BAB 2

LANDASAN TEORI

2.1 Video

Video atau gambar bergerak adalah data digital yang terdiri dari beberapa gambar. Istilah video biasanya mengacu pada beberapa format penyimpanan gambar bergerak. Terbagi menjadi dua, yaitu video analog, contohnya VHS dan Betamax, dan video digital, contohnya DVD, Quicktime, dan MPEG-4. Video dapat direkam dan ditransmisikan dalam berbagai media fisik, pada pita maknetik ketika direkam sebagai Pal atau NTSC *signal* elektrik dengan video camera, atau MPEG-4 ketika direkam menggunakan kamera digital. Gambar tersebut dapat ditemui di kehidupan sehari-hari, seperti saat melihat benda-benda yang bergerak. Melalui sebuah teknologi (kamera), gambar-gambar bergerak dapat disimpan pada sebuah data digital (video). Untuk dapat menampilkan data digital (video) ini maka diperlukan sebuah alat untuk mengubah data digital menjadi analog misalnya televisi dan monitor.

2.1.1 Karakteristik Video Digital

Video digital pada dasarnya tersusun atas serangkaian *frame*. Rangkaian *frame* tersebut ditampilkan pada layar dengan kecepatan tertentu, bergantung pada laju *frame* yang diberikan (dalam *frame*/detik). Jika laju *frame* cukup tinggi, mata manusia tidak dapat menangkap gambar per *frame*, melainkan menangkapnya sebagai rangkaian yang kontinue. Masing-masing *frame* merupakan gambar/ citra

(*image*) digital. Suatu image digital direpresentasikan sebuah matriks yang masing-masing elemennya merepresentasikan nilai intensitas. Jika I adalah matriks dua dimensi, I (x,y) adalah nilai intensitas yang sesuai pada posisi baris x dan kolom y pada matriks tersebut. Titik-titik di tempat *image* di-*sampling* disebut *picture elements,* atau sering dikenal sebagai piksel.

Karakteristik suatu video digital ditentukan oleh resolusi (*resolution)* atau dimensi *frame* (*frame dimention*), kedalaman piksel (*pixwl depth)*, dan laju *frame* (*frame rate*). Karakteristik-karakteristik ini menentukan perbandingan antara kualitas video dan jumlah bit yang dibutuhkan untuk menyimpan dan mentransmisikannya.

2.1.2 Resolusi

Resolusi (*resolution*) atau dimensi *frame* (*frame dimention*) adalah ukuran sebuah *frame*, yang dinyatakan dalam piksel x piksel. Semakin tinggi resolusinya, semakin baik kualitas video tersebut, dalam pengertian bahwa dalam ukuran fisik yang sama, video dengan resolusi tinggi akan lebih detil. Tetapi, resolusi yang tinggi akan mengakibatkan jumlah bit yang diperlukan untuk menyimpan atau mentransmisikannya meningkat.

2.1.3 Kedalaman Bit

Kedalaman bit (*bit depth*) menentukan jumlah bit yang digunakan untuk merepresentasikan tiap piksel pada sebuah frame, dan dinyatakan dalam bit/piksel. Semakin banyak jumlah bit yang digunakan untuk merepresentasikan sebuah piksel, yang berarti semakin tinggi kedalaman pikselnya, maka semakin tinggi pula kualitasnya, dengan bayaran jumlah bit yang diperlukan menjadi lebih tinggi.

Dengan 1 byte (8 bit) untuk tiap piksel, diperoleh 2^8 atau 256 level intensitas. Dengan level intensitas tersebut, umumnya mata manusia sudah dapat dipuaskan. Kedalaman piksel paling rendah terdapat pada *binary-value image* yang hanya menggunakan 1 bit untuk tiap piksel, sehingga hanya ada dua kemungkinan bagi tiap piksel, yaitu 0 (hitam) atau 1 (putih). Dengan demikian, semakin sedikit jumlah bit yang digunakan untuk tiap piksel, maka semakin turun pula kualitas gambar.

2.1.4 Laju *Frame*

Laju *frame* (*frame rate*) menunjukan jumlah *frame* yang digambar tiap detik dinyatakan dalam *frame*/detik. Sehubungan dengan laju *frame* ini, ada dua hal yang perlu diperhatikan, yaitu kehalusan gerakan (*smooth motion*) dan kilatan (*flash*). Kehalusasn gerakan ditentukan oleh jumlah *frame* yang berbeda per detik. Untuk mendapatkan gerakan yang halus, video digital setidaknya harus berada pada 25 *frame*/detik. Kilatan ditentukan oleh jumlah berapa kali layar digambar per detik. Dengan 20 *frame*/detik, kilatan sudah dapat dilenyapkan.

Video yang berkualitas baik akan memiliki laju *frame* minimal 30 *frame*/detik, setidaknya sesuai dengan mata manusia, yang membutuhkan jumlah bit yang lebih tinggi.

2.1.5 Representasi Warna

Pada umumnya, data video dipisahkan menjadi komponen warna (*crominance*) dan komponen kecerahan (*luminance*). Komponen tersebut dipisahkan dengan beberapa cara, yaitu: RGB, YUV, dan YIQ.

RGB

Data video dapat dipisahkan menjadi komponen-komponen untuk masing-masing warna, yaitu merah (*red*), hijau (*green*), dan biru (*blue*). Warna tiap piksel ditentukan oleh kombinasi intensitas dari masing-masing komponen warna.

YUV

Pemisahan data video menjadi komponen kecerahan dan komponen warna dapat dilakukan sesuai dengan format PAL, komponen kecerahan dinyatakan dengan Y, sedangkan dua komponen warna dinyatakan dengan U dan V. Masing-masing komponen tersebut diperoleh dengan mentransformasikan RGB dengan rumus:

$$Y = 0{,}299\ R + 0{,}587\ G + 0{,}114\ B$$

$$U = (B-Y) \times 0{,}493$$

$$V = (R-Y) \times 0{,}877$$

YIQ

Pemisahan data video menjadi komponen kecerahan dan komponen warna dapat dilakukan juga sesuai dengan format NTSC, komponen kecerahan dinyatakan dengan Y, sedangkan dua komponen warna dinyatakan dengan I dan Q. Masing-masing komponen tersebut diperoleh dengan mentransformasikan RGB dengan rumus:

$$Y = 0{,}299\ R + 0{,}587\ G + 0{,}114\ B$$

$$I = 0{,}596\ R - 0{,}275\ G - 0{,}321\ B$$

$$Q = 0{,}212\ R - 0{,}523\ G - 0{,}311\ B$$

Oleh karena persepsi mata manusia lebih peka pada kecerahan dari pada warna, maka *crominanc*e cukup di-*sampling* separuh dari *luminance*.

2.2 Audio

Gelombang audio adalah gelombang akustik satu dimensi. Ketika suatu gelombang akustik memasuki telinga, gendang telinga akan bergetar, yang menyebabkan tulang rawan telinga tengah bergetar juga, dan mengirimkan pulsa-pulsa saraf ke otak. Pulsa-pulsa saraf ini diketahui sebagai suara oleh pendengar. Dengan cara yang sama, ketika suatu gelombang akustik mengenai sebuah mikropon, maka mikropon akan menghasilkan signal listrik, yang merupakan amplitudo suara sebagai fungsi waktu. Representasi, pengolahan, penyimpanan, dan transmisi signal-sidnal audio seperti itu merupakan bagian penting penelitian system multimedia.

Range frekuensi pendengaran manusia berada di antara 20Hz hingga 20.000Hz, walaupun sebagian binatang, terutama anjing, dapat mendengar frekuensi yang lebih tinggi. Telinga mendengar gelombang suara secara logaritmik, karena itu ratio dua suara dengan amplitude A dan B secara konvesi diekspresikan dalam dB(decibel) menurut rumus:

$$dB=20\ Log_{10}(A/B)$$

bila mendefinisikan batas bawah daya dengar (suatu tekanan kira-kira 0,0003 dyne/cm^2) bagi sebuah gelombang sinus 1 kHz sama dengan 0 dB, maka percakapan biasa akan sama dengan kira-kira 50 dB dan ambang sakit kira-kira pada 120 dB, range dinamik dengan factor 1 juta. Agar tidak membingungkan, *A*

dan *B* diatas adalah *amplitude.* Bila kita memakai tingkat daya, yang proporsional dengan kuadrat amplitude, koefisien logaritma akan sama dengan 10, bukan 20.

Telinga sensitive terhadap variasi suara yang hanya berlangsung beberapa milidetik. Sebaliknya, mata tidak mengetahui adanya perubahan tingkat cahaya yang berlangsung beberapa milidetik. Hasil pengamatan ini adalah bahwa kejutan-kejutan beberapa milidetik selama transmisi multimedia mempengaruhi kualitas suara yang diterima lebih kuat disbanding dengan kualitas gambar yang diterima.

Gelombang suara dapat dikonversi ke dalam bentuk digital dengan ADC (*Analog Digital Converter*). Sebuah ADC mengambil tenaga listrik sebagai input dan menghasilkan bilangan biner sebagai output.

2.3 Jaringan Komputer

2.3.1 Konsep Dasar Jaringan Komputer

Istilah komunikasi berasal dari bahasa Inggris *communication.* Kata *communication* berasal dari bahasa Latin *communicare* yang berarti saling berbagi (*share*).

Komunikasi dapat diartikan sebagai proses menampilkan, mengubah, menginterpresentasikan, atau mengolah informasi antara manusia atau mesin. Proses ini melibatkan satu pengirim (*transmitter*), penerima (*receiver*), dan sebuah medium transmisi untuk tempat mengalirnya informasi.

Contohnya adalah perbincangan antara dua orang. Pada contoh ini, manusia bertukar informasi dengan cara berbicara. Suara yang berasal dari orang-

orang tersebut akan mengubah tekana udara yang pada akhirnya ketika sampai di telinga pendengar akan menggetarkan gendang telinga si pendengar tersebut.

Permasalahan yang timbul pada bentuk dasar komunikasi tersebut diatas adalah jarak. Jika jarak antara kedua orang yang sedang berbicara tersebut bertambah, informasi yang dibawa oleh udara akan kehilangan kekuatannya sebelum dapat mencapai tujuan. Dengan demikian, kebutuhan akan medium transmisi yang lebih baik muncul bila jarak antara dua pihak yang saling berkomunikasi berjauhan. Komunikasi jarak jauh ini disebut *telekomunikasi.*

2.3.2 Definisi Jaringan Komunikasi

Jaringan komunikasi dapat diartikan sebagai suatu system yang terbentuk dari interkoneksi failitas-fasilitas yang dirancang untuk membawa trafik dari beragam sumber telekomunikasi.

Suatu jaringan terdiri dari *link* dan *node.* Isitilah *node* digunakan untuk merepresentasikan sentral, *junction* atau keduanya. Istilah *link* digunakan untuk merepresentasikan kabel, peralatan terminasi, dan sebagainya. Sedangkan trafik adalah informasi yang terdapat di dalam jaringan, yang mengalir melalui *node* dan *link.*

Suatu jaringan komunikasi merupakan sumberdaya yang dapat dipakai secara bersama (*shared*) oleh sejumlah *end user* untuk berkomunikasi dengan *user* lain yang lokasinya berjauhan. Tidak semua *user* menggunakan jaringan pada waktu yang bersamaan, oleh karena itu merupakan suatu hal logis apabila sumber daya jaringan yang sangat penting ini dipakai secara bersama-sama. Penggunaan

sumber daya secara bersama ini melahirkan konsep sentral. Apabila tidak ada sentral di dalam jaringan, maka untuk lokasi sebanyak *m* diperlukan:

1. Jumlah link sebanyak m(m-1)/2
2. Jumlah telepon yang diperlukan m(m-1)

Bila digunakan satu sentral, maka jumlah link dan telepon yang diperlukan dapat dikurangi hanya sampai m buah.

2.3.3 Model Referensi Jaringan Komputer

Suatu jaringan komunikasi data memerlukan tingkat *compability* dan *interopability* yang tinggi diantara elemen-elemen jaringan, khususnya yang berhubungan dengan interface fisik dan lojik serta pengendaliannya. Dengan pertimbangan masalah-masalah tersebut, *international Organization for Standarization* (ISO) membentuk suatu subkomite (*Technical Committee 77*) pada tahun 1977 yang bertugas mengembangkan arsitektur standard dengan tujuan memperoleh sasaran jangka panjang berupa suatu *open system interconnection* (ISO). Komite inilah yang membuatstandard komunikasi data yang ada sekarang.

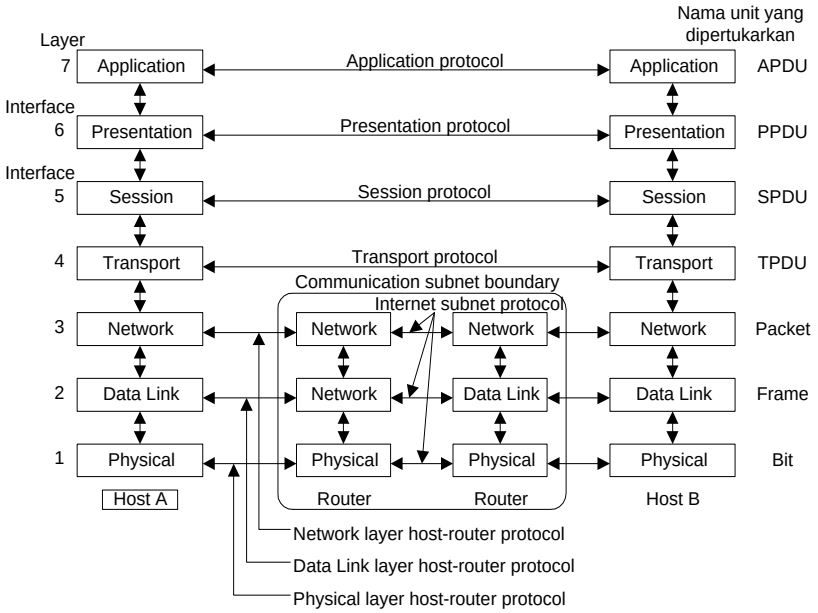
2.3.3.1 Model Referensi OSI

Model ini diciptakan berdasarkan sebuah proposal yang dibuat oleh *The International Standards Organization* (ISO) sebagai langkah awal menuju standarisasi protokol internasional yang digunakan pada berbagai layer. Model ini disebut ISO OSI (*Open Systems Interconnection*) *Reference Model*, karena model ini ditunjukan bagi penyambungan system terbuka. System terbuka dapat

diartikan sebagai suatu system yang terbuka untuk berkomunikasi dengan system-sistem lainnya.

Model OSI memiliki tujuh layer seperti pada gambar 2.1 . Prinsip-prinsip yang digunakan bagi ketujuh layer tersebut adalah:

1. Sebuah layer harus dibuat bila diperlukan tingkat abstraksi yang berbeda.
2. Setiap layer harus memiliki fungsi-fungsi tertentu.
3. Fungsi setiap layer harus dipilih dengan teliti sesuai dengan ketentuan protokol berstandar internasional.
4. Batas-batas layer harus dipilih untuk meminimalkan aliran informasi yang melewati *interface.*
5. Jumlah layer harus cukup banyak, sehingga fungsi-fungsi yang berbeda tidak perlu disatukan dalam satu layer diluar keperluannya. Akan tetapi jumlah layer juga harus diusahakan sedikit mungkin sehingga pemakaian arsitektur jaringan tidak menjadi sulit.



Gambar 2.1 Lapisan OSI

Ketujuh lapisan dari model referensi OSI dapat dibagi kedalam dua kategori, yaitu lapisan atas dan lapisan bawah.

Lapisan paling atas dari model OSI berurusan dengan persoalan aplikasi dan pada umunya diimplementasi hanya pada *software.* Lapisan tertinggi adalah lapisan penutup sebelum ke pengguna, pengguna dan lapisan aplikasi saling berinteraksi proses dengan *software* aplikasi yang berisi sebuah komponen komunikasi.

Lapisan paling bawah dari model OSI mengendalikan persoalan transport data. Lapisan fisik dan lapisan data link diimplementasikan ke dalam *hardware* dn *software.* Lapisan-lapisan bawah yang lain pada umunya hanya diimplementasikan dalam *software*. Lapisan fisik adalah lapisan penutup bagi media jaringan fisik (misalnya jaringa kabel), dan sebagai penanggung jawab bagi

penempatan informasi pada media jaringan. Table 2.1 di bawah ini menampilakan pemisahan antara lapisan atas dan lapisan bawah model OSI.

Tabel 2.1 Pemisahan Lapisan Atas dan Lapisan Bawah Pada Model OSI

Application	Application	Lapisan Atas
Presentation		
Session		
Transport	Data Transport	Lapisan Bawah
Network		
Data Link		
Physical		

2.3.3.1.1 Lapisan-lapisan Model OSI

2.3.3.1.1.1 Lapisan Fisik

Lapisan fisik (*Physical Layer*) berfungsi dalam pengiriman *raw bit* ke kanal komunikasi. Masalah yang harus diperhatikan disini adalah memastikan bahwa bila satu sisi mengirimkan data 1 bit, maka data tersebut harus diterima oleh sisi lainnya sebagai 1 bit juga, bukan 0 bit.

2.3.3.1.1.2 Lapisan Jalur Data

Lapisan jalur data (data *link layer*) berfungsi sebagai fasilitas transmisi *raw data* dan mentransformasikan data tersebut ke saluran yang bebas dari kesalahan transmisi sebelum diteruskan ke *network layer*, *data link layer* melaksanakan tugas ini dengan memungkinkan pengirim memecah-mecah data input menjadi sejumlah data *frame*(biasanya berjumlah ratusan atau ribuan byte).

Selanjutnya data *frame* mentransmisikan *frame* tersebut secara berurutan, dan memproses *acknowledgement frame* yang dikirim kembali oleh penerima. Karena *physical layer* menerima dan mengirim aliran bit tanpa mengindahkan arti atau arsitektur *frame*, maka tergantung pada data link layer untuk membuat dan mengenali batas-batas *frame* itu.

2.3.3.1.1.3 Lapisan Jaringan

Lapisan jaringan (*network layer*) berfungsi untuk pengendalian operasi subnet. Masalah desain yang penting adalah cara menentukan *route* pengiriman paket dari sumber ke tujuannya. *Route* dapat dipusatkan.

Bila pada saat yang sama dalam sebuah *subnet* terdapat terlalu banyak paket, ,aka paket-paket tersebut akan saling memasuki jalan paket lainnya, sehingga menimbulkan kemacetan. Pengendalian kemacetan seperti ini juga merupakan tugas network layer.

2.3.3.1.1.4 Lapisan *Transport*

Lapisan transport (*transport layer*)berfungsi menerima data dari *session layer*, bila perlu memeceah data menjadi bagian-bagian yang lebih kecil, meneruskan potongan data ke *network layer*, dan menjamin bahwa semu potongan data tersebut bisa tiba di sisi lainnya dengan benar. Semuanya harus dilakukan secara efisien agar dapat melindungi layer-layer bagian atas dari perubahan teknologi perangkat keras.

Dalam keadaan normal, *transport layer* membuat hubungan jaringan yang berbeda bagi setiap hubungan transport yang diperlukan oleh *session layer*. Bila hubungan transport memerlukan *throughput* yang tinggi, maka *transport layer*

dapat membuat hubungan jaringan yang banyak. Dengan kata lain transport *layer* memberikan hubungan jaringan yang proporsional sesuai dengan kebutuhan *session layer*.

Transport layer juga menentukan jenis layanan untuk *session layer* dan jenis layanan bagi pengguna jarinagan. Jenis transport layer yang paling popular adalah saluran *error free point to point* yang meneruskan pesan atau byte sesuai dengan urutan pengirimannya.

Transport *layer* merupakan layer *end-to-end* sejati, dari sumber ke tujuan. Artinya sebuah program pada mesin sumber membawa pecakapan dengan program yang sama pada mesin yang dituju. Pada *layer-layer* bawah, protokol terdapat diantara kedua mesin dan mesin-mesin lain yang berada di dekatnya. Protokol tidak terdapat pada mesin sumber terluar atau mesin tujuan terluar, yang mungkin dipisahkan oleh sebuah *router*. Perbedaan antara layer 1 sampai 3 yang terjalin, dan layer 4 sampai 7 yang *end-to-end*.

2.3.3.1.1.5 Lapisan Sesi

Lapisan sesi (*Session layer*) mengijinkan pengguna untuk menetaokan *session* dengan pengguna lainnya. Sebuah *session* selain memungkinkan transport data biasa, seperti yang dilakukan oleh *transport layer*, juga menyediakan layanan yang istimewa untuk aplikasi-aplikasi tertentu. Sebuah *session* digunakan untuk memungkinkan seorang pengguna untuk log ke *remote timesharing system* atau untuk memindahkan file dari satu mesin kemesin lainnya.

Sebuah layanan *session layer* adalah untuk melaksanakan pengendalian dialog. s*ession* dapat memungkinkan lalu lintas bergerak dalam bentuk dua arah

pada suatu saat, atau hanya satu arah saja. Jika suatu saat lalu lintas hanya satu arah saja (analog dengan rel kereta api tunggal), *session layer* membantu untuk menentukan giliran yang berhak menggunakan saluran pada suatu saat.

Layanan *session* si atas disebut manajemen *token*. Untuk sebagian protokol, adalah penting untuk memastikan bahwa kedua pihak yang bersangkutan tidak melakukan operasi pada saat yang sama. Untuk mengatur aktivitas ini, *session layer* nenyediakan *token-token* yang dapat digilirkan. Hanya pihak yang memegang token yang diijinkan melakukan operasi kritis.

Layanan *session* lainnya adalah *sinkronisasi*. Fungsi layanan *sinkronisasi* pada *session leyer* ini adalah menyisipkan sebuah *checkpoint* pada saat pengiriman data, sehingga jika terjadi *crash*, maka pengiriman data tidak dimulai dari awal melainkan data sesudah *checkpoint* saja yang dikirimkan.

2.3.3.1.1.6 Lapisan Presentasi

Lapisan presentasi (*Presentation layer*) melakukan fungsi-fungsi tertentu yang sering diminta untuk menjamin penemuan sebuah penyelesaian umum bagi masalah tertentu. Presentation layer tidak mengizinkan pengguna untuk menyelesaikan endiri suatu masalah. Tidak seperti layer-layer di bawahnya yang hanya melakukan pemindahan bit dari suatu tempat ke tempat lainnya, *presentation layer* memperhatikan syntax dan sematik informasi yang dikirimkan.

Encoding data adalah satu contoh layanan *presentation*.kebanyakan pengguna tidak memindahkan *string bit biner* yang *random* atau acak. Para pengguna data saling bertukar data seperti nama orang, tanggal, jumlah uang, dan

tagihan. Item-item tersebut dinyatakan dalam bentuk *string* karakter, bilangan *integer*, bilangan *floating point*, struktur data yang dibentuk dari beberapa item yang lebih sederhana. Terdapat perbedaan antara satu computer dengan computer lainnya dalam member kode untuk menyatakan *string* karakter(misalnya ASCII dan Unicode), integer (misalnya, komplemen satu dan komplemen dua), dan lain-lain. Untuk memungkinkan dua buah computer yang memiliki presentasi yang berbeda untuk dapat berkomunikasi, struktur data yang akan dipertukarkan dapat dinyatakan dengan cara abstrak, sesuai dengan encoding standard yang akan digunakan "pada saluran". *Presentation layer* mengatur data-struktur abstrak dan mengkonfersi dari *representation* yang digunakan pada sebuah computer menjadi *representation* standard jaringan, dan sebaliknya.

2.3.3.1.1.7 Lapisan Aplikasi

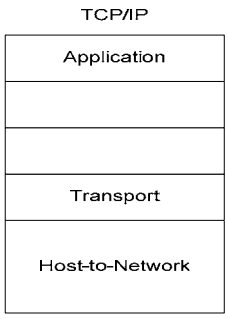
Lapisan aplikasi (*layer application)*terdiri dari bermacam-macam protokol. Misalnya terdapat ratusan jenis terminal yang tidak kompatibel di seluruh dunia. Ambil keadaan dimana editor layar penuh yang diharapkan bekerja pada jaringan dengan bermacam-macam terminal, yang masing-masing memiliki *layout* layar yang berlainan, mempunyai cara urutan penekanan tombol yang berbeda untuk penyisipan dan penghapusan teks, memindahkan sensor dan sebagainya.

Cara untuk mengatasi masalah di atas adalah, dengan menentukan terminal virtual jaringan abstrak, sehingga editor dan program-program lainnya dapat ditulis agar saling bersesuaian. Untuk menangani setiap jenis terminal, satu bagian *software* harus ditulis untuk memetakan fungsi terminal jaringan virtual ke terminal jaringan sebenarnya.

Fungsi lain dari lapisan ini adalah sebagai pemindah *file*. karena system file yang satu dan yang lainnya berbeda baik dalam hal konvesi nama yang berbeda, cara menyatakan baris-baris teks yang berbeda, dan sebagainya, maka diperlukan penanganan untuk mengatasi adanya ketidak cocokkan ini. Hal ini juga merupakan tugas dari lapisan aplikasi, seperti pada *e-mail*, *remote job entry*, *directory lookup*, dan sebagainya.

2.3.3.2 Model Referensi TCP/IP

Dibuatnya model referensi TCP/IP oleh DoD (*US Department of Defence*) ini bertujuan agar jaringan-jaringan yang telah ada mampu mempertahankan diri dari hilangnya perangkat keras *subnet*, dengan percakapan yang ada tidak terputus. Dengan kata lain, DoD mengharapkan hubungan tetap utuh selama mesin sumber dan mesin yang dituju tetap berfungsi, sekalipun sebagian mesin dan jalur transmisi yang menghubungkannya tiba-tiba terputus. Selain itu diperlukan arsitektur yang sangat fleksibel, karena aplikasi-aplikasi dengan persyaratan tertentu sangat diinginkan, mulai dari pemindahan data hingga transmisi pembicaraan *real time.* Gambar 2.3 dibawah ini adalah model TCP/IP.



Gambar 2.3 Model TCP/IP

2.3.3.2.1 Lapisan *Internet*

Lapisan *internet* (*internet layer*) ini merupakan simpul yang mengikat keseluruhan bentuk arsitektur secara bersama-sama. *Internet layer* bertugas untuk memungkinkan *host* mengirimkan paket ke jaringan dan memungkinkan paket-paket ini berjalan sendiri-sendiri ke tempat tujuannya (yang besar kemungkinan berada di jaringan lain). Paket-paket ini mungkin tiba di tujuan dengan urutan yang berbeda dari urutan sebelum dikirimkan. Dalam kasus seperti ini, layer-layer yang berada diatasnya bertugas untuk mengatur kembali, bila pengiriman terurut diinginkan.

Lapisan *internet* menentukan format paket yang resmi dan protokol resmi yang disebut *IP (Internet Protokol)*. Dalam hal ini, tugas i*nternet layer* dalah untuk mengirimkan paket-paket IP ke tempat tujuan seharusnya.

2.3.3.2.2 Lapisan Transport

Lapisan *transport* (*layer transport*) dirancang untuk memungkinkan *peer entity-peer entity* pada host sumber dan host tujuan untuk melakukan percakapan, seperti pada transport layer OSI. Disini telah ditentukan dua buah protokol *end-to-end*. Yang pertama, TCP (*Transmission Control Protocal*) merupaka protokol yang berorientasi pada hubungan yang andal (*reliable connection-oriented protokol*) yang mengijinkan sebuah aliran byte yang berasal pada suatu mesin untuk dikirkan tanpa *error* ke sebuah mesin yang ada di *internet.* TCP memecah aliran byte data menjadi pesan-pesan *diskret* dan meneruskannya ke *internet layer*. Pada mesin tujuan, proses TCP penerima merakit kembali pesan-pesan yang diterimanya menjadi aliran output. TCP juga menangani pengendalian aliran

untuk memastikan bahwa pengirim yang cepat tidak akan meembanjiri pesan-pesan yang akan diterima penerima yang lambat.

Protokol ke dua pada *layer* ini adalah UDP (*User Datagram Protokol*) yang merupakan protokol yang tidak andal (*unreliable*) dan tnpa sambungan (*connectionless*) bagi aplikasi-aplikasi yang tidak memerlukan pengurutan TCPau pengendalian aliran dan bagi aplikasi-aplikasi yang ingin melayani dirinya sendiri. UDP juga digunakan secara meluas pada *query* dan aplikasi *client-server* jenis *request-reply*, dimana pengirim yang cepat lebih diutamakan dengan pengiriman yang akurat.

2.3.3.2.3 Lapisan Aplikasi

Model TCP/IP tidak memiliki *session layer* dan *presentation layer*. Karena kedua layer tersebut dirasakan tidak diperlukan. Karena bagi kebanyakan aplikai, keduanya hanya mempunyai manfaat yang sedikit saja.

Application layer terdapat di puncak model TCP/IP. *Layer* ini berisi protokol-protokol tingkat tinggi. Protokol-protokol terdahulu terdiri dari terminal virtual (TELNET), *transfer file* (FTP), surat elektronik(SMTP). Protokol terminal virtual mengijinkan pengguna pada sebuah mesin untuk log ke mesin yang ada di tempat yang jauh dan bekerja di terminal jarak jauh itu. Protokol transfer file memungkinkan untuk mengirimkan data dari mesin yang satu ke mesin yang lain secara efisien. Surat elektronik pada awalnya merupakan salah satu jenis transfer file, akan tetapi kemudian dibuat protokol khusus untuk keperluan tersebut. Banyak jenis protokol yang telah ditambahkan ke layer ini, misalnya *Domain Name Service* (DNS) untuk memetakan nama-nama *host* ke alamat-alamat

jaringannya, NNTP yaitu protokol yang digunakan untuk memindahkan artikel-artikel berita (*newsgroup*), dan HTTP yaitu protokol yang berguna untuk mengambil halaman (*page*) di *World Wide Web*, dan banyak lagi protokol-protokol lainnya.

2.3.3.2.4 Lapisan *Host-to-Network*

Di bawah *internet layer* terdapat ruang kosong yang besar. Model TCP/IP tidak banyak menjelaskan tentang yang terjadi pada *layer* ini. Model TCP/IP hanya menyatakan bahwa *host* harus terhubung ke jaringan dengan menggunakan protokol, sehingga *host* dapat mengirim paket IP melalui *layer* ini. Protokol ini tidak tetap dan berbeda dari satu *host* ke *host* dan juga dari satu jaringan ke jaringan lainnya.

2.3.4 Media Transmisi

Kabel adalah alat penghubung untuk mengirim informasi dari satu computer ke computer yang lain. Ada beberapa macam tipe kabel yang umunya digunakan pada LAN. Dalam beberapa kasus, sebuah jaringan hanya menggunakan satu macam tipe kabel, sedangkan di jaringan lain menggunakan kabel yang berbeda. Kabel yang dipilih harus sesuai dengan topologi jaringan, protokol jaringan, dan ukurannya. Hal ini sangat penting untuk diketahui karena keberhasilan jaringa bergantung dari semua hal tersebut.

2.3.5 Macam-macam Tipe Kabel Pada Jaringan Local

2.3.5.1 Kabel *Twisted Pair*

Kabel *twisted pair* terdiri dari dua tipe yaitu *shielded twisted pair* (STP) dan *unshielded*. *Unshielded twisted pair* (UTP) adalah yang paling popular dan umumnya merupakan pilihan yang terbaik untuk jaringan sederhana.

Kualitas kabel UTP berbeda dengan kabel telepon, kabel UTP ini mempunyai empat pasang kabel di dalamnya. Setiap pasangan adalah jenis kabel kembar. Jenis konektor untuk kabel jenis ini adalah RJ-45.

Kabel STP mempunyai keunggulan seperti sangat baik digunakan pada jaringan yang menggunakan topologi *token ring.* Kekurangan kabel STP adalah, sangat sensitive terhadap sinyal radio dan listrik.

2.3.5.2 Kabel *Coaxial*

Kabel *coaxial* adalah kabel yang memiliki satu *Copper Conductor* dibagian tengahnya. Sebuah lapisan plastik menutupi diantara konduktor dan lapisan pengaman serat besi. Tersebut membantu menutupi gangguan dari arus listrik, lalu lintas kendaraan atau mesin, dan computer.

Selain sangat sulit untuk proses konfigurasi, kabel ini juga tidak tahan terhadap serangan dari sinyal-sinyal tertentu. Tetapi kelebihannya adalah mempunyai jangkauan yang lebih panjang dari pada kabel *twisted pair.*

Ada dua jenis tipe kabel ini yaitu kabel *thick coaxial* dan kabel *thin coaxial*. Kabel *thick coaxial* disebut juga dengan 10base2 (*thinnet*). Dimana angka 2 merujuk pada panjang maksimum untuk setiap segmen kabel tersebut, walaupun

pada kenyataanya hanya dapat menjangkau sampai 185 meter. Kabel ini sangat popular terutama pada penggunaan jaringan yang *linier*.

Kabel *thick coaxial* disebut juga dengan kabel 10base5 (*thicket*) dimana angka 5 merujuk untuk panjang maksimal dari setiap segmen kabel tersebut, yaitu 500 meter. Satu kekurangan kabel jenis ini adalah tidak lentur dan sangat relative sulit untuk mengkonfigurasinya. Tipe konektor untuk kabel jenis ini adalah konektor *bayone-Neill-Concelman* (BNC).

2.3.5.3 Kabel Serat Optik

Kabel serat optic (*fibre optic*) mempunyai kemampuan mentransmisi sinyal melalui jarak yang relative jauh dibandingkan kabel *coaxial* maupun kabel *twisted,* juga mempunyai kecepatan yang baik. Kabel ini sangat baik untuk digunakan sebagai fasilitas konferensi radio atau layanan interaktif. 10basF adalah merujuk ke spesifikasi untuk kabel serat optic dengan membawa sinyal *Ethernet.*

2.3.6 Tipe-Tipe Jaringan Komputer

2.3.6.1 LAN (*Local Area Network*)

local area network seringkali disebut LAN, merupakan jaringan milik pribadi di dalam sebuah gedung atau kampus yang berukuran sampai beberapa kilometer. LAN sering digunakan untuk menghubungkan computer-komputer pribadi dan *workstation* dalam kantor perusahaan atau pabrik-pabrik untuk pemakaian *resource* bersama dan saling bertukar informasi. LAN dapat dibedakan dari jenis jaringan lainnya berdasarkan tiga karakteristik : ukuran, teknologi transmisi, dan topologinya.

LAN mempunyai ukuran yang terbatas, yang berarti bahwa waktu transmisi pada keadaan terburuknya terbatas dan dapat diketahui sebelumnya. Dengan mengetahui keterbatasannya menyebabkan adanya kemungkinan untuk menggunakan jenis desain tertentu.

LAN tradisional beroprasi pada kecepatan mulai dari 10 sampai 100 mbps dengan delay rendah (puluhan mikrosekon) dan mempunyai factor kesalahan yang kecil. LAN modern dapat beroperasi pada kecepatan yang lebih tinggi, sampai ratusan mbps.

2.3.6.2 MAN (*Metropolitan Area Network)*

MAN merupakan suatu jaringa yang cakupannya meliputi suatu kota. MAN menghubungkan LAN-LAN yang lokasinya berjauhan. Jangkauan MAN bisa mencapai 10 km samapi beberapa ratus km, suatu MAN biasanya bekerja pada kecepatan 1,5 sampai 150 Mbps.

2.3.6.3 WAN(*Wide Area Network*)

WAN dirancang untuk menghubungkan computer-komputer yang terletak pada suatu cakupan geografis yang luas, seperti hubungan kota ke kota lain di dalam suatu negara. Cakupan WAN bisa mencapai 100km sampai 1000km, dan kecepatan antar kota bisa bervariasi antara 1,5 Mbps sampai 2,4Gbps. Dalam WAN, biaya untuk perawatan transmisi sangat tinggi, dan biasanya jaringan WAN dimiliki dan dioperasikan sebagai suatu jaringan publik.

2.3.6.4 GAN (*Global Area Network*)

Gan merupakan suatu jaringan yang menghubungkan Negara-negara di seluruh dunia. Kecepatan GAN bervariasi mulai dari 1,5 Mbps – 100Gbps dan cakupannya mencapai ribuan kilometer, contohnya *internet*.

2.3.7 Macam-Macam Topologi Jaringan

2.3.7.1 Garis Lurus

Topologi garis lurus (*linear bus*) terdiri dari satu jalur kabel utama dimana pada masing-masing ujungnya diberikan sebuah *terminator*. Semua *nodes* pada jaringan (*file server*, *workstation*, dan perangkat lainnya) terkoneksi pada sebuah kabel utama (*backbone*). Jaringan-jaringan *Ethernet* dan *Local Talk* menggunakan topologi *linear* ini. Gambar 2.4 dibawah adalah topologi *bus.*

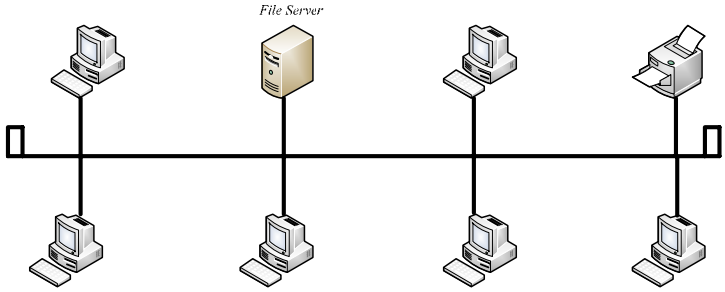
Kelebihan dari topologi *Linear Bus* adalah:

a. Mudah didalam mengkonfigurasi computer atau perangkat lain ke dalam sebuah kabel utama;
b. Tidak banyak menggunakan kabel dibandingkan dengan topologi *star*/bintang;

Kekurangan dari topologi *linear bus* adalah:

a. Seluruh jaringan akan mati jika terjadi kerusakan pada kabel utama(*backbone*);
b. Membutuhkan terminator pada kedua sisi dari kabel utamanya;
c. Sangat sulit mengidentifikasi permasalahan jika jaringa sedang jatuh atau rusak;

d. Sangat tidak disarankan dipakai sebagai salah satu solusi penggunaan jaringam di gedung besar;



Gambar 2.4 Topologi *Bus*

2.3.7.2 Bintang

Topologi bintang (*star*) dirancang, dimana setiap *nodes* (*file server*, *workstation,* dan perangkat lainnya) terkoneksi ke jaringan melewati sebuah *concentrator*.

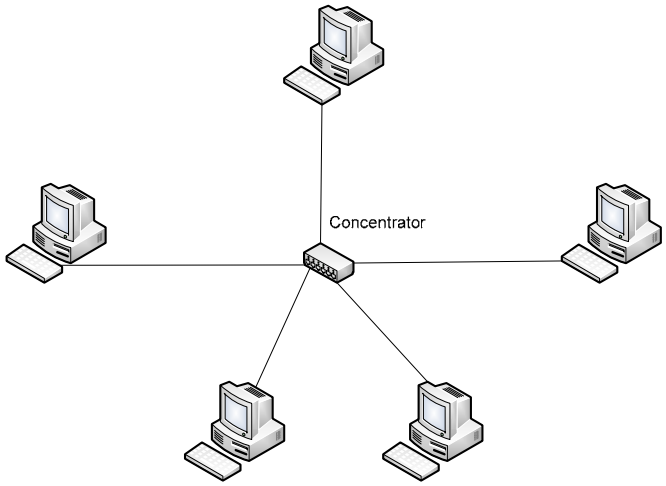
Data yang dikirim ke jaringan lokal akan melewati *concentrator* sebelum melanjutkan ke tempat tujuannya. *Concentrator* akan mengatur dan mengendalikan keseluruhan fungsi jaringan, dan juga bertindak sebagai *repeater* (penguat aliran data). Konfigurasi pada jaringan model ini menggunakan kabel *Twistead Pair*, dan dapat pula digunakan kabel *coaxial* atau kabel *fibre optic*. Gambar 2.5 dibawah adalah topologi *star*.

Kelebihan topologi *star*:

a. Mudah didalam pemasangan dan pengkabelan;
b. Tidak mengakibatkan gangguan pada jaringan ketika akan memasang atau memindahkan perangkat jaringan lainnya.

Kekurangan dari topologi *star*:

a. Membutuhkan lebih banyak kabel dari pada topologi *linear bus*;
b. Membutuhkan *concentrator*, dan bila mana *concentrator* rusak maka semua *node* yang terkoneksi tidak dapat terdeteksi;
c. Lebih mahal dari pada topologi *linear bus*, karena biaya untuk pengadaan *concentrator*.



Gambar 2.5 Topologi *Star*

2.3.7.3 Cincin

Topologi cincin (*ring*) menggunakan teknik konfigurasi yang sama pada topologi *star* tetapi pada topologi ini terlihat bahwa media transmisi menyerupai satu lingkaran tertutup menyerupai cincin, sehingga diberi nama topologi bintang dalam lingkaran atau *star-wired ring.*

2.3.7.4 Pohon

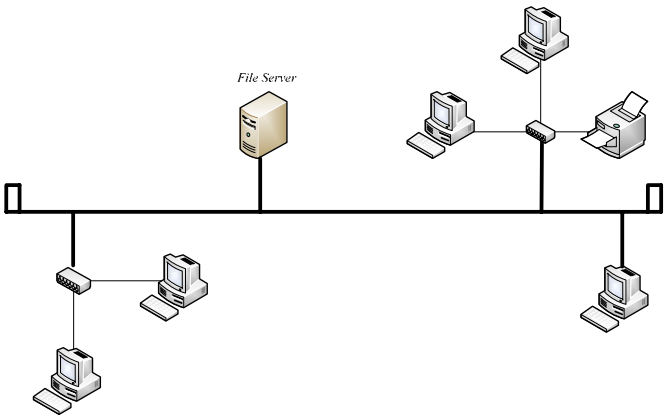
Topologi pohon (*tree*) merupakan perpaduan antara topologi *linera bus* dan *star*, yang mana terdiri dari kelompok-kelompok dari workstation dengan konfigurasi *star* yang terkoneksi ke kabel utama yang menggunakan topologi *linier bus*. Topologi ini memungkinkan untuk pengembangan jaringan yang telah ada, dan memungkinkan untuk mengkonfigurasi jaringan sesuai dengan kebutuhan. Gambar 2.6 dibawah ini adalah topologi pohon

Kelebihan topologi *tree:*

a. Proses konfigurasi jaringan dilakukan dari titik ke titik pada masing-masing segmen;
b. Didukung oleh banyak perangkat keras an perangkat lunak.

Kekurangan dari topologi *star*:

a. Keseluruhan panjang kabel pada tiap-tiap segmen dibatasi oleh tipe kabel yang digunakan;
b. Jika jaringan utama (*backbone*) rusak, maka keseluruhan segmen ikut rusak juga;
c. Sangat relative sulit untuk dikonfigurasi dan proses pengkabelannya dibandingkan topologi jaringan yang lain.



Gambar 2.6 Topologi *Tree*

2.3.8 Model Hubungan Pada LAN

2.3.8.1 Peer-to-Peer

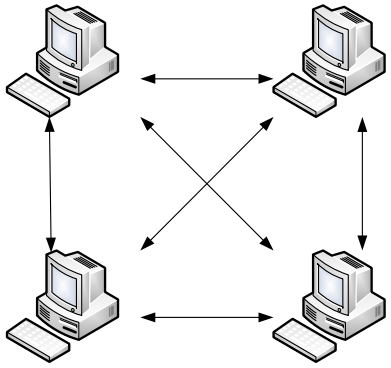
Model hubungan *peer to peer* memungkinkan u*user* membagi sumber daya yang ada pada komputernya baik itu berupa file, layanan p*printer* dan lain-lain serta mengakses sumber daya yang ada pada computer lain. Namun model ini tidak mempunyai sebuah *file server* atau sumber daya yang terpusat di dalam model hubungan *peer to peer* ini, seluruh computer adalah sama, yang mana mempunyai kemampuan yang sama untuk memakai sunber daya yang tersedia di dalam jaringan model ini di desain untuk berskala kecil dan menengah.

Kelebihan model hubungan *peer to peer*:

a. Tidak terlalu mahal, karena tidak membutuhkan *dedicated file server;*
b. Mudah dalam konfigurasi programnya, hanya tinggal mengatur untuk operasi model hubunga *peer to peer*.

Kekurangan model hubungan *peer to peer*:

a. Tidak terpusat, terutama untuk menyimpan data dan aplikasi;
b. Tidak aman, karena tidak menyediakan fasilitas untuk kepentingan itu.



Gambar 2.7 Model Hubungan *Peer to Peer*

2.3.8.2 Client/Server

Model hubungan *clien server* memungkinkan jaringan untuk mensentralisasi fungsi dan aplikasi kepada satu atau dua *dedicated file server*. Sebuah *file server* menjadi jantung bagi keseluruhan system, memungkinkan untuk mengakses sumber daya, dan menyediakan keamanan. *Workstation* yang berdiri sendiri dapat mengambil sumber daya yang ada pada *file server*. Model hubungan ini, menyediakan mekanisme untuk mengintegrasikan seluruh komponen yang ada di jaringan dan memungkinkan banyak pengguna secara bersama-sama memakai sumber daya pada *file server*.

Kelbihan model hubungan *client server*:

a. Terpusat (sumber daya dan keamanan data dikontrol melalui server);
b. Skalbilitas;

c. Fleksibel;
d. Teknologi baru dengan mudah terintegrasi kedalam system;
e. Keseluruhan komponen dapat bekerja bersama.

Kekurangan model hubungan *client server*:

a. Mahal;
b. Membutuhkan investasi untuk *dedicated file server*;
c. Biaya perbaikan mahal karena membutuhkan seorang staff untuk mengatur agar sistem berjalan secara efisien;
d. Berketergantungan;
e. Ketika server jatuh menyebabkn keseluruhan operasi pada jaringan akan jatuh juga.

2.3.9 Macam Perangkat Keras Pada LAN

Perangkat keras yang dibutuhkan yaitu computer, *network interface card*, dan segala sesuatu yang berhubungan dengan koneksi jaringan seperti: *bridges*, *router*, dan perangkat lain yang dibutuhkan untuk proses transformasi data dalam jaringan.

2.3.9.1 File Servers

Sebuah *file* server merupakan pusat dari suatu jaringan, merupakan computer yang memiliki spesifikasi relative tinggi, mempunyai memori yang besar, media penyimpanan yang besar, dengan kartu jaringan yang memiliki kecepatan tinggi. System operasi jaringan tersimpan di sini, juga termasuk didalamnya beberapa aplikasi dan data yang dibutuhkan untuk jaringan.

Sebuah *file server* bertugas mengontrol komunikasi dan pertukaran informasi diantara *nodes* dalam satu jaringan.

2.3.9.2 Workstations

Keseluruhan computer yang terhubung ke *file server* dalam jaringan disebut sebagai *workstation.* Sebuah *workstation* minimal mempunyai kartu jaringan, aplikasi jaringan, kabel untuk menghubungkanke jaringan, biasanya tidak terlalu membtuhkan sebuah *floppy disk*, karena data yang akan disimpan bisa diletakkan di *file server.*

2.3.9.3 Network Interface Card

Merupakan perangkat yang menyediakan media untuk menghubungkan computer. Kebanyakan kartu jaringan adalah *internal card*, yaitu kartu jaringan yang di pasang pada slot ekspansi dalam computer.

Kartu jaringan yang banyak terpakai saat ini adalah: kartu jaringan *Ethernet, Local Talk Connector,* dan kartu jaringan *Token Ring*.

2.3.9.4 Repetears

Merupakan perangkat yang berguna untuk menguatkan sinyal. Sebagai contoh, pada sebuah LAN menggunakan topologi *star* dengan menggunakan kabel UTP. Dimana panjangn maksimal kabel UTP adalah 100 meter, maka dipasang sebuah *repeater* pada jaringan tersebut untuk menguatkan sinyal.

2.3.9.5 Bridges

Adalah perangkat yang membagi sebuah jaringan ke dalam dua buah jaringan, hal ini digunakan untuk mendapatkan jaringan yang efisien. Kebanyakan *bridges* dapat mengetahui masing-masing alamat dari tuap-tiap segmen computer

pada jaringan sebelahnya dan juga pada jaringan yang lain di sebelahnya pula. *Bridges* juga dapat digunakan untuk mengkoneksi diantara network yang menggunakan tipe kabel yang berbeda ataupun topologi yang berbeda pula, namun harus dalam protokol yang sama.

2.3.9.6 Router

Hampir sama dengan *bridge*, tetapi mempunyai kemampuan lebih yaitu, dapat mencari jalur yang terbaik untuk mengirimkan sebuah pesan yang berdasarkan atas alamat tujuan dan alamat asal.

2.3.10 Kelas IP *Address*

2.3.10.1 Network ID dan Host ID

Pembagian kelas IP *address* didasarkan pada dua hal: *Network* ID dan *Host* ID dari suatu IP *address*. Setiap IP *address* selalu merupakan sebuah pasangan dari *network-ID* (identitas jaringan) dan *host* ID (Identitas Host dalam jaringan tersebut). *Network-ID* ialah bagian dari IP *address* yang digunakan untuk menunjukkan *workstation*, server, router, dan semua *host* TCP/IP lainnya dalam jaringan tersebut. Dalam satu jaringan, *hoset-ID* ini harus unik.

2.3.10.1.1 Kelas A

Karakteristik

Format	: 0nnnnnnn.hhhhhhhh.hhhhhhhh.hhhhhhhh
Bit Pertama	: 0
Panjang NET ID	: 8 bit
Panjang Host ID	: 24 bit
Byte Pertama	: 0-127

Jumlah : 126 Kelas A (0 dan 127 dicadangkan)

Range IP : 1.xxx.xxx.xxx sampai 126.xxx.xxx.xxx

Jumlah IP : 16.777.214 IP *address* pada tiap kelas A

IP *address* kelas A diberikan untuk jaringan dengan jumlah *host* yang sangat besar. Bit pertama dari IP *address* kelas A selalu di *set* 0 sehingga byte terdepan dari IP kelas A selalu bernilai antara angka 0 dan 127. Pada IP *address* kelas A, *network* ID ialah delapan *bit* pertama, sedangkan *host* ID adalah 24 *bit* berikutnya. Cara membaca IP kelas A, misalnya 113.46.5.6 ialah:

NetworkID = 113

HostID = 46.5.6

Sehingga IP *address* diatas berarti host nomor 46.5.6 pada *network* nomor 113. Dengan panjang *host* ID yang 24nit, *network* dengan IP *address* kelas A ini dapat menampung 16 juta *host.*

2.3.10.1.2 Kelas B

Karakteristik

Format : 10nnnnnn.hhhhhhhh.hhhhhhhh.hhhhhhhh

Bit Pertama : 10

Panjang NET ID : 16 bit

Panjang Host ID : 16 bit

Byte Pertama : 128-191

Jumlah : 16.184 kkelas B

Range IP : 128.0.xxx.xxx sampai 191.255.xxx.xxx

Jumlah IP : 65.532 IP *address* pada tiap kelas B

Ip *address* kelas B biasanya dialokasikan untuk jaringan berukuran sedang dan besar. Dua bit pertama dari IP address kelas B selalu di set 10 sehingga byte terdepan dari IP address kelas B selalu bernilai antara 128 sampai 191. Pada IP address kelas B, network ID ialah enam belas bit pertama. Cara membaca IP address kelas B, misalnya 132.92.121.1 adalah:

Network ID = 132.92

HostId = 121.1

Sehingga IP address di atas berarti host nomor 121.1 pada network nomor 132.92 dengan panjang host ID yang 16 bit, network dengan IP address kelas B ini dapat menampung sekitar 65000 host.

2.3.10.1.3 Kelas C

Karakteristik

Format : 110nnnnn.hhhhhhhh.hhhhhhhh.hhhhhhhh

Bit Pertama : 110

Panjang NET ID : 24 bit

Panjang Host ID : 8 bit

Byte Pertama : 192-223

Jumlah : 2.097.152 kelas C

Range IP : 192.0.0.xxx sampai 223.255.255.xxx

Jumlah IP : 254 IP *address* pada tiap kelas C

2.3.10.1.4 Kelas D

Karakteristik

Format : 1110mmmm.mmmmmmmm. mmmmmmmm. mmmmmmmm

4 Bit Pertama : 1110

Nit Multicast : 28 bit

Byte inisial : 224-239

Byte Pertama : 0-127

Deskripsi : Kelas D adalah ruang alamat multicast

IP address kelas D digunakan untuk keperluan IP multicasting. 4 bit pertama IP address kelas D di set 1110. Bit-bit berikutnya diatur sesuai keperluan multicast group yang menggunakan IP address ini. Dalam multicasting tidak dikenal network bit dan host bit.

2.3.10.1.5 Kelas E

Karektiristik:

Format : 1111rrrr.rrrrrrrr.rrrrrrrr.rrrrrrrr

Format :1111

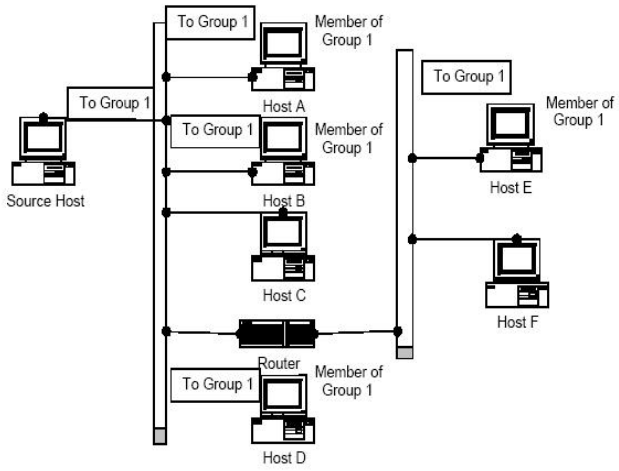
4 bit pertama : 28 bit

Byte inisial : 248-255

Kelas E adalah ruang alamat yang dicadangkan untuk keperluan eksperimental. IP address kelas E tidak digunakan untuk umum. $ bit pertama IP address ini di set 1111.

2.3.11 *Multicast*

Multicast adalah komunikasi *one-to-many, yaitu* antara satu komputer pengirim dan beberapa komputer penerima di suatu jaringan komputer. Komunikasinya berupa satu arus (sepaket) data yang ditransmisikan secara terus menerus ke banyak *host* tertentu yang telah berada pada grup *multicast*. *Multicast* dapat tercapai dengan cara mencantumkan satu *multicast address* sebagai tujuan dari datagram yang dikirim. Jadi *multicast* tidak dipakai untuk alamat satu *host* saja, melainkan ditunjukkan untuk mengalamatkan sejumlah *host* yang bergabung dalam suatu grup yang menjalankan aplikasi yang sama. Gambar 2.8 dibawah ini adalah proses multicasting.



Gambar 2.8 Pengiriman Datagram ke Suatu *Multicast Group*

Pada gambar di atas, sejumlah *host* melakukan komunikasi *multipoint* untuk menjalankan suatu aplikasi bersama. *Host* yang terlibat dalam komunikasi *multipoint* ini sebagian ada pada *network* 1, sebagian lagi pada *network* 2. Antara *network* 1 dan *network* 2 dihubungkan melalui *router*. Salah satu *host* (pada gambar disebut *source host*) mengirimkan datagram ke *suatu multicast address*

(misalkan 224.22.33.44). Untuk lebih memudahkan, *multicast address* ini kita identikan dengan grup 1, karena ada kemungkinan penggunaan *multicast address* lain sebagai grup 2, grup 3 , dan seterusnya. *Source host* ini hanya mengirimkan 1 datagram ke jaringan.

Pada *network* 1 yang menggunakan shared media, seluruh *host* sebenarnya mendengar datagram ini. Khusus bagi *host-host* yang terlibat dan menyatakan dirinya sebagai grup 1 (memiliki *multicast address* 224.22.33.44) , datagram akan diproses lebih lanjut oleh lapisan di atas IP, sementara bagi *host* yang tidak terlibat (*host* C), datagram akan diabaikan sebagaimana datagram lain yang memiliki *address* tujuan bukan kepadanya.

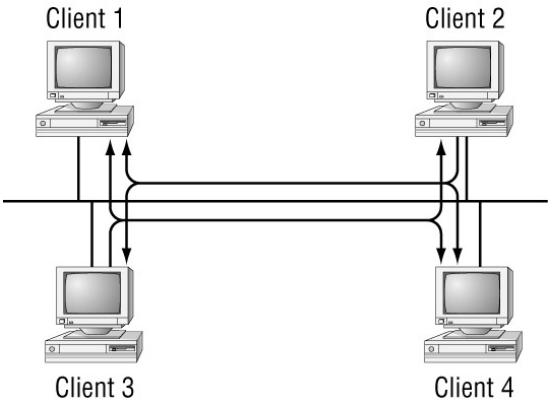
Dengan demikian maka beban kerja host pengirim cukup ringan, karena tidak perlu melakukan replikasi datagram. Serta kebutuhan *bandwidth* untuk transisi datagram tidak bergantung pada jumlah *host* yang terlibat.

2.3.11.1 Teknik *Multicast*

Ada dua teknik yang digunakan untuk mengontrol sesi *multicast*, yaitu:

Teknik *Peer to Peer*

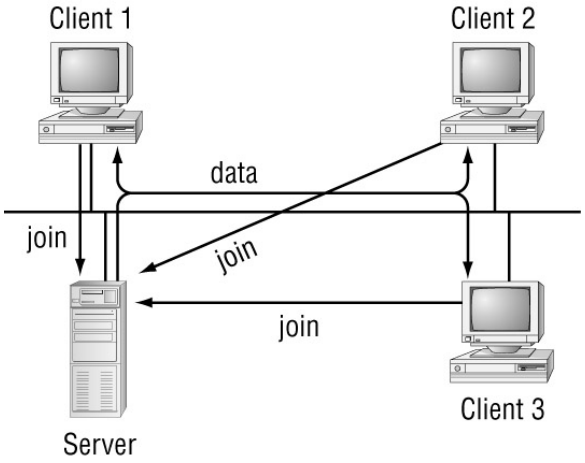
Teknik *peer-to-peer*, dimana semua client dapat mengirim pesan ke semua client lainnya yang ada didalam suatu grup. Dalam suatu grup multicast *peer-to-peer*, semua client dalam grup multicast mempunyai hak yang sama. Setiap client dalam grup dapat bertukar pesan dengan setiap client lain dalam grup. System IP mendukung grup multicast peer-to-peer dengan mengijinkan setiap alat di jaringan untuk menerima dan mengirim paket-paket yang diperuntukkan untuk IP address grup multicast. Gambar 2.9 dibawah ini adalah teknik *peer to peer* pada *multicast*



Gambar 2.9 Teknik *Peer to Peer Pada Multicast*

Teknik *Client-Server*

Teknik client-server, dimana sebuah server mengrimkan pesan ke grup client. Sistem multicast yang lain mempekerjakan sebuah server, alat yang tunggal di jaringan yang mengontrol semua kegiatan grup multicast. Sebuah client yang ingin bergabung dengan grup multicast harus meminta ijin dari server. Jika server menolak akses client ke grup multicast, maka tidak ada paket-paket yang akan disampaikan ke permintaan client tersebut. Gambar 2.10 dibawah ini merupakan teknik *client-server* pada *multicast*

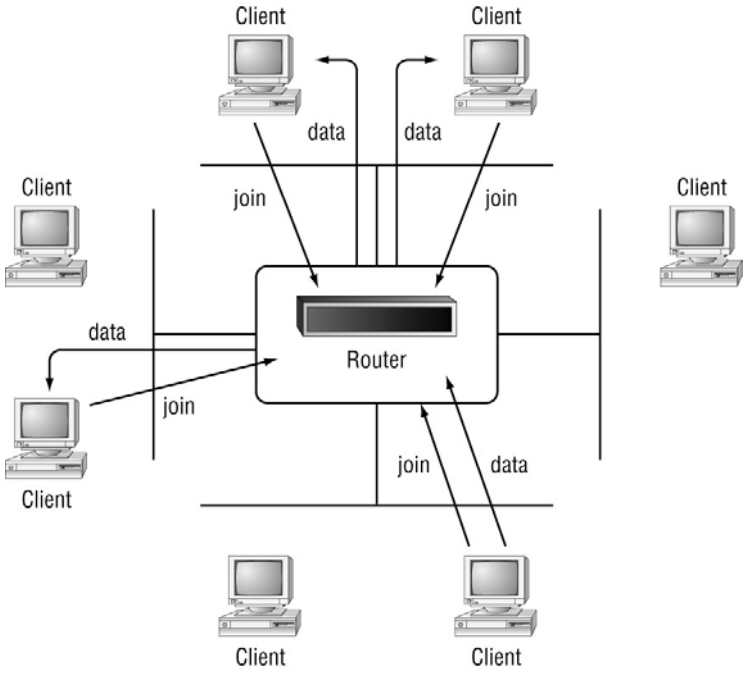


Gambar 2.10 Teknik *Client-Server* Pada *Multicast*

Mengirim Paket-Paket Multicast melalui Router

Walaupun paket-paket multicast dapat dilewati ke seberang batasan-batasan jaringan, memerlukan beberapa usaha pada bagian dari router jaringan. Kebanyakan router tidak dapat melewatkan paket multicast sampai ke subnet-subnet yang lain. Jika sebuah router dilewati tiap-tiap paket multicast yang diterima ke tiap-tiap interface, maka akan menaruh jaringan berhadapan dengan resiko dibanjiri dengan paket multicast. Sebagai gantinya, suatu sistem telah dikembangkan untuk mengijinkan penyampaian yang selektif tentang paket multicast.

Internet Group Management Protocol (IGMP) dikembangkan untuk membantu dalam memberitahu router ketika paket multicast perlu diberikan kepada berbagai subnet. Ketika suatu alat jaringan ingin bergabung dengan suatu grup multicast, ia akan mengirim suatu paket IGMP kepada router lokal di subnetnya. Paket IGMP mendaftarkan alat jaringan dan alamat grup multicast darimana alat itu harus menerima pesan. Ini membuka peluang router untuk mengetahui bahwa ia harus meneruskan pesan multicast manapun yang diterima untuk grup itu kepada subnet dari alat jaringan yang spesifik.



Gambar 2.11 Pengiriman Paket Melalui Router Menggunakan Metode *Multicast*

Seperti pada gambar 2.11, ketika masing-masing alat jaringan mendaftarkan untuk menerima paket multicast untuk grup tersebut, router harus meneruskan paket multicast manapun yang diterima ke masing-masing interface yang berisi sebuah host yang terdaftar. Interface-interface yang tidak berisi host yang terdaftar tidak harus menerima paket multicast.

Dengan cara yang sama, ketika sebuah host jaringan meninggalkan grup multicast, paket IGMP yang lain dikirim ke router, yang memberitahukan bahwa host tidak lagi harus mempunyai paket untuk diteruskan ke grup multicast. Ketika host jaringan yang terakhir di suatu interface router yang spesifik meninggalkan grup, router dapat berhenti meneruskan paket multicast ke interface tersebut.

2.3.11.2 Kelebihan *Multicast*

Kesimpulannya, penggunaan metode komunikasi *multicast* ini memiliki beberapa keunggulan, yaitu:

1. Beban kerja *host* pengirim cukup ringan, karena tidak perlu melakukan replikasi datagram.
2. Kebutuhan *bandwith* untuk transisi datagram tidak bergantung kepada jumlah *host* yang terlibat. Satu atau banyak *host* yang terlibat pada satu jaringan, *bandwith* yang dibutuhkan tetap sama. Demikian juga jika pada *network* 2 terdapat puluhan *host* sebagai anggota grup, *router* hanya perlu meneruskan satu datagram saja untuk mencapai seluruh *host* tersebut.

2.4 Video Konferensi Multicast

Video konferensi adalah suatu kombinasi dari transmisi video dan audio dimana memungkinkan dua orang atau lebih di lokasi yang berbeda untuk melihat dan mendengar satu sama lain seperti jika melakukan percakapan tatap muka. Sebuah *webcam* digunakan untuk menangkap *signal* video dan sebuah *mic* untuk menangkap *signal* audio. Kemudian data akan dikirimkan ke jaringan menggunakan metoda multicast ke grup multicast sehingga dapat diterima oleh semua komputer yang terdapat dalam grup multicast yang sama. Penggunaan metoda multicast ini dapat meminimalisasi penggunaan *bandwith,* karena hanya satu paket (video dan audio) data yang dikirimkan ke n-komputer yang bergabung pada grup multicast, sehingga tidak terjadi proses yang mengakibatkan penduplikasian paket data.

2.5 Rekayasa Sistem Berorientasi Objek

Rekayasa adalah bentuk upaya manusia untuk mencari atau merancang produk yang berkualitas. Produk dapat berupa sistem atau perangkat lunak yang sedang atau akan dibangun dengan suatu model pengembangan. Kegiatan rekayasa sistem dapat dilakukan dengan beberapa cara, misalnya melalui studi literatur tentang masalah yang diteliti antara lain: teknologi *multicasting*, *streaming*, jaringan komputer, dan lain-lain.

2.5.1 Pemrograman Berorientasi Objek

Orientasi objek adalah cara pandang bukan sekedar algoritma yang diimplementasikan pada bahasa pemrograman berorientasi objek. Cara berpikir orientasi objek adalah segala sesuatu dipandang sebagai objek. Objek dapat berupa konsep, abstraksi atau sesuatu dengan batas-batas tegas dan mempunyai arti untuk persoalan yang ditangani. Objek mempunyai identitas dan dapat dibedakan. Ada dua kegunaan objek, yaitu untuk meningkatkan pemahaman dunia nyata dan menyediakan dasar pengetahuan praktis untuk implementasi komputer.

Istilah ”objek” digunakan pada dua disiplin yang berbeda, yaitu:

1. Pada pemodelan informasi, objek menggambarkan suatu benda pada dunia nyata, misalnya objek file Avi.
2. Pada bahasa pemrograman berorientasi objek, objek adalah variabel dengan tipe data bentukan yang memiliki nilai dan operasi (prosedur/fungsi) tertentu, yaitu ”kelas”. Dengan demikian objek adalah instans dari kelas pada saat program dijalankan.

2.5.1.1 Karakteristik Utama Pemrograman Berorientasi Objek

Karateristik dari pemrograman berorientasi objek, yaitu: abstraksi, enkapsulasi, pewarisan (*inheritance*), *reuseabilitiy*, spesialisasi, generalisasi, komunikasi antar objek, dan *polymorphisme*.

2.5.1.2 Tujuan Pemrograman Berorientasi Objek

Tujuan utama dari pemrograman berorientasi objek adalah pengkapsulasian data dan fungsi yang mengakses data di dalam kelas. Keuntungannya, yaitu:

1. Memudahkan dalam perawatan program.
2. Memudahkan dalam pencarian kesalahan.

2.5.2 UML (*Unified Modeling Language*)

UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (*Object-Oriented*). UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem *software*.

UML sebagai sebuah bahasa yang memberikan *vocabulary* dan tatanan penulisan kata-kata dalam *'MS Word'* untuk kegunaan komunikasi. Sebuah bahasa model adalah sebuah bahasa yang mempunyai *vocabulary* dan konsep

tatanan/aturan penulisan serta secara fisik mempresentasikan dari sebuah sistem. Seperti halnya UML adalah sebuah bahasa *standard* untuk pengembangan sebuah *software* yang dapat menyampaikan bagaimana membuat dan membentuk model-model, tetapi tidak menyampaikan apa dan kapan model yang seharusnya dibuat yang merupakan salah satu proses implementasi pengembangan *software*.

Untuk dapat memahami UML membutuhkan bentuk konsep dari sebuah bahasa model, dan mempelajari tiga elemen utama dari UML seperti *building block*, aturan-aturan yang menyatakan bagaimana *building block* diletakkan secara bersamaan, dan beberapa mekanisme umum (common).

Tiga macam yang terdapat dalam building block adalah benda/*things*, hubungan, dan diagram.

2.5.2.1 Benda/*Things*

Adalah hal yang sangat mendasar dalam model UML, juga merupakan bagian paling statik dari sebuah model, serta menjelaskan elemen-elemen lainnya dari sebuah konsep dan atau fisik. Bentuk dari beberapa benda/*thing* adalah sebagai berikut:

1. **Kelas (*Class*)**

   Sekelompok objek yang mempunyai atribut, operasi, dan hubungan yang semantik. Sebuah kelas mengimplementasikan 1 atau lebih antarmuka. Sebuah kelas dapat digambarkan sebagai sebuah persegi panjang, yang mempunyai sebuah nama, atribut, dan metode pengoperasiannya.

2. Antarmuka (*Interface*)

Sebuah antarmuka yang menghubungkan dan melayani antar kelas dan atau elemen. Antarmuka mendefinisikan sebuah set/kelompok dari spesifikasi pengoperasian, umumnya digambarkan dengan sebuah lingkaran yang disertai dengan namanya. Sebuah antarmuka berdiri sendiri dan umumnya merupakan pelengkap dari kelas atau komponen.

3. *Collaboration*

Interaksi dan sebuah kumpulan / kelompok dari kelas-kelas/elemen-elemen yang bekerja secara bersama-sama. C*ollaborations* mempunyai struktur dan dimensi. Pemberian sebuah kelas memungkinkan berpartisipasi didalam beberapa *collaborations* dan digambarkan dengan sebuah '*elips*' dengan garis terpotong-potong.

4. *Use Case*

Rangkaian/uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. *Use case* digunakan untuk membentuk tingkah laku benda/*things* dalam sebuah model serta direalisasikan oleh sebuah *collaboration*. Umumnya *use case* digambarkan dengan sebuah *elips* dengan garis yang solid, biasanya mengandung nama.

5. *Node*

Fisik dari elemen-elemen yang ada pada saat dijalankannya sebuah sistem, contohnya adalah sebuah komputer, umumnya mempunyai sedikitnya *memory* dan *processor*. Sekelompok komponen mungkin terletak pada sebuah

node dan juga mungkin akan berpindah dari *node* satu ke *node* lainnya. Umumnya *node* ini digambarkan seperti kubus serta hanya mengandung namanya.

2.5.2.2 Hubungan (*Relationship*)

Ada empat macam hubungan didalam penggunaan UML, yaitu: *dependency, association, generalization,* dan *realization.*

1. ***Dependency***

   Hubungan semantik antara dua benda/*things* yang mana sebuah benda berubah mengakibatkan benda satunya akan berubah pula. Umumnya sebuah *dependency* digambarkan sebuah panah dengan garis terputus-putus.

2. ***Association***

   Hubungan antar benda struktural yang terhubung diantara objek. Kesatuan objek yang terhubung merupakan hubungan khusus, yang menggambarkan sebuah hubungan struktural diantara seluruh atau sebagian. Umumnya *association* digambarkan dengan sebuah garis yang dilengkapi dengan sebuah label, nama, dan status hubungannya.

3. ***Generalization***

   Hubungan khusus dalam objek anak yang menggantikan objek induk. Dalam hal ini, objek anak memberikan pengaruhnya dalam hal struktur dan tingkah lakunya kepada objek induk.

4. ***Realization***

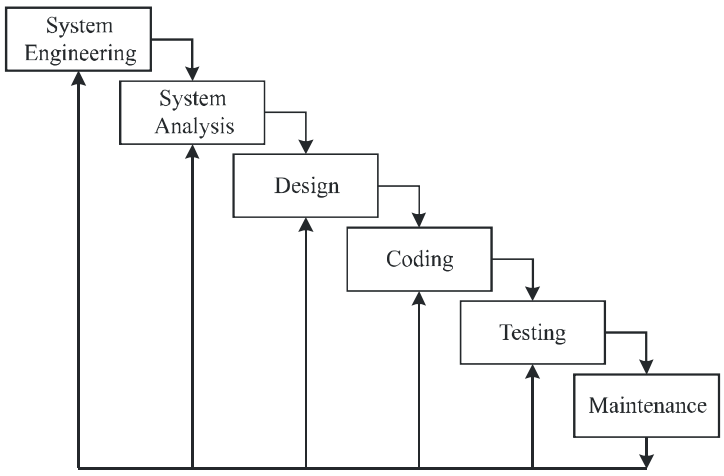
Hubungan semantik antara pengelompokkan yang menjamin adanya ikatan diantaranya. Hubungan ini dapat diwujudkan diantara *interface* dan kelas atau *elements*, serta antara *use cases* dan *collaborations*.

2.5.2.3 Diagram

UML sendiri terdiri atas pengelompokkan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. UML mempunyai sembilan diagram, yaitu: *use-case*, *class*, *object*, *sequence*, *collaboration*, *activity*, *state*, *component*, dan *deployment diagram*.

2.5.3 Metode *Waterfall*

Metode *waterfall* mempunyai tahapan-tahapan sebagai berikut:

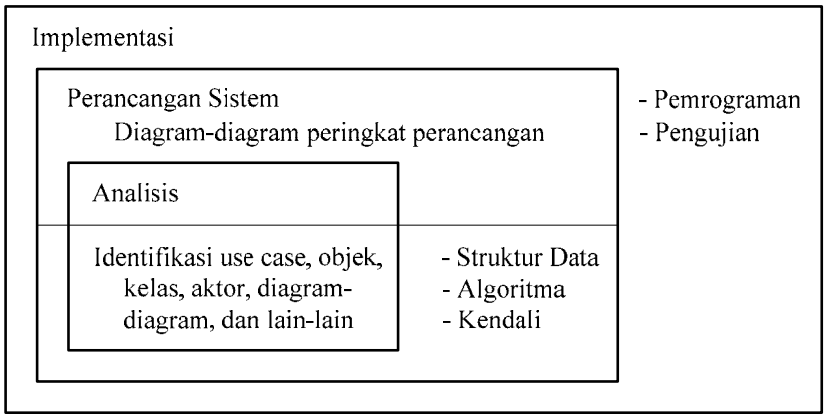


Gambar 2.12. *Waterfall* (*Clasic Life Cycle*)

Keterangan :

1. ***System Engineering***, melakukan pengumpulan data dan penetapan kebutuhan semua elemen sistem.
2. ***System Analysis***, melakukan analisis terhadap permasalahan yang dihadapi dan menetapkan kebutuhan perangkat lunak, fungsi performansi dan interfacing.
3. ***Design***, menetapkan domain informasi untuk perangkat lunak, fungsi dan interfacing.
4. ***Coding***, pengkodean yang mengimplementasikan hasil disain kedalam kode atau bahasa yang dimengerti oleh mesin komputer dengan menggunakan bahasa pemrograman tertentu.
5. ***Testing***, melakukan pengujian kebenaran lojik dan fungsionalitas. Disinilah akan diketahui kekurangan-kekurangan yang terjadi pada program.
6. ***Maintenance,*** menangani perangkat lunak yang sudah selesai supaya dapat berjalan dan terhindar dari gangguan-gangguan yang menyebabkan kerusakan.

Model *waterfall* mengandung 3 langkah utama, yaitu: analisis, perancangan, dan implementasi, sehingga dalam kaitannya dengan pengembangan sistem berorientasi objek, siklus tersebut dapat dilukiskan kembali pada Gambar 2.13.

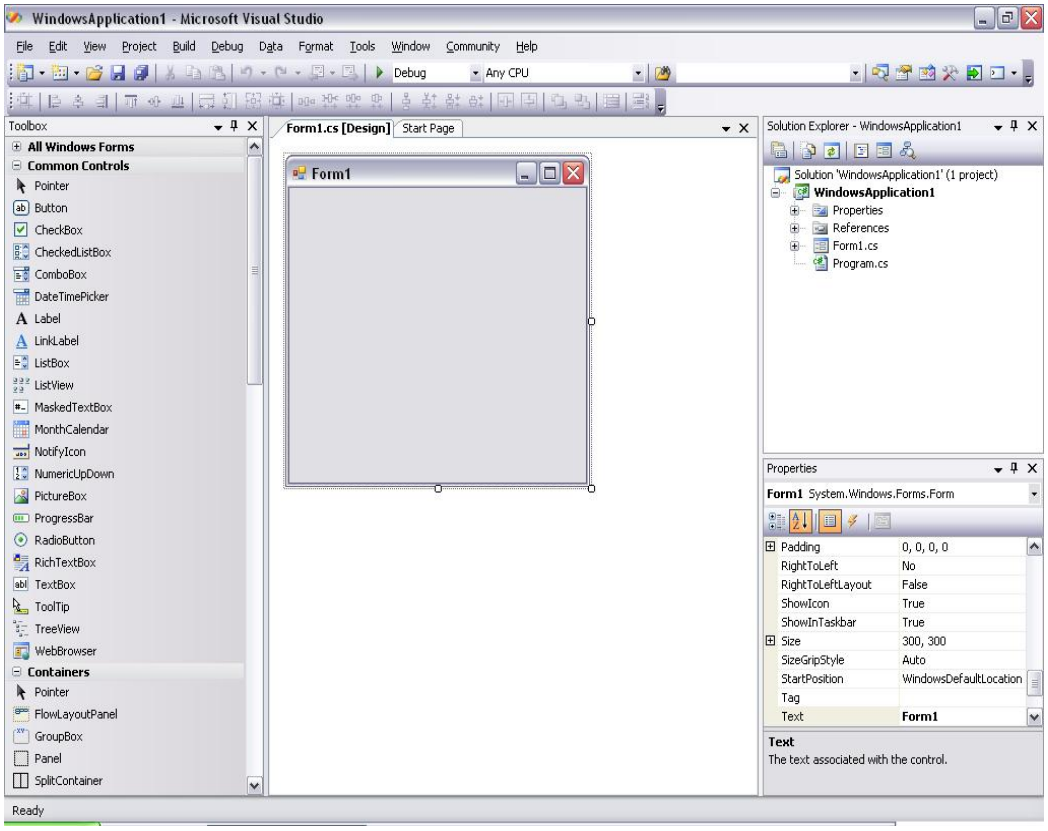


Gambar 2.13. SDLC Berorientasi Objek

2.6 Visual C Sharp (*C#)* 2005

C Sharp (*C#*) merupakan bahasa pemrograman perusahaan Microsoft yang didesain dengan target diimplementasikan dalam teknologi Framework .Net. Bahasa pemrograman C# merupakan bahasa baru yang sangat andal dan konsisten serta membawa kesan bahasa pemrograman yang modern. C# pada dasarnya bukan bahasa sulit karena intinya mengambil dari bahasa pemrograman yang telah ada. Bahasa yang paling banyak digunakan dalam sintaksis C# adalah bahasa Java, yaitu sekitar 70%.

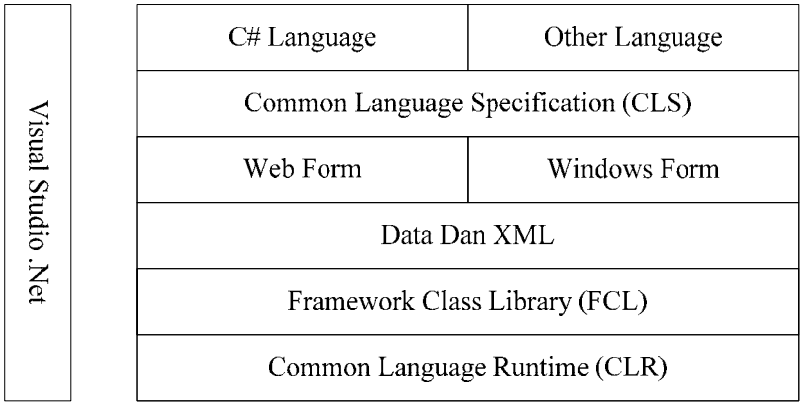
Visual C# 2005 merupakan bagian teknologi .Net yang dikembangkan untuk memenuhi kebutuhan industri akan keberadaan teknologi IT modern, yang mampu mengatasi sistem informasi. Gambar 2.14 dibawah ini adalah tampilan program *Visual C#2005*



Gambar 2.14. Program Visual C# 2005

2.5.1 Arsitektur Visual .Net

Arsitektur Visual C# 2005 hampir sama dengan Visual Studio .Net. Pada teknologi sekarang, setiap CLS (*Common Language Specification*) memiliki IDE (*Integrated Development Enviroment*) atau lingkungan kerja masing-masing. Gambar di bawah adalah arsitektur visual C# 2005



Gambar 2.15. Arsitektur Visual C# .Net

2.5.2 Keuntungan Visual C# 2005

Ada beberapa keuntungan menggunakan teknologi .Net dalam membangun sebuah aplikasi, baik aplikasi web service maupun aplikasi biasa yang sering digunakan seperti Windows Form, diantaranya:

1. Mudah, Visual C# yang sangat sederhana mudah digunakan dalam mengimplementasikan pengembangan sebuah aplikasi.
2. Efisien, kemudahan saat pembuatan aplikasi sangat berpengaruh pada efisiensi waktu pengerjaan aplikasi dan berdampak pula pada efisiensi biaya.
3. Produktivitas, kemudahan pengerjaan aplikasi akan berdampak besar dalam menghasilkan produk yang berupa aplikasi.
4. Konsisten, kemudahan yang dihadirkan oleh Visual C# akan berdampak pada konsisten program.