika pengguna hanya dapat menjalankan satu program pada satu waktu, sistem operasi perlu untuk mendukung aktivitas program internalnya sendiri, seperti managemen memori. Dalam banyak hal, seluruh aktivitas ini adalah serupa, maka kita menyebut seluruh program itu proses-proses.

Istilah job dan proses digunakan hampir dapat dipertukarkan pada tulisan ini. Walau istilah proses, banyak teori dan terminologi sistem operasi dikembangkan selama suatu waktu ketika aktivitas utama sistem operasi adalah *job processing*.

2.4.2 Definisi Proses

Secara tidak langsung, proses merupakan program yang sedang dieksekusi. Menurut Silberschatz, suatu proses adalah lebih dari sebuah kode program, yang terkadang disebut *text section*. Proses juga mencakup program *counter*, yaitu sebuah stack untuk menyimpan alamat dari instruksi yang akan

dieksekusi selanjutnya dan *register*. Sebuah proses pada umumnya juga memiliki sebuah stack yang berisikan data-data yang dibutuhkan selama proses dieksekusi seperti parameter metode, alamat *return* dan variabel lokal, dan sebuah data *section* yang menyimpan variabel global.

Sama halnya dengan Silberschatz, Tanenbaum juga berpendapat bahwa proses adalah sebuah program yang dieksekusi yang mencakup program *counter*, *register*, dan variabel di dalamnya. Suatu program adalah satu entitas pasif seperti isi dari sebuah berkas yang disimpan didalam disket. Sedangkan sebuah proses dalam suatu entitas aktif, dengan sebuah program *counter* yang menyimpan alamat instruksi selanjutnya yang akan dieksekusi oleh seperangkat sumber daya (*resource*).

Untuk mempermudah membedakan program dengan proses, akan digunakan analogi yang diberikan oleh Tanenbaum. Misalnya ada seorang tukang kue yang ingin membuat kue ulang tahun untuk anaknya. Tukang kue tersebut memiliki resep kue ulang tahun dan bahan-bahan yang dibutuhkan untuk membuat kue ulang tahun di dapurnya seperti: tepung terigu, telur, gula, bubuk vanila dan bahan-bahan lainnya. Dalam analogi ini, resep kue ulang tahun adalah sebuah program, si tukang kue tersebut adalah *prosesor* (CPU), dan bahan-bahan untuk membuat kue tersebut adalah data input. Sedangkan proses-nya adalah kegiatan sang tukang kue untuk membaca resep, mengolah bahan, dan memanggang kue tersebut.

Walau dua proses dapat dihubungkan dengan program yang sama, program tersebut dianggap dua urutan eksekusi yang berbeda. Sebagai contoh,

beberapa pengguna dapat menjalankan salinan yang berbeda pada *mail* program, atau pengguna yang sama dapat meminta banyak salinan dari program editor. Tiap-tiap proses ini adakah proses yang berbeda dan walau bagian *text section* adalah sama, data *section*-nya bervariasi. Adalah umum untuk memiliki proses yang menghasilkan banyak proses begitu ia bekerja.

2.4.3 Status Proses

Bila sebuah proses dieksekusi, maka statusnya akan berubah-ubah. Status dari sebuah proses mencerminkan aktivitas atau keadaan dari proses itu sendiri. Berikut ini adalah status-status yang mungkin dimiliki sebuah proses menurut Tanenbaum:

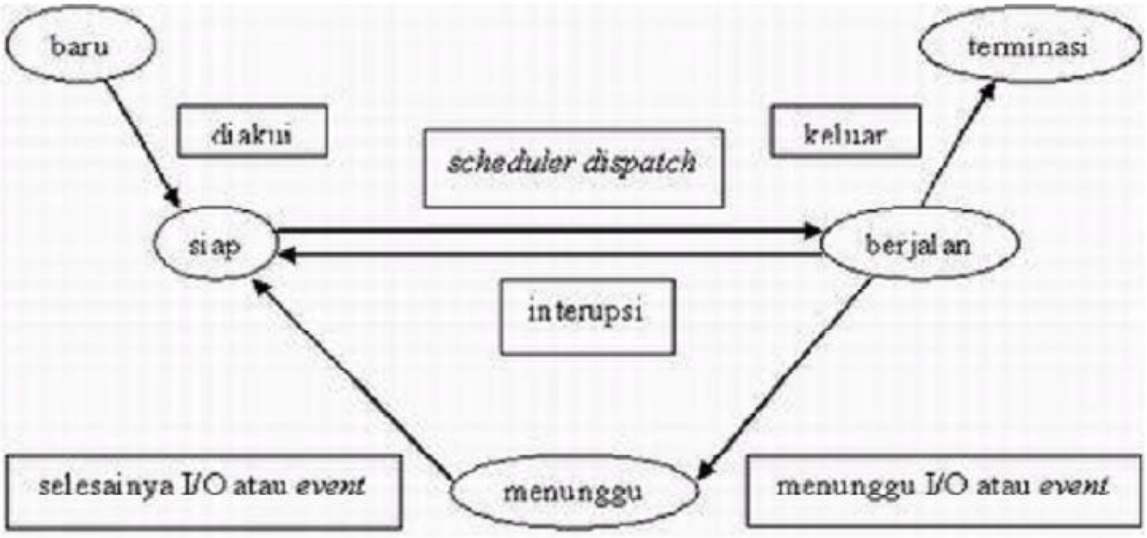
1. *Running* : pada saat menggunakan CPU pada suatu waktu.
2. *Ready* : proses diberhentikan sementara karena menunggu proses lain untuk dieksekusi.
3. *Blocked* : tidak dijalankan sampai *event* dari luar, yang berhubungan dengan proses tersebut terjadi.

Sedangkan menurut Silberschatz, terdapat lima macam jenis status yang mungkin dimiliki oleh suatu proses:

1. *New* : status yang dimiliki pada saat proses baru saja dibuat.
2. *Running* : status yang dimiliki pada saat instruksi-instruksi dari sebuah proses dieksekusi.
3. *Waiting* : status yang dimiliki pada saat proses menunggu suatu *event* (contohnya: proses I/O).

4. *Ready* : status yang dimiliki pada saat proses siap untuk dieksekusi oleh prosesor.
5. *Terminated* : status yang dimiliki pada saat proses telah selesai dieksekusi.

Nama-nama tersebut adalah berdasar opini, istilah tersebut bervariasi di sepanjang sistem operasi. Keadaan yang mereka gambarkan ditemukan pada seluruh sistem. Namun, pada sistem operasi tertentu lebih baik menggambarkan keadaan/status proses. Penting untuk diketahui bahwa hanya satu proses yang dapat berjalanpada prosesor mana pun pada satu waktu. Namun, banyak proses yang dapat berstatus *ready* atau *waiting*. Keadaan diagram yang berkaitan dengan keadaan tersebut dijelaskan pada gambar 2.17.



Gambar 2.18 Status proses

Ada tiga kemungkinan bila sebuah proses memiliki status running :

1. Jika program telah selesai dieksekusi maka status dari proses tersebut akan berubah menjadi *terminated*.

2. Jika waktu yang disediakan oleh OS untuk proses tersebut sudah habis maka akan terjadi *interrupt* dan proses tersebut kini berstatus *ready*.
3. Jika suatu *event* terjadi pada saat proses dieksekusi (seperti ada *request* I/O) maka proses tersebut akan menunggu *event* tersebut selesai dan proses berstatus *waiting*.

2.4.4 Proses Control Bl*ock*

Tiap proses digambarkan dalam sistem operasi oleh sebuah *process control block* (PCB) juga disebut sebuah *control block*. Sebuah PCB ditunjukkan dalam Gambar 3.2. PCB berisikan banyak bagian dari informasi yang berhubungan dengan sebuah proses yang spesifik, termasuk hal-hal di bawah ini:

1. Status proses : status mungkin, *new*, *ready*, *running*, *waiting*, *halted*, dan juga banyak lagi.
2. Program *counter*: suatu *stack* yang berisi alamat dari instruksi selanjutnya untuk dieksekusi untuk proses ini.
3. CPU *register*: *Register* bervariasi dalam jumlah dan jenis, tergantung pada rancangan komputer. *Register* tersebut termasuk *accumulator*, *register indeks*, *stack pointer*, *general purposes register*, ditambah *code information* pada kondisi apa pun. Beserta dengan program counter, keadaan/status informasi harus disimpan ketika gangguan terjadi, untuk memungkinkan proses tersebut berjalan/bekerja dengan benar setelahnya.

Tiap proses digambarkan dalam sistem operasi oleh sebuah PCB (*process control block)* juga disebut sebuah *control block*. PCB berisikan banyak bagian dari informasi yang berhubungan dengan sebuah proses yang spesifik.

4. Status proses : status mungkin, *new, ready*, *running*, *waiting*, *halted*, dan juga banyak lagi.
5. Program *counter*: suatu *stack* yang berisi alamat dari instruksi selanjutnya untuk dieksekusi untuk proses ini.
6. CPU *register*: *Register* bervariasi dalam jumlah dan jenis, tergantung pada rancangan komputer. *Register* tersebut termasuk *accumulator, register indeks, stack pointer, general purposes register*, ditambah *code information* pada kondisi apa pun. Beserta dengan program *counter*, keadaan/status informasi harus disimpan ketika gangguan terjadi, untuk memungkinkan proses tersebut berjalan/bekerja dengan benar setelahnya.
7. Informasi managemen memori: Informasi ini dapat termasuk suatu informasi sebagai nilai dari dasar dan batas *register*, tabel page/halaman, atau tabel segmen tergantung pada sistem memori yang digunakan oleh sistem operasi.
8. Informasi pencatatan: Informasi ini termasuk jumlah dari CPU dan waktu riil yang digunakan, batas waktu, jumlah akun jumlah *job* atau *proses*, dan banyak lagi.
9. Informasi status I/O: Informasi termasuk daftar dari perangkat I/O yang di gunakan pada proses ini, suatu daftar berkas-berkas yang sedang diakses

dan banyak lagi. PCB hanya berfungsi sebagai tempat penyimpanan informasi yang dapat bervariasi dari proses yang satu dengan yang lain.

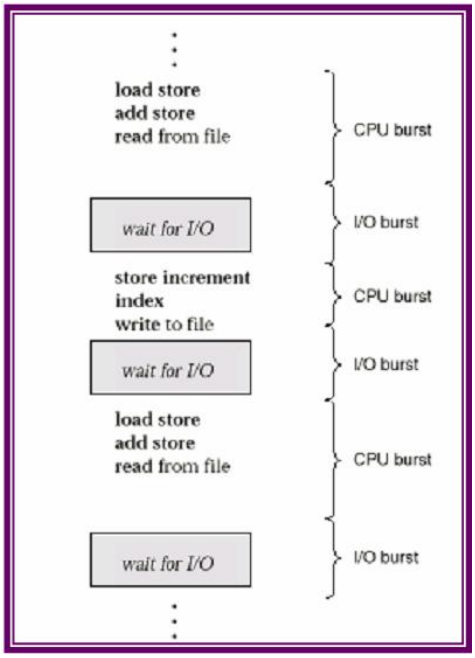


Gambar 2.19 *Proses Control Block*

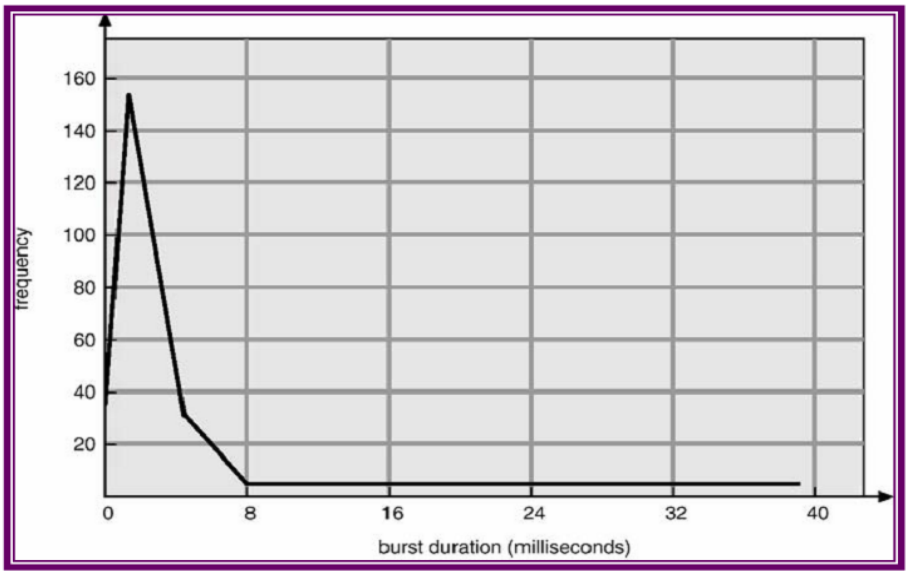
2.5 Penjadwalan CPU

Pada sistem *multiprogramming*, selalu akan terjadi beberapa proses berjalan dalam suatu waktu. Sedangkan pada *uniprogramming* hal ini tidak akan terjadi, karena hanya ada satu proses yang berjalan pada saat tertentu. Sistem *multiprogramming* diperlukan untuk memaksimalkan utilitas CPU. Pada saat proses dijalankan terjadi siklus eksekusi CPU dan menunggu I/O yang disebut dengan siklus CPU-I/O *burst*. Eksekusi proses dimulai dengan CPU *burst* dan

dilanjutkan dengan I/O *burst*, diikuti CPU *burst* lain, kemudian I/O *burst* lain dan seterusnya seperti pada Gambar 2.19.



Gambar 2.20 Siklus CPU-I/O *Burst*



Gambar 2.21 Histogram Waktu CPU *Burst*

Pada saat suatu proses dieksekusi, terdapat banyak CPU *burst* yang pendek dan terdapat sedikit CPU *burst* yang panjang. Program yang I/O *bound* biasanya sangat pendek CPU *burst*nya, sedangkan program yang CPU bound kemungkinan CPU burst nya sangat lama. Hal ini dapat digambarkan dengan grafik yang eksponensial atau *hipereksponensial* seperti pada Gambar 2.20. Oleh karena itu sangat penting pemilihan algoritma penjadwalan CPU.

2.5.1 CPU Scheduler

Pada saat CPU menganggur, maka sistem operasi harus menyeleksi proses-proses yang ada di memori utama (*ready queue*) untuk dieksekusi dan mengalokasikan CPU untuk salah satu dari proses tersebut. Seleksi semacam ini disebut dengan *shortterm scheduler* (CPU *scheduler*). Keputusan untuk menjadwalkan CPU mengikuti empat keadaan di bawah ini :

1. Apabila proses berpindah dari keadaan *running* ke *waiting*.
2. Apabila proses berpindah dari keadaan *running* ke *ready*.
3. Apabila proses berpindah dari keadaan *waiting* ke *ready*.
4. Apabila proses berhenti.

Apabila model penjadwalan yang dipilih menggunakan keadaan 1 dan 4, maka penjadwakan semacam ini disebut non-peemptive. Sebaliknya, apabila yang digunakan adalah keadaan 2 dan 3, maka disebut dengan *preemptive*. Pada *non-preemptive*, jika suatu proses sedang menggunakan CPU, maka proses tersebut akan tetap membawa CPU sampai proses tersebut melepaskannya (berhenti atau dalam keadaan *waiting*). *Preemptive scheduling* memiliki kelemahan, yaitu biaya

yang dibutuhkan sangat tinggi. Antara lain, harus selalu dilakukan perbaikan data. hal ini terjadi jika suatu proses ditinggalkan dan akan segera dikerjakan proses yang lain.

2.5.2 Dispatcher

Dispatcher adalah suatu modul yang akan memberikan kontrol pada CPU terhadap penyeleksian proses yang dilakukan selama *short term scheduling*. Fungsi-fungsi yang terkandung di dalam-nya mel*IP*uti:

1. *Switching context*.
2. *Switching* ke *user-mode*.
3. Melompat ke lokasi tertentu pada user program untuk memulai program.

Waktu yang diperlukan oleh dispatcher untuk menghentikan suatu proses dan memulai untuk menjalankan proses yang lainnya disebut *dispatch latency*.

2.5.3 Kriteria Penjadwalan

Algoritma penjadwalan CPU yang berbeda akan memiliki perbedaan properti. Sehingga untuk memilih algoritma ini harus dipertimbangkan dulu properti-properti algoritma tersebut. Ada beberapa kriteria yang digunakan untuk melakukan pembandingan algoritma penjadwalan CPU, antara lain:

1. CPU *utilization*. Diharapkan agar CPU selalu dalam keadaan sibuk. Utilitas CPU dinyatakan dalam bentuk prosen yaitu 0-100%. Namun dalam kenyataannya hanya berkisar antara 40-90%.
2. *Throughput*. Adalah banyaknya proses yang selesai dikerjakan dalam satu satuan waktu.

3. *Turnaround time*. Banyaknya waktu yang diperlukan untuk mengeksekusi proses, dari mulai menunggu untuk meminta tempat di memori utama, menunggu di *ready queue*, eksekusi oleh CPU, dan mengerjakan I/O.
4. *Waiting time*. Waktu yang diperlukan oleh suatu proses untuk menunggu di *ready queue*. *Waiting time* ini tidak mempengaruhi eksekusi proses dan penggunaan I/O.
5. *Response time*. Waktu yang dibutuhkan oleh suatu proses dari minta dilayani hingga ada respon pertama yang menanggapi permintaan tersebut.
6. *Fairness*. Meyakinkan bahwa tiap-tiap proses akan mendapatkan pembagian waktu penggunaan CPU secara terbuka (*fair*).

2.5.4 Algoritma Penjadwalan

Penjadwalan CPU menyangkut penentuan proses-proses yang ada dalam ready queue yang akan dialokasikan pada CPU. Terdapat beberapa algoritma penjadwalan CPU seperti dijelaskan pada sub bab di bawah ini.

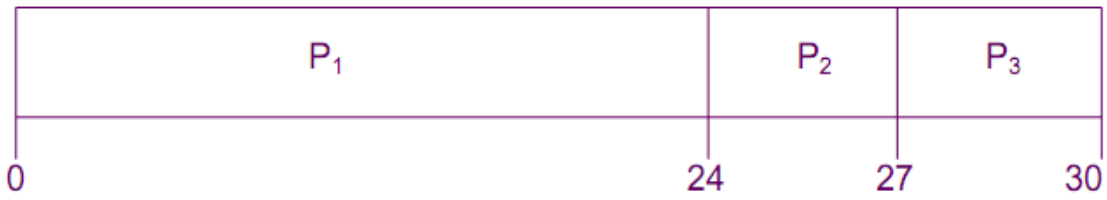
2.5.4.1 First-Come First-Served Scheduling (FCFS)

Proses yang pertama kali meminta jatah waktu untuk menggunakan CPU akan dilayani terlebih dahulu. Pada skema ini, proses yang meminta CPU pertama kali akan dialokasikan ke CPU pertama kali. Misalnya terdapat tiga proses yang dapat dengan urutan P1, P2, dan P3 dengan

Waktu CPU *burst* dalam milidetik yang diberikan sebagai berikut :

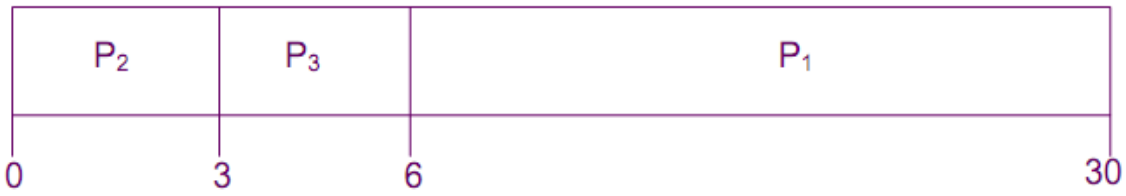
Process Burst	*Time*
P1	24
P2	3
P3	3

GantChart dengan penjadwalan FCFS adalah sebagai berikut :



Gambar 2.22 *Ganchart* FCFS(1)

Waktu tunggu untuk P1 adalah 0, P2 adalah 24 dan P3 adalah 27 sehingga rata-rata waktu tunggu adalah (0 + 24 + 27)/3 = 17 milidetik. Sedangkan apabila proses datang dengan urutan P2, P3, dan P1, hasil penjadwalan CPU dapat dilihat pada gant chart berikut :



Gambar 2.23 *Ganchart* FCFS(2)

Waktu tunggu sekarang untuk P1 adalah 6, P2 adalah 0 dan P3 adalah 3 sehingga rata-rata waktu tunggu adalah (6 + 0 + 3)/3 = 3 milidetik. Rata-rata waktu tunggu kasus ini jauh lebih baik dibandingkan dengan kasus sebelumnya. Pada penjadwalan CPU dimungkinkan terjadi *convoy effect* apabila proses yang pendek berada pada proses yang panjang.

Algoritma FCFS termasuk *non-preemptive*. karena, sekali CPU dialokasikan pada suatu proses, maka proses tersebut tetap akan memakai CPU sampai proses tersebut melepaskannya, yaitu jika proses tersebut berhenti atau meminta I/O.

2.5.4.2 Shortest Job First Scheduler (SJF)

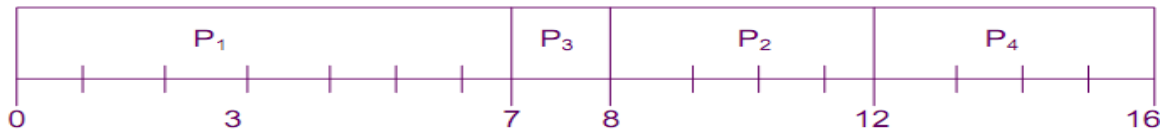
Pada penjadwalan SJF, proses yang memiliki CPU *burst* paling kecil dilayani terlebih dahulu. Terdapat dua skema :

1. *Nonpreemptive*, bila CPU diberikan pada proses, maka tidak bisa ditunda sampai CPU burst selesai.
2. *Preemptive*, jika proses baru datang dengan panjang CPU *burst* lebih pendek dari sisa waktu proses yang saat itu sedang dieksekusi, proses ini ditunda dan diganti dengan proses baru. Skema ini disebut dengan *Shortest Remaining Time First (SRTF)*.

SJF adalah algoritma penjadwalan yang optimal dengan rata-rata waktu tunggu yang minimal. Misalnya terdapat empat proses dengan panjang CPU burst dalam milidetik.

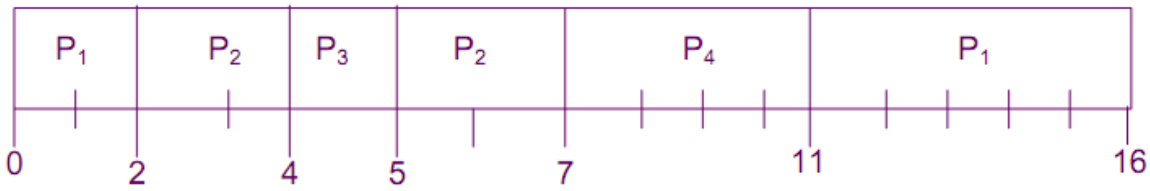
Process	*Arrival Time*	B*urst T*ime
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

Penjadwalan proses dengan algoritma SJF (*non-preemptive*) dapat dilihat pada *gant chart* berikut :



Gambar 2.24 *Client* SJF(1)

Waktu tunggu untuk P1 adalah 0, P2 adalah 26, P3 adalah 3 dan P4 adalah 7 sehingga rata-rata waktu tunggu adalah (0 + 6 + 3 + 7)/4 = 4 milidetik. Sedangkan Penjadwalan proses dengan algoritma SRTF (*preemptive*) dapat dilihat pada gant chart berikut :



Gambar 2.25 Gan Chart SRTF

Waktu tunggu untuk P1 adalah 9, P2 adalah 1, P3 adalah 0 dan P4 adalah 4 sehingga rata-rata waktu tunggu adalah (9 + 1 + 0 + 4)/4 = 3 milidetik. Mesk*IP*un algoritma ini optimal, namun pada kenyataannya sulit untuk diimplementasikan karena sulit untuk mengetahui panjang CPU *burst* berikutnya. Namun nilai ini dapat diprediksi. CPU *burst* berikutnya biasanya diprediksi sebagai suatu rata-rata eksponensial yang ditentukan dari CPU burst sebelumnya atau "*Exponential Average*".

$$\tau_{n+1} = \alpha t_0 + (1-\alpha)\tau_n$$

Dengan:

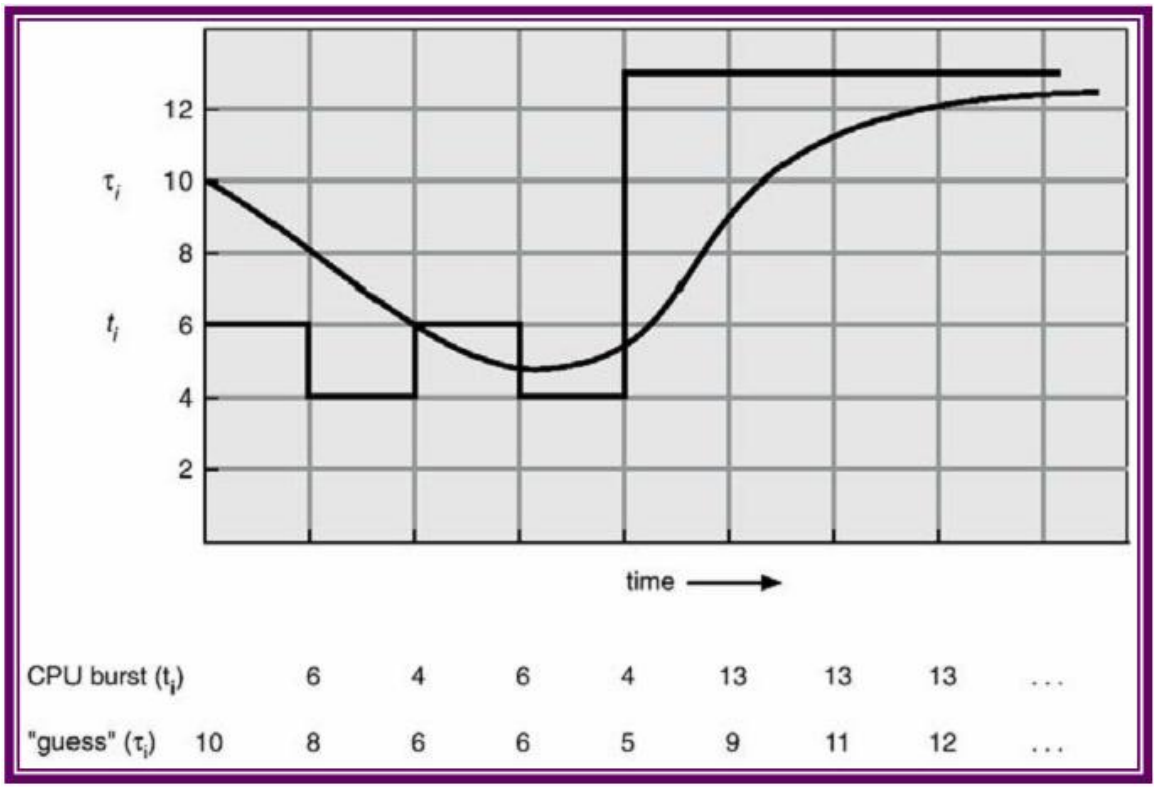
τ_{n+1} = panjang CPU burst yang diperkirakan

τ_0 = panjang CPU burst sebelumnya

τ_n = panjang CPU burst yang ke-*n* (yang sedang berlangsung)

α = ukuran pembanding antara τ_{n+1} dengan τ_n (0 sampai 1)

Grafik hasil prediksi CPU *burst* dapat dilihat pada Gambar 2.26.



Gambar 2.26 Grafik Prediksi CPU Burst

2.5.4.3 Priority Scheduling

Algoritma SJF adalah suatu kasus khusus dari penjadwalan berprioritas. Tiap - tiap proses dilengkapi dengan nomor prioritas (*integer*). CPU dialokasikan untuk proses yang memiliki prioritas paling tinggi (nilai *integer* terkecil biasanya merupakan prioritas terbesar). Jika beberapa proses memiliki prioritas yang sama, maka akan digunakan algoritma FCFS. Penjadwalan berprioritas terdiri dari dua skema yaitu *non preemptive* dan *preemptive*. Jika ada proses P1 yang datang pada saat P0 sedang berjalan, maka akan dilihat prioritas P1. Seandainya prioritas P1 lebih besar dibanding dengan prioritas P0, maka pada *non-preemptive*, algoritma tetap akan menyelesaikan P0 sampai habis CPU *burst*-nya, dan meletakkan P1 pada posisi *head queue*. Sedangkan pada *preemptive*, P0 akan dihentikan dulu, dan CPU ganti dialokasikan untuk P1. Misalnya terdapat lima proses P1, P2, P3, P4 dan P5 yang datang secara berurutan dengan CPU *burst* dalam milidetik.

Process	*Burst Time*	*Priority*
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

Penjadwalan proses dengan algoritma priority dapat dilihat pada gant chart berikut :



Gambar 2.27 *Ganchart* Algoritma *Priority*

Waktu tunggu untuk P1 adalah 6, P2 adalah 0, P3 adalah 16, P4 adalah 18 dan P5 adalah 1 sehingga rata-rata waktu tunggu adalah (6 + 0 +16 + 18 + 1)/5 = 8.2 milidetik.

2.5.4.4 Round-Robin Schedulling

Konsep dasar dari algoritma ini adalah dengan menggunakan *time-sharing*. Pada dasarnya algoritma ini sama dengan FCFS, hanya saja bersifat *preemptive*. Setiap proses mendapatkan waktu CPU yang disebut dengan waktu quantum (*quantum time*) untuk membatasi waktu proses, biasanya 1-100 milidetik. Setelah waktu habis, proses ditunda dan ditambahkan pada *ready queue*.

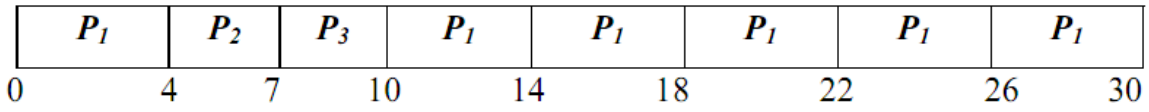
Jika suatu proses memiliki CPU burst lebih kecil dibandingkan dengan waktu quantum, maka proses tersebut akan melepaskan CPU jika telah selesai bekerja, sehingga CPU dapat segera digunakan oleh proses selanjutnya. Sebaliknya, jika suatu proses memiliki CPU *burst* yang lebih besar dibandingkan dengan waktu *quantum*, maka proses tersebut akan dihentikan sementara jika sudah mencapai waktu *quantum*, dan selanjutnya mengantri kembali pada posisi ekor dari *ready queue*, CPU kemudian menjalankan proses berikutnya.

Jika terdapat n proses pada *ready queue* dan waktu *quantum* q, maka setiap proses mendapatkan 1/n dari waktu CPU paling banyak q unit waktu pada sekali penjadwalan CPU. Tidak ada proses yang menunggu lebih dari (n-1)q unit waktu. Performansi algoritma *round robin* dapat dijelaskan sebagai berikut, jika q besar, maka yang digunakan adalah algoritma FIFO, tetapi jika q kecil maka sering terjadi *context switch*.

Misalkan ada 3 proses: P1, P2, dan P3 yang meminta pelayanan CPU dengan *quantum time* sebesar 4 milidetik.

Process	*Burst Time*
P1	24
P2	3
P3	3

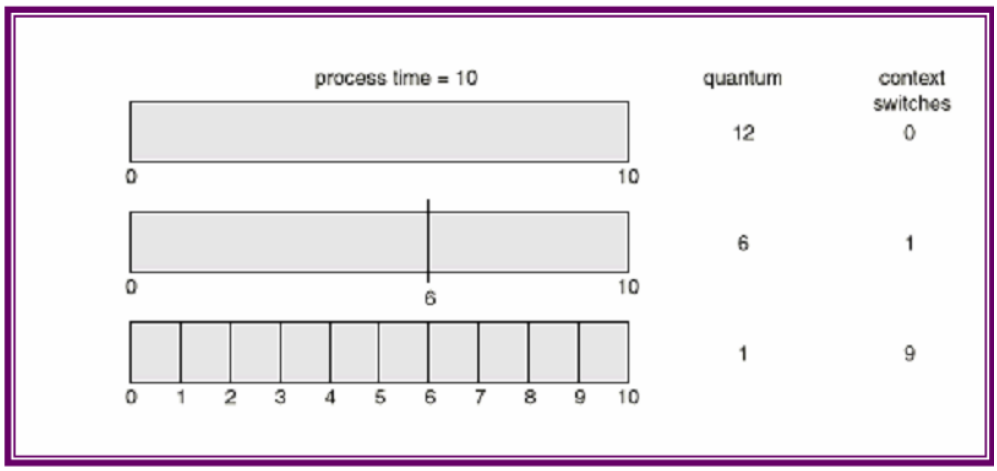
Penjadwalan proses dengan algoritma *round robin* dapat dilihat pada gant chart berikut :



Gambar 2.28 *Ganchart* Algoritma *Round Robin*

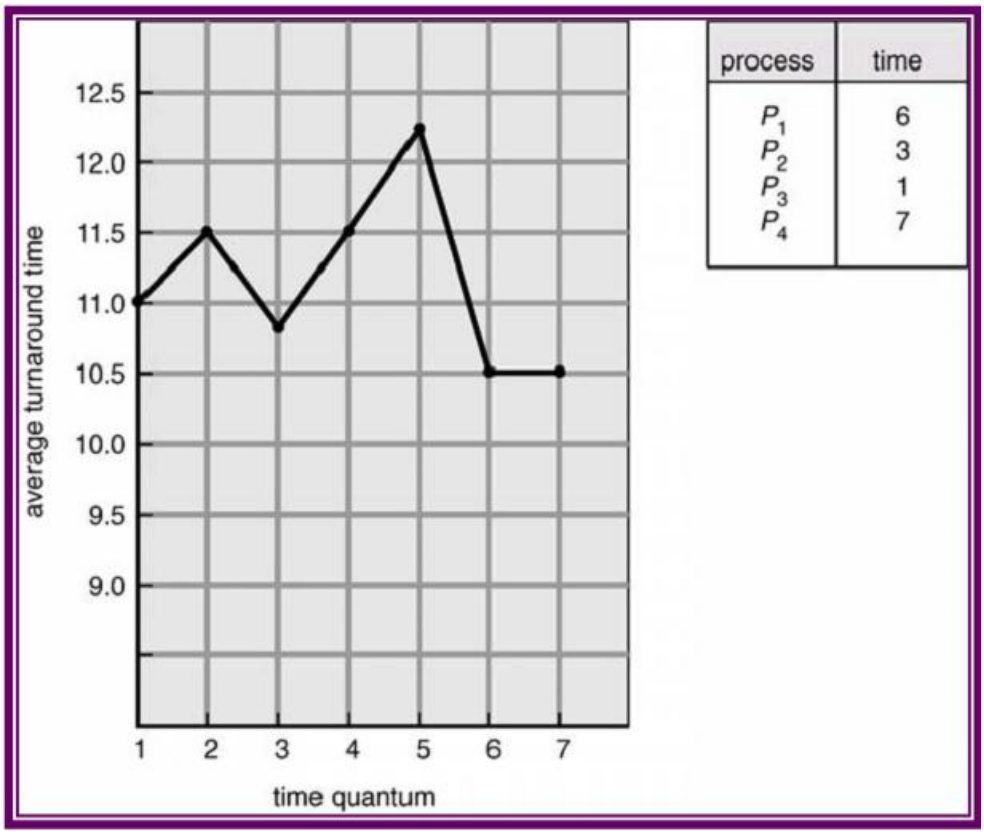
Waktu tunggu untuk P1 adalah 6, P2 adalah 4, dan P3 adalah 7 sehingga rata-rata waktu tunggu adalah (6 + 4 + 7)/3 = 5.66 milidetik.

Algoritma *round robin* ini di satu sisi memiliki keuntungan, yaitu adanya keseragaman waktu. Namun di sisi lain, algoritma ini akan terlalu sering melakukan switching seperti yang terlihat pada gambar 2.27. Semakin besar *quantum time*-nya maka *switching* yang terjadi akan semakin sedikit.



Gambar 2.29 Menunjukkan Waktu Kuantum yang Lebih Kecil

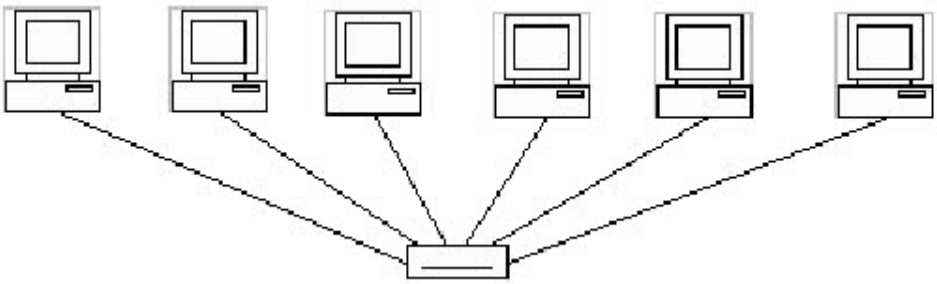
Meningkatkan waktu *turn around* juga tergantung ukuran waktu *quantum*. Seperti pada Gambar 4-5, rata-rata waktu *turn around* tidak meningkat bila waktu quantum dinaikkan. Secara umum, rata-rata waktu *turn around* dapat ditingkatkan jika banyak proses menyelesaikan CPU *burst* berikutnya sebagai satu waktu *quantum*. Sebagai contoh, terdapat tiga proses masing-masing 10 unit waktu dan waktu *quantum* 1 unit waktu, rata-rata waktu *turn around* adalah 29. Jika waktu *quantum* 10, sebaliknya, rata-rata waktu *turn around* turun menjadi 20.



Gambar 2.30 Menunjukkan Waktu *Turn Around* Berbeda pada Waktu *quantum*

2.6 *Clustering*

Cluster secara tata bahasa berarti seikat, sekelompok, segerombol. Sebagian besar orang mengenalnya dengan istilah *clustering*. Teknologi *cluster*ing mengacu suatu teknologi yang memungkinkan sejumlah komputer untuk bekerja sama menyelesaikan permasalahan komputasi, yaitu bisa berupa komputasi *sains* (dengan pemakaian CPU yang intensif) sampai beragam proses yang hanya mampu diselesaikan oleh sebuah superkomputer.



Gambar 2.31 Ilustrasi *Cluster*

Sistem *cluster* mengikat beberapa komputer agar menjadi suatu system dengan sumber daya komputasi tunggal yang mampu melakukan pekerjaan besar. Dari sisi user sebagai pengguna sistem tidak akan merasa bahwa pekerjaan yang dijalankan pada sistem tersebut sebenarnya tidak hanya dikerjakan oleh satu komputer saja, namun telah tersebar ke mesin komputer yang berbeda. Unit terkecil sebuah *cluster* disebut dengan *node*, yaitu komputer yang terkoneksi dan menjadi bagian dari sebuah sistem *cluster*.

Tidak semua komputer yang terkoneksi antara komputer yang satu dengan komputer yang lain dapat langsung membentuk sebuah *cluster*. Pada dasarnya adalah di dalam *cluster* harus telah ada suatu konfigurasi yang menyatakan bahwa suatu komputer merupakan bagian dari sistem *cluster* tertentu sehingga suatu komputer tidak harus secara *dedicated* menjadi bagian dari suatu *cluster*. Komputer-komputer di dalam suatu jaringan LAN atau bahkan jaringan yang lebih luas lagi dapat langsung memberikan dukungan menjadi sebuah *cluster*.

Cluster terbentuk karena sebelumnya telah ter-install sebuah program *software cluster* pada salah satu komputer dari sejumlah komputer yang saling terkoneksi tersebut. Komputer utama biasa disebut dengan *node* master.

Sedangkan komputer-komputer yang lain biasa disebut sebagai *node client*. Dari salah satu *node* di dalam suatu *cluster*, suatu proses dapat dijalankan dan secara *pre-emptive* atau *non-preemptive* proses tersebut akan tersebar ke *node-node* yang lain. Penyebaran proses dapat terjadi berdasarkan konfigurasi yang terdapat pada *node* master atau berdasarkan alur proses yang telah *embedded* di dalam program paralel yang dijalankan.

Secara sistem *hardware*, tidak ada keharusan bahwa komputer-komputer yang akan dijadikan sebagai bagian sebuah *cluster* memiliki spesifikasi yang tinggi. Hal inilah yang menyebabkan bahwa pembuatan sebuah *cluster* tidak memerlukan biaya yang tinggi. *Cluster* dengan *node-node* yang memiliki spesifikasi yang tinggi dan telah memperhitungkan komposisi yang tepat antara masing-masing *hardware* yang terlibat di dalamnya, serta telah di konfigurasikan dengan benar maka akan menghasilkan sebuah *cluster* dengan kemampuan yang sungguh luar biasa setara dengan sebuah *supercomputer*. Pada umumnya pengembangan sebuah *cluster* disesuaikan dengan melihat kebutuhan dan tujuan dari pemakaian *cluster* tersebut.

2.6.1 HPC Cluster

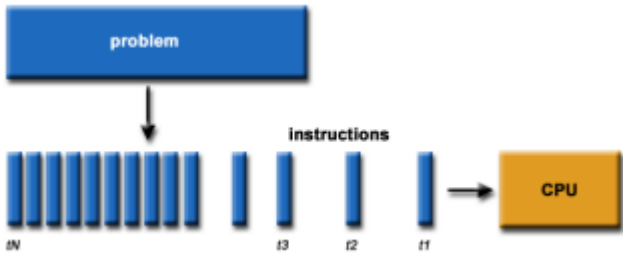
Proses pembagian kerja yang terjadi di dalam sebuah *cluster* bergantung pada jenis *cluster* yang digunakan. Jenis *cluster* yang paling populer adalah HPC *cluster*. Hpc *cluster* merupakan sebuah *cluster* dengan kemampuan yang scalable bergantung pada sistem *hardware*, sistem jaringan dengan akses terbatas (*private network* system), dan berjalan langsung di atas infrastruktur

sistem operasi yang open source software, yaitu Linux. Untuk membuat HPC *cluster* dibutuhkan perangkat lunak yang bertujuan sebagai *middleware* untuk mengirim pesan antar komputer salah satu nya adalah *Massive Passing Interface* (MPI).

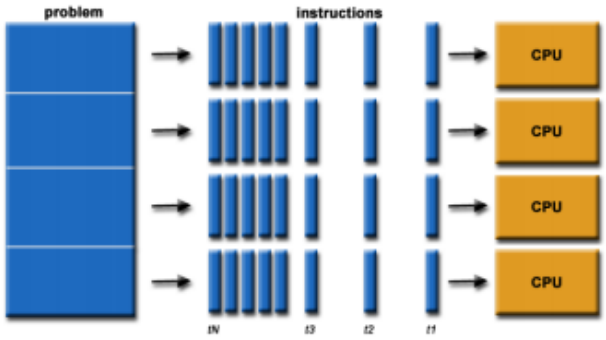
Tidak semua proses langsung dapat berjalandan mampu memanfaatkan kelebihan dari HPC *cluster*. Semua proses yang ingin dijalankan pada hpc *cluster* memerlukan perubahan, yaitu secara programming bahwa proses tersebut harus telah mendukung pekerjaan paralel dimana antara pekerjaan yang satu dengan pekerjaan yang lain harus berkomunikasi dengan menggunakan library yang telah disediakan, yaitu MPI . *Library-library* tersebut merupakan sistem software yang memberikan sarana bagi pengguna *cluster* agar dapat membuat sebuah program paralel dengan dukungan *message-passing* yang dapat berjalanpada sebuah *cluster* dengan menggunakan bahasa pemrograman Fortran atau C.

HPC *cluster* tidak memberikan dukungan pada proses yang menggunakan *shared memory*. Sebuah program yang *multithreaded* tidak secara otomatis akan memiliki peningkatan kecepatan dan kinerja apabila dijalankan pada HPC *cluster* karena program tersebut menggunakan *shared memory* di dalam prosesnya. Semua proses yang dijalankan pada HPC *cluster* harus mengalami perubahan dan dikembangkan dengan dukungan MPI. Sehingga hal tersebut menyebabkan pengguna dari HPC *cluster* adalah hanya mereka yang berada di lapangan *sains*, para akademisi dan komunitas riset. Karena hanya

sebagian besar dari mereka saja yang mampu menulis sebuah program dari nol (*code in house*) dan merupakan sebuah aplikasi yang *cluster*-aware.



Gambar 2.32 Cara kerja komputer tunggal



Gambar 2.33 Cara kerja HPC *cluster*

2.6.2 Perangkat Lunak MPI

Message Passing Interface (MPI) merupakan implementasi standar dari model "*message passing*" komputasi *parallel*. Sebuah komputasi parallel terdiri dari sejumlah proses, dimana masing-masing bekerja pada beberapa data. Setiap prsoses mempunyai *variable* , dan tidak ada mekanisme suatu proses yang mengakses secara langsung memori yang lain. Sharing data antar proses-proses dilakukan dengan *message passing*., yaitu dengan mengirim dan menerima message antar proses-proses. Alasan menggunakan model ini sudah sangat umum dan tipe komputasi *parallel* dapat dibuat dalam bentuk *message passing*.

Model ini juga dapat diimplementasikan pada bermacam-macam platform, seperti *shared-memory multiprocessors* maupun *single processor*.

2.6.3 Penggunaan MPI

MPI perlu digunakan ketika kita menginginkan :

1. Menulis kode *parallel* yang portable
2. Mendapatkan performa yang tinggi dalam pemrograman parallel
3. Menghadapi permasalahan yang melibatkan hubungan data *irregular* atau dinamis yang tidak begitu cocok dengan model data-*parallel*.

Fitur dasar dari program-program *Message Passing Program message* passing terdiri dari *multiple* instan dari program serial yang berkomunikasi dengan pemanggilan *library*. Pemanggilan ini dapat dibagi menjadi empat kelas :

1. Pemanggilan untuk inisialisasi, mengatur, dan memutuskan komunikasi.
2. Pemanggilan untuk mengkomunikasikan beberapa pasang prosesor.
3. Pemanggilan yang mewujudkan operasi-operasi komunikasi diantara kelompok-kelompok prosesor.
4. Pemanggilan untuk membuat tipe data acak (*arbitrary*).

Kelas pertama dari pemanggilan terdiri dari pemanggilan untuk memulai komunikasi, mengidentifikasi jumlah prosesor yang sedang digunakan, membuat group dari prosesor-prosesor dan mengidentifikasi prosesor mana yang sedang menjalanjutkan instan khusus dari

program.Kelas kedua dari pemanggilan, dinamakan komunikasi *point-to-point*, terdiri atas tipe pengiriman dan penerimaan yang berbeda. Kelas ketiga dari pemanggilan adalah operasi kolektif yang memberikan sinkronisasi antar grup proses dan pemanggilan yang melakukan operasi komunikasi/kalkulasi. Kelas terakhir dari pemanggilan memberikan fleksibilitas dalam berurusan dengan struktur data yang rumit.

2.6.4 Fungsi-fungsi Dalam MPI

Berikut adalah fungsi-fungsi dari *library* MPI yang digunakan dalam pembuatan HPC *cluster*.

Tabel 2.4 Fungsi – fungsi MPI

Nama Fungsi	Tujuan Fungsi
MPI_Init	Inisialisasi lingkungan MPI. Fungsi ini hanya boleh dipanggil sekali, sebelum penggunakan fungsi-fungsi MPI yang lain.
MPI_Comm_size	Untuk menentukan jumlah prosesor yang tergabung dalam sebuah communicator (contohnya MPI_COMM_WORLD)
MPI_Comm_rank	Untuk menentukan rank dari prosesor yang memanggil fungsi ini dalam sebuah communicator (contohnya MPI_COMM_WORLD)
MPI_Abort	Untuk menghentikan semua proses MPI yang berjalan dalam communicator.

MPI_Wtime	Untuk melihat waktu yang telah berjalan semenjak panggilan terakhir terhadap fungsi ini sebelumnya.
MPI_Finalize	Untuk menghentikan lingkungan eksekusi MPI pada communicator.
MPI_Send	Fungsi dasar untuk mengirim data.
MPI_Recv	Fungsi dasar untuk memerintahkan komputer agar menunggu data yang akan masuk.
MPI_Bcast	Mem-*broadcast* sebuah pesan ke semua prosesor yang tergabung dalam sebuah communicator (contohnya MPI_COMM_WORLD)