BAB II

LANDASAN TEORI

2.1 Jaringan Komputer

Jaringan komputer adalah sekelompok komputer otonom yang dihubungkan satu dengan yang lainnya dengan menggunakan protokol komunikasi melalui media transmisi atau media komunikasi sehingga dapat saling berbagi data informasi, program–program, *share* perangkat keras seperti *printer* atau *harddisk*. Tujuan dibangunnya suatu jaringan komputer adalah untuk membawa data informasi dari sisi pengirim menuju penerima secara cepat dan tepat tanpa adanya kesalahan melalui media transmisi atau media komunikasi tertentu.

2.1.1 Jenis Jaringan Komputer

Jaringan komputer yang digunakan dewasa ini dapat dikelompokkan berdasarkan luasan area yang dapat dijangkau. Luasan area ini pada mulanya ditentukan berdasarkan jarak jangkau dalam satuan meter. Namun dalam perkembangan selanjutnya berdasarkan peralatan dan fasilitas. Penggunaan peralatan–peralatan yang dimaksud adalah peralatan–peralatan *inter* jaringan seperti *Repeater, Router* atau *Gateway* dari peralatan tersebut dapat dilihat jangkauan area jaringan dan luasan segmen jaringan yang dibangun. Berdasarkan kriteria–kriteria tersebut secara umum jaringan komputer dapat digolongkan sebagai berikut :

2.1.1.1 *Local Area Network* (LAN)

Sebuah LAN adalah jaringan yang dibatasi oleh area yang relatif kecil, umumnya dibatasi oleh area lingkungan seperti sebuah perkantoran di sebuah gedung, atau sebuah sekolah, dan biasanya sekitar 1 km persegi.. Komputer-komputer yang terhubung ke dalam jaringan itu biasanya disebut dengan *workstation.* Kebanyakan LAN menggunakan media kabel untuk menghubungkan antara satu komputer dengan komputer lainnya.

2.1.1.2 *Metropolitan Area Network* (MAN)

Sebuah MAN, biasanya meliputi area yang lebih besar dari LAN, misalnya antar wilayah dalam satu provinsi. Dalam hal ini jaringan menghubungkan beberapa buah jaringan-jaringan kecil ke dalam lingkungan area yang lebih besar, sebagai contoh yaitu : jaringan Bank dimana beberapa kantor cabang sebuah Bank di dalam sebuah kota besar dihubungkan antara satu dengan lainnya.

2.1.1.3 *Wide Area Network* (WAN)

Wide Area Networks (WAN) adalah jaringan yang lingkupnya biasanya sudah menggunakan sarana satelit ataupun kabel bawah laut sebagai contoh keseluruhan jaringan suatu Bank yang ada di Indonesia ataupun yang ada di negara - negara lain. menggunakan sarana WAN, Sebuah Bank yang ada di Bandung bisa menghubungi kantor cabangnya yang ada di Jepang, hanya dalam beberapa menit. Biasanya WAN agak rumit dan sangat kompleks, menggunakan banyak sarana untuk menghubungkan antara LAN dan WAN ke dalam komunikasi global seperti Internet. Tapi bagaimanapun juga antara LAN, MAN

dan WAN tidak banyak berbeda dalam beberapa hal, hanya lingkup areanya saja yang berbeda satu diantara yang lainnya.

2.1.1.4 Internet

Internet merupakan jaringan Global Internasional di seluruh dunia. Biasanya teknologi ini menggunakan suatu fasilitas layanan yang biasa kita sebut *World Wide Web* (WWW).

2.1.2 Tipe Jaringan

Dalam jaringan komputer terdapat dua tipe jaringan, yaitu :

2.1.2.1 Model *Peer to Peer*

Pada model jaringan terdapat hanya dua komputer yang saling terhubung. Masing-masing komputer mempunyai kedudukan yang setara atau setingkat sehingga komputer manapun dapat memulai (*initiate*) dan mengendalikan hubungan dalam jaringan. Model Jaringan ini merupakan jenis yang paling sederhana. Model jaringan *peer to peer* dapat dilihat pada gambar 2.1.



Gambar 2.1 *Model Jaringan Peer to peer*

2.1.2.2 Model *Client Server*

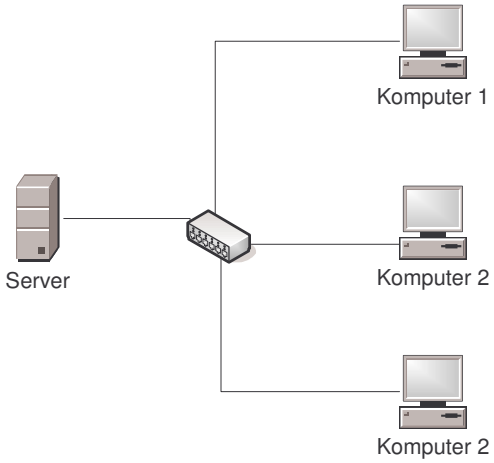
Konsep Jaringan *client-server* ini membedakan dengan jelas kedudukan suatu komputer, mana yang dapat memberikan layanan jaringan (*server*) dan mana yang hanya meminta layanan (*client*). Jika suatu komputer di*install* sebagai *server* yang memberikan segala sumber daya (*resource*), contohnya : *printer*, *modem* kepada komputer lain (*client)* yang terhubung ke jaringan. Untuk dapat saling berkomunikasi antara *server* dan *client*, maka digunakan suatu aplikasi jaringan yang dinamakan *client-server program,* dimana *server* menggunakan aplikasi jaringan yang disebut *server program*, sementara *client* menggunakan *client program* untuk dapat berkomunikasi dengan *server program* pada *server*. Model jaringan *client server* dapat dilihat pada gambar 2.2.

Beberapa sifat dari *client-server* diantaranya sebagai berikut :

1. *Server* dan *Client* berada pada posisi serta proses yang berbeda.
2. *Server* dan *Client* dapat dijalankan pada mesin yang sama atau berbeda.
3. Setiap proses dapat menyembunyikan informasi tersendiri.
4. Komunikasi yang dilakukan bisa bersifat protokol *peer to peer*.

Koneksitas *client ke Server :*

1. *Server* harus terlebih dahulu dijalankan (*running*).
2. *Server* harus menentukan *port* untuk koneksi yang masuk.
3. *Client* harus mengetahui nama *host* dari *server* beserta *port* yang digunakan untuk koneksi.
4. *Client* mengirim pesan ke *server* untuk melakukan suatu pekerjaan (*task*). *Server* dan *Client* bersifat *Asimetrik*

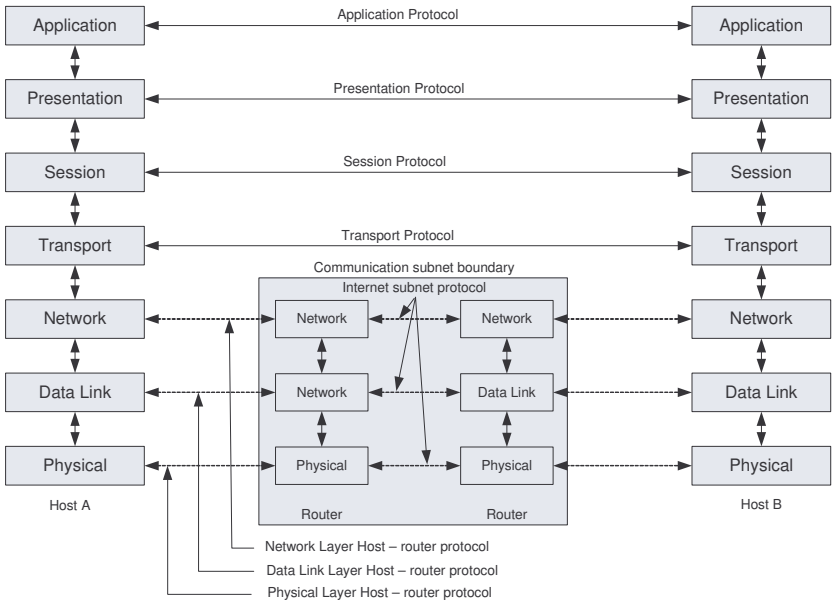


Gambar 2.2 *Model Jaringan Client Server*

2.1.3. Model OSI

Model OSI merupakan sistem terbuka yang dapat diartikan sebagai suatu sistem yang terbuka untuk berkomunikasi dengan sistem-sistem yang lainnya. Model referensi OSI ini memiliki tujuh *layer*. Adapun prinsip-prinsip yang digunakan bagai 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-fungsi tersebut sesuai dengan standarisasi internasional
4. Batas-batas layer harus dipilih untuk meminimalkan aliran informasi yang melewati *interface*.



Gambar 2.3 *Model Referensi OSI*

Sebuah *layer* tidak mendefinisikan protokol tunggal, tapi mendefinisikan suatu fungsi komunikasi data yang dapat dilakukan oleh sejumlah protokol. Jadi setiap *layer* dapat berisi banyak protokol, masing-masing menyediakan *service* yang cocok dengan fungsi *layer* tersebut. Sebagai contoh, *file transfer protocol* (FTP) dan *electronic mail protocol (E-Mail Protocol)* yang keduanya menyediakan *service* pada *user* dan keduanya merupakan bagian dari *layer* aplikasi. Model referensi OSI dapat dilihat pada gambar 2.3.

Setiap protokol berkomunikasi dengan pasangan (*peer)* protokol yang sama dalam *layer* yang sama dalam *remote-system* misalnya FTP lokal merupakan *peer* dari FTP *remote*. Jadi secara abstrak setiap protokol hanya peduli terhadap komunikasi dengan *peer*-nya, tak perduli dengan *layer* di bawah dan di atasnya. Adapun urutan *layer* dari *layer 1* sampai *layer* 7 sebagai berikut :

1. **Lapisan *Physical***
    a. Lapisan ini bertugas mendefinisikan besaran-besaran elektrik yang dipakai untuk berhubungan, misalnya atas tegangan yang benar (*Valid*) logika 0 dan 1, bentuk konektor yang digunakan, cara penyandian isyarat pada kabel dan lain-lain.
    b. Melaksanakan hubungan fisik melalui jalur transmisi .
2. **Lapisan *Data Link***
    a. Lapisan ini bertanggung jawab terhadap pengiriman data yang benar (*error free*) .
    b. Mengirim data secara paket dan memeriksa kesalahan pada pengiriman data.
    c. Menetapkan *prosedur* pengaksesan.
3. **Lapisan *Network***
    a. Lapisan ini menentukan jalur atau *route* yang harus ditempuh oleh paket untuk mencapai tujuannya, yang biasanya disebut *routing layer.*
    b. Pada lapisan ini pula menentukan *prosedur* untuk menghindari terjadinya kemacetan serta mengalihkan perjalanan paket ke jalur yang lain serta memungkinkan bukan jalur yang pendek.
4. **Lapisan *Transport***
    a. Lapisan ini mengatur bagaimana data atau informasi itu akan dibawa ke tempat tujuan termasuk juga jaminan kualitas dari pelayanan pengiriman data.
    b. Lapisan ini pula memilih fasilitas pengiriman yang sebagaimana yang dikehendaki.

5. **Lapisan *Session***

   a. Lapisan ini berfungsi menentukan kapan dimulai dan kapan diakhirinya suatu babak atau *session* komunikasi atau koneksi.

   b. Suatu babak akan dimulai jika babak sebelumnya telah selesai.

   c. Melihat data yang sudah dikirim dan diterima oleh proses.

6. **Lapisan *Presentation***

   a. Lapisan ini mendefinisikan bagaimana aplikasi dapat memasuki jaringan dan mengatur supaya data yang dipertukarkan atau dikirim ke lapisan aplikasi atau pemakai akhir dapat dimengerti bentuknya.

   b. Dalam lapisan ini, peralatan atau sistem yang menggunakan format data yang berlainan dapat saling berkomunikasi .

   c. Lapisan ini juga menyediakan *encryption* ataupun kompresi data jika diperlukan.

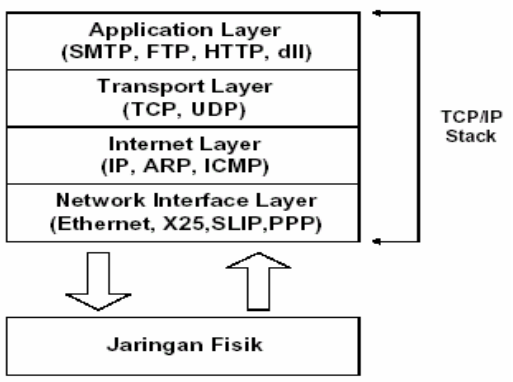
7. **Lapisan *Aplication***

   a. Lapisan ini berfungsi mengelola interaksi antara program dan pemakai.

   b. Lapisan ini juga menerima perintah atau input dari pemakai dan mengembalikan kode *error* atau pesan kesalahan kepada pemakai jika terjadi *error*.

2.1.4. Model TCP/IP

Model ini terdiri dari empat lapisan. Pada model ini tidak memiliki *presentation* dan *session layers*. Fungsi kedua lapisan ini dapat dilakukan sesuai dengan kebutuhan. Model model TCP/IP dapat dilihat pada gambar 2.4.



Gambar 2.4 *Model TCP/IP*

Berikut merupakan penjelasan lapisan-lapisan atau *layer-layer* pada model TCP/IP mulai dari lapisan teratas :

1 Aplication Layer

a. *Layer* ini melingkupi semua proses yang menggunakan protokol *transport layer* untuk mengirimkan data.

b. Karena *layer* ini berhubungan langsung dengan *service* yang ditawarkan pada pemakai jaringan maka protokol-protokol baru masih ditambahkan yang memperkaya pelayanan pada *user*.

2 Transport Layer

Dua protokol terpenting dalam layer ini adalah *Transmission Control Protocol* (TCP) dan *User datagram Protocol* (UDP). TCP menyediakan pelayanan pengiriman data yang handal dengan deteksi dan koreksi kesalahan dari ujung ke ujung (*end to end*). Sedangkan UDP menyediakan pelayanan pengiriman *datagram* yang *connection-less* dan tanpa dilengkapi deteksi dan koreksi

kesalahan. Kedua *protokol* mengirimkan data antara *layer* aplikasi dan *layer internet*. TCP merupakan protokol yang *connection oriented* dan *handshake*.

3 Internet Layer

a. Internet Protokol (IP)

IP merupakan inti dari TCP/IP dan merupakan protokol terpenting dalam *Internet Layer*. IP menyediakan pelayanan pengiriman paket elementer dimana jaringan TCP/IP dibangun.

Fungsi *Internet Protocol (IP)*

1. Mendefinisikan datagram, yang merupakan unit transmisi elementer di internet.
2. Mendefinisikan skema pengalamatan internet.
3. Melewatkan data antara *Network Access Layer* dan *Host to Host Transport layer* serta *routing datagram* ke *remote host.*
4. Menjalankan fragmentasi dan penyusunan kembali *datagram*.
5. IP merupakan protokol yang *Connectionless* (tidak memerlukan *handshake*), tidak dilengkapi dengan *error detection* dan *error recovery*.

b. Datagram

Datagram adalah format paket yang didefinisikan oleh IP. Internet sebagaimana asalnya *ARPANET* merupakan jaringan yang berbasis pada *packet-switching*. Jadi datagram merupakan unit transmisi elementer dalam jaringan TCP/IP. IP mengirimkan datagram dengan mengecek *destination address* dalam *header* kontrol di awal datagram. Jika *address* tujuan tidak berada di jaringan lokal maka paket dilewatkan ke *gateway* (*device* yang men-*switch* paket antara

jaringan fisik yang berbeda). Memutuskan *gateway* yang mana yang digunakan untuk mencapai *address* tujuan disebut sebagai *routing*.

4 Network access layer

a. Merupakah *layer* terbawah dari hirarki protokol TCP/IP.
b. Menyediakan sarana untuk sistem untuk mengirim data ke *device* lain yang terhubung ke *network*.
c. Mendefinisikan bagaimana datagram ditransmisikan pada *network*.
d. Fungsi lain yang ditangani pada *level* ini termasuk enkapsulasi datagram ke dalam *frame* yang ditransmisikan oleh jaringan dan konversi *IP address* ke dalam alamat yang cocok untuk jaringan fisik dimana *datagram* ditransmisikan.

2.2. Telekomunikasi

2.2.1. Mobile

Dalam dunia telekomunikasi, Amerika dan Eropa dapat dikatakan sudah berkembang lebih maju dibandingkan belahan dunia lainnya. Namun inovasi telepon seluler baru dimulai di tahun 1947. Waktu itu para peneliti telekomunikasi melihat peluang penggunaan frekuensi radio yang bisa dimanfaatkan sebagai media komunikasi. Inovasi teknologi ini tak lepas dari perjalanan telepon mobil tahun 1910 yang ditemukan oleh Lars Magnus Ericsson. Pria asal Swedia ini tak lain pendiri bisnis telekomunikasi yang kelak kita kenal dengan Ericsson (kini Sony Ericsson). Sebelumnya Ericsson hanyalah sebuah perusahaan kecil dengan memfokuskan bisnis pada peralatan telegraf. Telepon mobil kemudian juga dipakai oleh Departemen Kepolisian Detroit Michigan tahun 1921. Saat itu

hampir semua mobil polisi dilengkapi dengan peralatan komunikasi yang memudahkan pemantauan gerakan aparat. Sistem operasinya menggunakan frekuensi di bawah 2 MHz. Sekitar 20 tahun kemudian, revolusi bidang telekomunikasi itu pun dimulai. Sejalan dengan penemuan komponen baru seperti gulungan kabel dan tabung triode, para peneliti mulai merancang sebuah alat telekomunikasi yang bisa digunakan tanpa kabel. Terlebih saat mikroprosesor yang kecil ukurannya dan teknologi digital ditemukan. Dengan komponen ini, biaya bisa ditekan.Di Amerika sendiri, teknologi seluler baru dikembangkan setelah Perang Dunia II. Teknologi seluler sesungguhnya berisi sel-sel dengan berbagai macam model. Setiap sel memiliki base station dengan pancaran pada range 900 hingga 1.800 MHz. Dari base station kemudian dipancarkan ke jangkauan area yang bisa mencapai puluhan kilometer.

Melihat celah pemanfaatan teknologi radio, kemudian memacu berbagai perusahaan untuk melakukan bisnis di bidang ini. Di Skandinavia, Ericsson melancarkan bisnis telekomunikasi hingga Stockholm pernah disebut sebagai kota dengan penggunaan telepon paling gencar. Begitu halnya dengan Finlandia. Tahun 1960, Finnis Cable Works yang semula berbisnis di bidang kabel melakukan ekspansi dan mendirikan divisi elektronik. Bjorn Westerlund, sang presiden perusahaan itu dua tahun kemudian mengembangkan bisnis transmisi radio. Namun baru setahun kemudian, bidang telekomunikasi disabet untuk memperkuat jajaran bisnis elektronik. Dan di tahun 1967, Westerlund giat mendirikan perusahaan yang kelak kita kenal dengan nama Nokia. Telepon mobile alias telepon bergerak kemudian menjelma menjadi peralatan bagi orang-orang yang memiliki mobilitas tinggi. Era 1970-an, di Eropa Nokia dan Ericsson

menjelma menjadi sebuah perusahaan telekomunikasi yang besar. Di Amerika, Motorola pun unjuk gigi.

2.2.2. Symbian OS

Symbian OS adalah **sistem operasi 32 bit**, dengan konsep *little endian* dan berjalan pada beberapa tipe arsitektur mikroprosessor ARM. Symbian proses bekerja dengan prinsip ***pre-emptive multitasking***. Dukungan terhadap *device-device* terintegrasi dalam *kernel* sebagai *kernel extension* yang ditulis dalam *dynamic linking library* (DLL) yang terpisah. *Kernel* berjalan dalam mode *privileged* dan memberikan servis ke aplikasi yang berjalan dalam mode *unprivileged* lewat *user library*. Symbian OS juga memberikan kumpulankumpulan *library* seperti *networking* (TCP/IP, PPP, FTP), *Communication* (*Bluetooth*, IrDA). Untuk mengakses servis-servis tersebut dengan menggunakan konsep hubungan *client-server*. *Client* menggunakan servis API yang diberikan oleh *server* untuk
berkomunikasi dengan *server*. Semua hubungan komunikasi *client-server* diatur oleh *kernel*.

Symbian OS memiliki beberapa kelebihan diantaranya sebagai berikut:

1. *Small*, kaya *feature*.
2. *Platform* terbuka untuk aplikasi-aplikasi *third-party*.
3. Konektifitas yang baik dengan perangkat lain.
4. *Platform* yang berkembang.
5. *High performance*, 32 bit OS dengan *pre-emptive multitasking*.

6. *Long battery life*.
7. Dukungan dan komitmen dari pembuat ponsel dunia.
8. Aplikasi yang dapat dirancang sekali dan berjalan pada beberapa *device*.

Symbian OS mempunyai beberapa *design* sesuai dengan *device family*-nya atau yang disebut sebagai *Device Family Reference Design* (DFRD). DFRD ini merupakan spesifikasi dalam hal *user interface* dan *hardware configuration*, sehingga banyak tipetipe ponsel berbeda yang memanfaatkan sistem operasi ini, Macam-macam DFRD yaitu:

1. *Crystal*, untuk yang kaya *feature* seperti tipe *Communicator*, dimana mempunyai *full keyboard*, tampilan besar lcd mendatar dan menampung banyak informasi untuk kepentingan bisnis.
2. *Pearl*, untuk *smartphone* dimana seperti ponsel standar dengan lcd kecil dan *keyboard* yang terbatas.
3. *Quartz*, untuk ponsel dengan kemampuan *pen-based* atau *touch-screen* yang tidak memerlukan *keyboard*.

2.2.2.1. Komponen

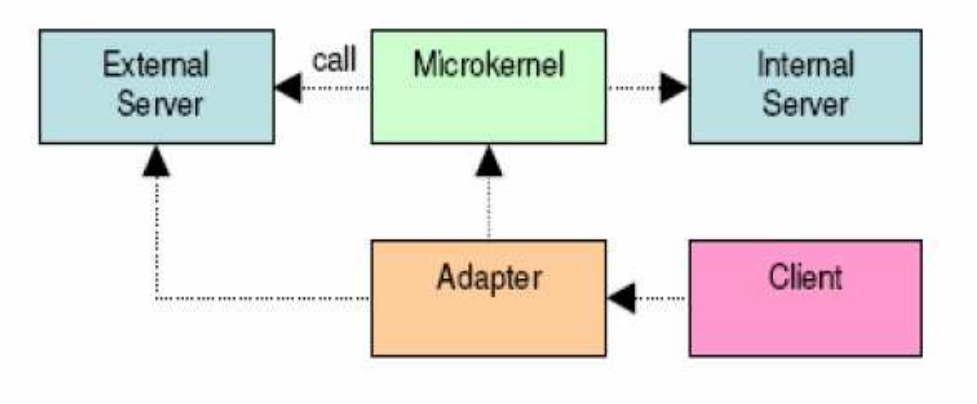
Symbian OS merupakan sebuah sistem operasi yang mengatur seluruh sumber daya yang ada didalam ponsel. Symbian OS disusun dari enam komponen, yaitu:

1. *Kernel*
2. *Middleware*
3. *Application Engine*
4. *UI Framework*

5. *Synchronization*
6. *JVM*

1. Kernel

Merupakan inti dari sistem operasi yang terdiri dari *device driver*, *data table*, dan program yang memungkinkan user berinteraksi dengan perangkat keras. *Kernel* merupakan program yang berjalan setiap waktu dan mengatur layanan-layanan yang diberikan ke user. Symbian OS merupakan sistem operasi berbasis *microkernel* yaitu hanya elemen-elemen yang penting yang berada di *kernel* sedangkan fungsi yang lain ada dalam *middleware*, sehingga membuat *kernel* sangat ringkas dan arsitekturnya menjadi lebih modular. Dari awalnya Symbian OS merupakan sistem operasi 32-bit yang mendukung *multitasking* dan *multithreading*. Ukuran *microkernel* sekitar 5% dari keseluruhan sistem operasi, yang berkisar antara 500kB sampai 15MB tergantung pada ada tidaknya dukungan java dan aplikasi-aplikasi lain yang ikut diinstal. Pemisahan inti dan komponen lain membuat sistem sangat modular, yang akan meningkatkan *portabilitas platform* dan membuat proses *upgrade* dan perubahan *platform* lebih mudah dilakukan. Berikut gambar 2.5 merupakan arsitektur *microkernel* Symbian OS.



Gambar 2.5 *Arsitektur Microkernel Symbian OS*

(Sumber : www.materikuliah.com)

Komponen *microkernel* menerapkan *atomic service* yang diperlukan keseluruhan aplikasi di sistem, mengontrol sumber daya seperti memori, proses, *thread*, dan IPC. Fungsi yang tidak mungkin dimasukkan ke *kernel* kerena alasan kompleksitas dan besar, dipisah ke *internal server*. *Internal server* mengembangkan fungsi inti, misalnya untuk menangani *graphic* dan media penyimpanan, dapat mempunyai proses sendiri atau *share library*. *External server* menggunakan servis dari *microkernel* dan *internal server* untuk memberikan servis ke *client*. *External server* menangani komunikasi (*serial comm server, socket server, message server, telephony server*), *graphics* (*window, font, bitmap server*), *audio*(*media server*), *storage media* (*file server*). *External server* dieksekusi pada proses tersendiri, setiap *external server* memberikan *client side* API yang membungkus IPC antara *client* dengan *server*. Sedangkan *Adapter* memberikan *interface* yang transparan untuk *client* seperti detail komunikasi.

2. Middleware

Merupakan kumpulan *library*, *data storage*, dan program yang mengimplementasikan sistem servis. Kesemuanya itu tidak perlu diletakkan dalam *kernel*. Manajemen data, komunikasi dan *graphics* termasuk servis sistem tersebut. Sebagai contoh *window system* yaitu yang mengatur bagaimana user berinteraksi dengan perangkat keras, hal ini tidak cukup penting untuk diletakkan didalam *kernel*. Symbian OS menggunakan *server* untuk implementasi *middleware*. Idenya adalah *server* yang dapat mengatur servis dari beberapa *client* dan merespon permintaan-permintaan tersebut. Dengan membuat sebuah *layer* baru untuk *middleware*, *designer* dapat dengan mudah merancang sistem servis baru tanpa harus merubah *kernel*.

3. Application Engine

Aplikasi *user level* dapat memanfaatkan servis yang diberikan pada *level middleware* yaitu dengan menggunakan *application engine*. *Application engine* yang melakukan koordinasi untuk mengakses sumber daya yang tidak begitu penting. *Application engine* diantaranya adalah sebagai berikut: *Agenda engine*, *Contact engine*, *Sheet engine*, *Alarm server and WorldTime engine, Spell engine*, dan *Help engine*.

4. User Interface Framework

User interface merupakan faktor utama untuk sebuah ponsel, yang menyebabkan ponsel mudah digunakan, mudah dirubah dan diprogram. Symbian OS menerapkan *framework* untuk *user interface* sehingga mudah diadaptasi. Dalam Symbian OS ada dua *user interface* dalam *framework*-nya yang

menggunakan komponen yang umum seperti kontrol dan dialog yaitu *Uikon* dan *Eikon*.

5. Synchronization

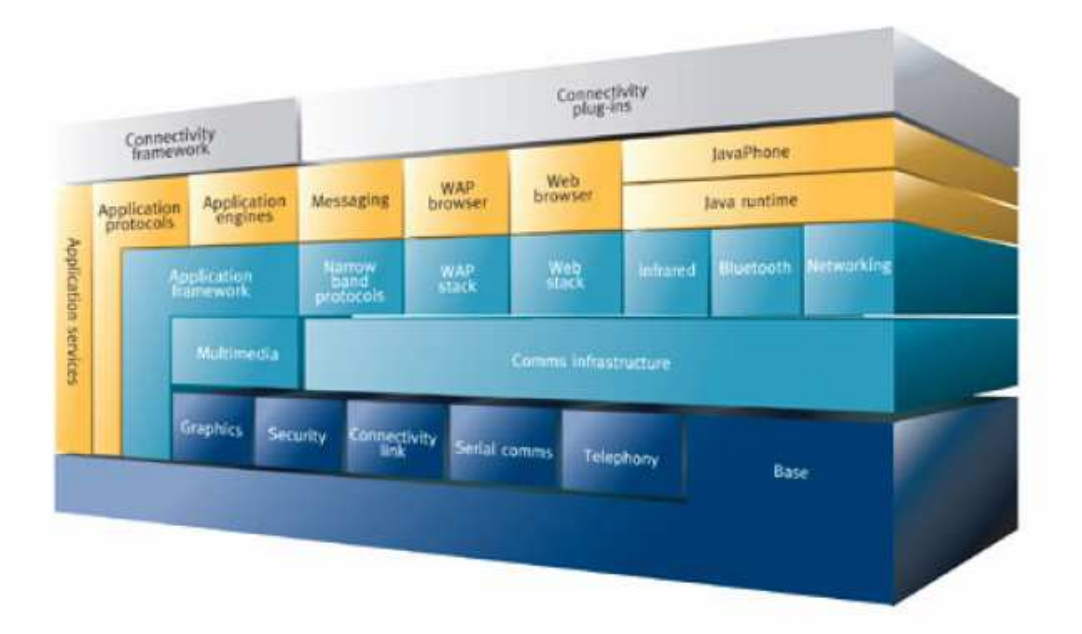
Sinkronisasi dengan peralatan lain ataupun dengan komputer desktop menjadi perhatian dalam teknologi ponsel. Symbian OS menerapkan 3 bagian untuk sinkronisasi yaitu *Connection manager* yang melakukan inisialisasi koneksi antar *device* dan mendeteksi jikalau ada *device* yang ingin melakukan koneksi. *Connectivity server* yang mengimplementasi sesi sinkronisasi seperti *file browsing*, *file sinkronisasi*, *back and restoration*. *File Converter* yang melakukan transfer data antar format yang berbeda.

6. JVM

Symbian OS mengimplementasikan teknologi java yang dikenal dengan J2ME. JVM merupakan salah satu komponen dalam Symbian OS yang untuk perangkat ponsel dikenal dengan *kilo virtual machine* (KVM). *JavaPhone* dan *PersonalJava* yang merupakan bagian dari J2ME *specification* juga diimplementasikan dalam Symbian OS.

2.2.2.2. Arsitektur

Gambar 2.6 menunjukkan Arsitektur Symbian OS versi 6.1. *Subsystem* yang lebih atas mempunyai ketergantungan dengan beberapa *subsystem* yang dibawahnya, walaupun tidak selalu demikian. Sebagai contoh, WAP *stack*, *Bluetooth*, *Infrared* dan *Narrow band* merupakan protokol-protokol yang ada dalam *comms infrastructure*. *Messaging* juga bergantung pada WAP *stack*, tetapi java tidak bergantung dengan *infrared*.



Gambar 2.6 *Arsitektur Symbian V6.x*

(Sumber : www.materikuliah.com)

Komponen-komponen dalam arsitektur diatas selengkapnya sebagai berikut:

1. *Base*, Terdiri atas Sistem *runtime* yang sangat dasar, *low-level security*.
2. *Application framework*, Terdiri atas API untuk manajemen data, *text, clipboard,*
3. *graphics, internationalization*, dan inti komponen GUI.

4. *Multimedia*, Terdiri atas *audio recording* dan *playback*, fungsi-fungsi yang berhubungan dengan *image*.
5. *Communication infrastructure and network stacks*, Terdiri atas stack komunikasi yang luas termasuk TCP/IP,GSM,GPRS dan WAP. Komunikasi Personal seperti
6. *infrared, Bluetooth* dan *serial*.
7. *Messaging*, Terdiri atas *internet mail*,SMS dan Fax.
8. *Browsing*, Terdiri atas WML dan HTML *browsing engine*.
9. *Application protocols, service and engines*, Terdiri atas *engine* untuk manajemen *contact, schedule* dan *to-do list* manajemen dan aplikasi-aplikasi yang lain.
10. *Java*, Terdiri atas PersonalJava 3.0 spec JVM-*based java runtime system* dengan JavaPhone 1.0 APIs.
11. *Connectivity*, Terdiri atas *converter* dan *viewer* untuk format data *foreign* termasuk *attachment mail* Microsoft word. *Framework* komunikasi untuk berhubungan dengan PC.
12. *Tools*, Terdiri atas *tool* untuk membuat aplikasi, ROMs dan untuk *debug target* aplikasi.

2.2.3. Java Programming

2.2.3.1. Object Oriented Programming

Object Oriented Programming (OOP) merupakan metode pemrograman yang mencoba melihat permasalahan lewat pengamatan dunia nyata dimana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi

tertentu. Jadi, ide dasar dari OOP adalah mengkombinasikan data dan fungsi untuk mengakses data menjadi satu kesatuan unit yang dikenal dengan nama object. OOP memiliki beberapa konsep, yakni:

1. Pengkapsulan *(Encapsulation)*

   Pengkapsulan adalah proses pemaketan data objek bersama method-methodnya. Manfaat utama pengkapsulan adalah untuk menyembunyikan rincian-rincian implementasi dari objek lain.

2. Pewarisan *(Inheritance)*

   Pewarisan adalah salah satu konsep terpenting pada OOP dan memiliki dampak langsung pada merancang dan menulis class. Pewarisan adalah penciptaan class baru dengan mewarisi karakteristik class yang telah ada, ditambah karakteristik dari class baru tersebut. Sehingga sebuah subclass akan memiliki semua method dan variable dari superclass secara otomatis tanpa perlu mendefinisikan ulang semua perilaku pada superclass tersebut.

3. *Polymorphism*

   Polymorphism berarti memiliki banyak bentuk. Dua objek atau lebih dikatakan polymorphic bila objek-objek tersebut memiliki interface yang identik namun memiliki perilaku yang berbeda.

Java adalah pemrograman yang berorientasi pada objek, merupakan bahasa yang dapat dijalankan dimanapun dan disembarang platform apapun. The Java 2 Platform tersedia dalam tiga edisi untuk keperluan berbeda, yaitu:

1. *Java 2 Standard Edition* (J2SE)

   J2SE menyediakan lingkungan pengembangan yang kaya fitur, stabil, aman, *cross-platform*. Edisi ini mendukung fitur konektifitas basisdata, rancangan user interface, input/output, dan pemrograman jaringan dan termasuk sebagai paket-paket dasar bahasa java.

2. *Java 2 Enterprise Edition* (J2EE)

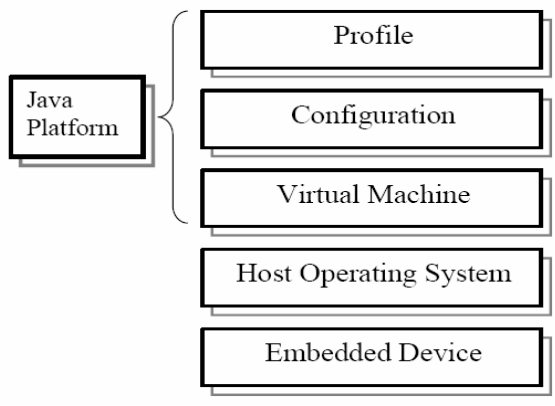
   J2EE menyediakan kakas untuk membangun dan menjalankan multitier enterprise application. J2EE berisi paket-paket pada J2SE ditambah paket-paket untuk mendukung pengembangan enterprise Java Beans, Java Servlets, Java Server Pages, XML, dan kendali transaksi yang fleksibel.

3. *Java 2 Micro Edition* (J2ME)

   J2ME diimplementasikan pada beragam consumer electronic product seperti pager, *smart card*, *cell phone*, *handheld* PDA, dan *set-top box*. J2ME menggunakan sekumpulan paket lebih kecil. J2ME berisi subset dari paket-paket pada J2SE ditambah paket spesifik Micro Edition berupa javax.microedition.io.

2.2.3.2. J2ME

Sun Microsystem telah mengembangkan platform *Java 2 Micro Edition* (J2ME), sebuah lingkungan Java yang diarahkan untuk sistem yang kecil dengan *resource* yang terbatas. Adapun arsitektur dari J2ME, seperti pada gambar 2.7.



Gambar 2.7 *Arsitektur Platform J2ME*

(Sumber : www.java.sun.com)

Adapun komponen-komponen dari J2ME, terdiri atas:

1. *Java Virtual Machine* (JVM), yang digunakan untuk menjalankan program-program Java pada *emulator* atau *handheld device*.
2. Java *Application Programming Interface* (API), merupakan kumpulan *library* untuk menjalankan dan mengembangkan program Java pada *handheld device*
3. *Tools* lain untuk mengembangkan aplikasi java, semacam *emulator* Java Phone, *emulator* Motorolla

2.2.3.3. J2ME Configuration

J2ME *Configuration* dirancang untuk menyediakan *library* standar yang mengimplementasikan fitur standar dari sebuah *handheld devices* dikarenakan fitur dari setiap *handheld* berbeda-beda. J2ME Configuration mendefinisikan lingkungan kerja J2ME *runtime* yang meliputi *Java Virtual Machine* yang digunakan untuk menjalankan program Java. Sampai saat ini, J2ME *Configuration* memiliki 2 kategori yakni:

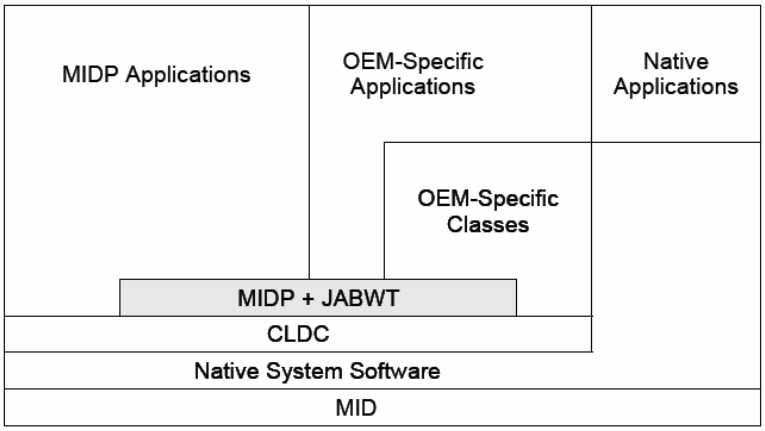
1. *Connected Limited Device Configuration* (DLDC), digunakan untuk aplikasi Java pada *handphone*, PDA, PocketPC, dan *two way pagers*. Yang pada umumnya hanya memiliki memori berukuran 160-512 KiloBytes.
2. *Connected Device Configuration* (CDC), umumnya digunakan untuk apliaksi Java pada perangkat-perangkat *handheld devices* dengan ukuran memori setidaknya 2 MegaBytes dan memiliki sekumpulan fitur yang lebih besar dari CLDC, seperti Internet TV, Nokia Communicator, dan TV pada mobil.

2.2.3.4. J2ME Profile

J2ME Profile menyediakan implementasi-implementasi tambahan yang sangat spesifik dari sebuah handheld devices yang pada umumnya sangat tergantung pada jenis perangkat handheld yang digunakan. J2ME Profile memiliki lima kategori, yakni: *Mobile Information Device Profile* (MIDP), *Foundation Profile*, *Personal Profile*, RMI *Profile*, dan *Personal Digital Assistance Profile*. Namun, yang akan digunakan untuk tugas akhir ini adalah MIDP. Seperti namanya, profile ini ditargetkan pada perangkat *mobile* khususnya *mobile phone*.

2.2.3.5. Java APIs For Bluetooth Wireless Technology

Java APIs for Bluetooth Wireless Technology (JABWT) merupakan J2ME optional *package* yang didefinisikan oleh *Java Community Process* (JSR-82). *Optional Package* ini menyediakan API secara umum untuk perkembangan bluetooth. Gambar 2.8 berikut merupakan ilustrasi hubungan antara JABWT pada platform J2ME.



Gambar 2.8 *Posisi JABWT pada J2ME*

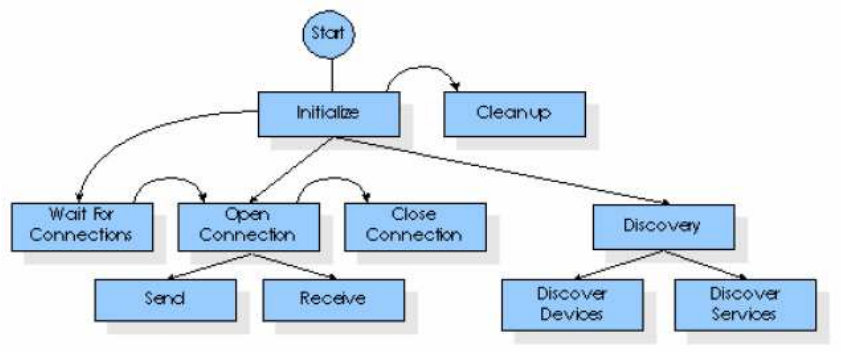
(Sumber : www.java.sun.com)

Gambar di atas menunjukkan bahwa JABWT beroperasi di atas CLDC dan dimaksudkan untuk memberikan kemampuan pada profile seperti MIDP. JABWT mendefinisikan dua *package*, seperti yang terlihat pada table di bawah.

Tabel 2.1 *JABWT Package*

Package	Menyediakan
javax.bluetooth	core bluetooth API
javax.obex	Object Exchange (OBEX) API

Menggunakan Java APIs for bluetooth terdiri dari inisialisasi *stack* bluetooth, menemukan *device* atau layanan yang dekat, membuka dan menutup serta menunggu koneksi, dan perform I/O. Gambar 2.9 merupakan gambar ilustrasi perbedaan operasi bluetooth yang dapat digunakan.



Gambar 2.9 *Menggunakan Bluetooth API*

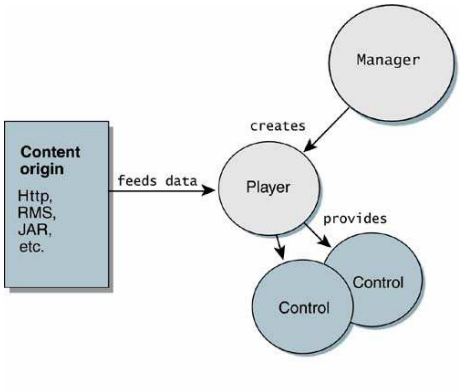
(Sumber : www.java.sun.com)

2.2.3.6. Mobile Media API (MMAPI)

Mobile Media API (MMAPI) merupakan sebuah *package* yang mendukung aplikasi multimedia pada perangkat J2ME, standar spesifikasi java ini didefinisikan oleh *Java Community Process* (JCP) pada JSR 135. MMAPI menyediakan beberapa kemampuan multimedia untuk perangkat *mobile*, diantaranya *playing* audio dan video, *playback*, merekam data audio dan video, mengambil gambar, dan fitur yang lainnya.

Mobile Media API menyediakan kekuatan, flexibilitas, dan *interface* yang *simple* dalam kemampuan multimedia. Adapun MMAPI memiliki tiga buah komponen utama, yaitu: *player*, *control*, dan *manager*. *Manager class* digunakan untuk membuat *player*. Sedangkan *player* itu sendiri merupakan sebuah *interface*.

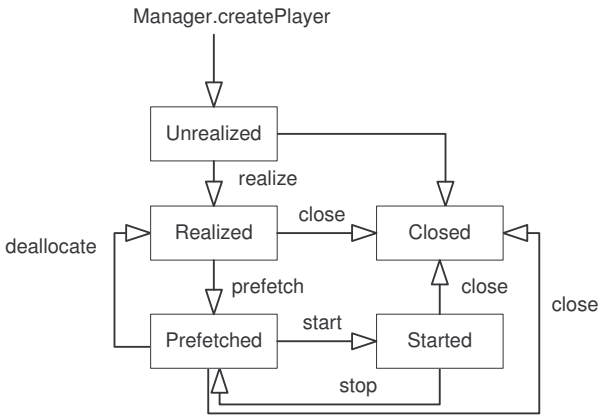
Ketika *player* terbentuk, sebuah aplikasi dapat me-*request* objek *control* yang mengenkapsulasi perilaku untuk mengendalikan *playback*, *capture*, dan yang lainnya. Hubungan dari ketiga media *framework* tersebut seperti terlihat pada gambar 2.10.



Gambar 2.10 *Arsitektur MMAPI*

(Sumber : www.java.sun.com)

Gambar 2.11 di bawah mengilustrasikan state diagram MIDP 2.0 player. Object *player* memiliki lima state: *UNREALIZED, REALIZED, PREFETCHED, STARTED,* dan *CLOSED*. Player pada pertama kali terbentuk berada pada *state unrealized*. Ketika *state* berubah dari *unrealized* menjadi *realized*, *player* melakukan komunikasi yang diperlukan untuk melokasikan semua *resource* yang diperlukan (seperti komunikasi ke server). Pada umumnya, *player* bergerak dari *unrealized* ke *realized state*, kemudian *prefetched state* dan pada akhirnya *started state*. *Player* akan berhenti ketika mencapai *end of media* atau ketika method *stop* dipanggil.



Gambar 2.11 *State Diagram MMAPI*

(Sumber : www.java.sun.com)

2.3. 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 kontinyu. Masing-masing *frame* merupakan gambar / citra (*image*) digital. Suatu *image* digital direpresentasikan dengan 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 dimana *image* disampling disebut sebagai *picture elements* atau sering dikenal sebagai pixel. Karakteristik video digital ditentukan oleh resolusi (*resolution*) atau dimensi *frame (frame dimension)*, kedalaman piksel *(pixel depth),* dan laju frame *(frame rate).* Karakteristik – karakteristik ini yang akan menentukan

kualitas video dan jumlah bit yang dibutuhkan untuk menyimpan atau mentransmisikannya.

2.3.1. Resolusi

Resolusi (*resolution*) atau dimensi frame (*frame dimension*) adalah ukuran sebuah frame pada video digital. Resolusi dinyatakan dalam pixel x pixel. Semakin tinggi resolusi, semakin baik kualitas video tersebut, dalam arti bahwa dalam ukuran fisik yang sama, video dengan resolusi tinggi akan lebih detil. Namun, resolusi yang tinggi akan mengakibatkan jumlah bit yang diperlukan untuk menyimpan atau mentransmisikannya meningkat. Misalnya pada format VGA, resolusinya adalah 320 pixel x 240 pixel, ukuran ini umum digunakan untuk file video.

2.3.2. Kedalaman Bit

Kedalaman bit (*bit depth*) menentukan jumlah bit yang digunakan untuk merepresentasikan tiap piksel pada sebuah frame. Kedalaman bit dinyatakan dalam bit per piksel. Semakin banyak jumlah bit yang digunakan untuk merepresentasikan sebuah piksel, berarti semakin tinggi kedalaman pikselnya, maka semakin tinggi pula kualitasnya, mengakibatkan jumlah bit yang diperlukan menjadi lebih tinggi. Satu byte (8 bit) untuk tiap piksel, diperoleh 2^8 atau 256 level intensitas. Dengan level intensitas sebanyak itu, umumnya mata manusia sudah dapat dipuaskan. Ke dalam 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).

2.3.3. Laju Frame

Laju frame (*Frame rate)* menunjukkan jumlah frame yang digambar tiap detik, dan dinyatakan dengan frame per detik. Sehubungan dengan laju frame ini, ada dua hal yang perlu diperhatikan, yaitu kehalusan gerakan (*smooth motion*) dan kilatan (*flash*). Kehalusan gerakan ditentukan oleh jumlah frame yang berbeda per detik. Untuk mendapatkan gerakan yang halus, video digital setidaknya harus menampilkan sedikitnya 25 frame per detik. Kilatan ditentukan oleh jumlah berapa kali layar digambar per detik. Dengan 20 frame per detik, kilatan sudah dapat dilenyapkan. Video yang berkualitas baik akan memiliki laju frame yang tinggi, setidaknya sesuai dengan mata manusia, yang berarti membutuhkan jumlah bit yang lebih tinggi.

2.3.4 Representasi Warna

Pada video digital, umumnya data video dipisahkan menjadi komponen-komponen, baik komponen warna maupun komponen kecerahan. Penyajian semacam ini disebut *component video*. Pada *component video*, tiap komponen dipisahkan dengan cara tertentu. Beberapa cara pemisahan komponen tersebut.

2.3.4.1 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. Sebagai contoh, pada RGB 24 bit, masing-masing komponen warna dinyatakan dalam 8 bit atau 256 level.

2.3.4.2 YUV

Pemisahan komponen tidak hanya dilakukan dengan pemisahan warna, namun dapat juga dilakukan dengan memisahkan menurut komponen kecerahan (*luminance*) dan komponen warna (*crominance*). Pada format PAL (*Phase Alternating Line*), sinyal kecerahan dinyatakan dengan Y, sedangkan dua sinyal warna dinyatakan dengan U dan V.

2.3.4.3 YIQ

Pemisahan sinyal video menjadi komponen kecerahan dan komponen warna dapat dilakukan juga sesuai dengan format NTSC (*National Television System Committee*), komponen kecerahan dinyatakan dengan Y, dan dua komponen warna dinyatakan dengan I dan Q. Karena persepsi mata manusia lebih peka pada kecerahan dari pada warna, maka *crominance* cukup di-*sampling* separuh dari *luminance.*

2.4 Kompresi Video

Beberapa standar kompresi video sudah banyak dikembangkan selama ini. *International Telecommunication Union* sektor standardisasi komunikasi (ITU-T) merekomendasikan H.261 dan H.263 untuk kompresi video. Karena keduanya merupakan aplikasi *real-time*, *delay* yang terjadi tidak terlalu besar.

Teknik kompresi video dapat digolongkan menjadi dua kelompok utama yaitu : *Lossy* dan *Lossless*. Teknik kompresi secara *lossy* dimaksudkan dengan teknik kompresi video dengan menghilangkan ketelitian data utama guna mendapatkan data sekecil mungkin (kompresi data sebesar mungkin). Teknik kompresi video secara *lossless* yaitu teknik kompresi data dengan mengurangkan

jumlah data yang terjadi duplikasi (memiliki simbol yang sama) sebelum terjadi kompresi.

a. *Lossy Compression*

Teknik kompresi dengan menghilangkan ketelitian dari data utama (*Lossy Compression*) membuktikan lebih efekftif diaplikasikan untuk kompresi data gambar dan data suara. Teknik ini efektif karena data yang berupa gambar dan suara relatif besar ukurannya bila disimpan dalam komputer, dan data yang telah dikompresi akan tetap memiliki kemiripan walau sebagian isi dari data yang telah dihilangkan.

b. *Lossless Compression*

Merupakan teknik kompresi untuk menghasilkan duplikasi dari inputan data dengan panjang bit yang lebih pendek. Tipe pemampatan ini digunakan untuk penyimpanan record pada database spreadsheet atau file pengolahan data.

2.4.1 Standar H.261

Standar H.261 dikembangkan untuk melayani kompresi video yang merupakan bagian dari standar H.320, kompresi H.261 didesain untuk pengiriman data melalui ISDN (*Integrated System Digital Network*) dengan *bandwidth* antara 64Kbps hingga 1.5Mbps, karena *bandwidth* melaui ISDN merupakan kelipatan 64Kbps, maka standar kompresi H.261 diatur dalam p x 64Kbps, dengan p berkisar antara 1 hingga 30fps, untuk mendapatkan kualitas video yang lebih tinggi. H.261 menerima format masukan CIF (*Common Intermediate Formats*) berukuran 352 x 288 pixel yang umumnya digunakan dalam *videoconferencing* dan format QCIF (*Quarter Common Intermediate Formats*) berukuran 176 x 144 pixel yang umumnya digunakan dalam *videophone*.

2.4.2 *Moving Picture Experts Group* (MPEG)

Moving Picture Experts Group (MPEG) berada di bawah gabungan 2 lembaga yang bekerja untuk standarisasi gambar bergerak dan kumpulan audiao, yaitu *International Standards Organization* (ISO) dan *the International Electro-Technical Commission* (IEC). Terdapat berbagai jenis standar MPEG.

MPEG-1 mempunyai aplikasi dengan target *bit rate* sekitar 1,5 Mbps. MPEG-1 hanya menangani *progresive video*, sehingga *interlaced video* harus dikonversi terlebih dahulu. Standard masukannya adalah SIF yang mempunyai resolusi 352 x 240 pixel, frame rate 30 *fps*, dan 8 bit per pixel. Namun, MPEG-1 juga dapat menangani frame hingga maksimum 4095 x 4095 pixel.

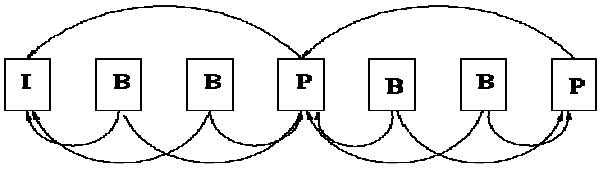
MPEG-2 merupakan perluasan yang kompatibel dengan MPEG-1. Target *bit rate*nya sekitar 4 Mbps hingga 10 Mbps. Aplikasi yang didukungnya meliputi *broadcast TV*, *video on demand*, maupun HDTV. MPEG-2 menerima *interlaced video* dan memanfaatkan redundansi antara field ganjil dan genap, inilah mengapa MPEG-2 populer untuk *TV encoding*.

MPEG-4 dikembangkan pada tahun 1998 oleh *Motion Pictures Expert Group*, dan digabungkan ke dalam spesifikasi *3rd Generation Partnership Project* (3GPP) untuk mobile multimedia. MPEG-4 merupakan tehnik video *compression* terbaru serta sangat cocok untuk aplikasi dengan *bandwidth* yang rendah, hal ini berpengaruh mengingat besar *bandwidth* untuk bluetooth sampai saat ini hanya mencapai 1 Mbps (732 kbps). MPEG-4 mendukung baik untuk *mobile* terminal yang kecil maupun untuk peralatan yang besar. Pada tabel 2.3 dapat dilihat berbagai macam standar kompresi video.

Tabel 2.2 Standar Kompresi Video

Standard	Digization Format	Compressed Bit rate	Example Applications
H.261	CIF / QCIF	X 64 Kbps	Video Telephony / Conferencing over x 64 Kbps channel (P=1:30) and LANs
H.263	SQCIF / QCIF	< 64 Kbps	Video Telephony / Conferencing and security surveillance over low bit-rate channels
MPEG-1 / ISO 11172	SIF	< 1.5 Mbps	Storage of VHS-quality Video on CD-ROM
MPEG-2 / ISO 13818 Low Main	SIF	< 4 Mbps	Recording of VHS quality Video
High 1440	4:2:0 4:2:0 4:2:0	< 15 Mbps < 30 Mbps < 60 Mbps	Digital Video Broadcasting
High	4:2:0 4:2:0 4:2:2	< 80 Mbps < 80 Mbps < 100 Mbps	High Definition Television (4/3 aspect ratio) High Definition Television (16/9 aspect ratio)
MPEG-4	Various		Versatile Multimedia coding standard
MPEG-7			Structure and content description of compressed multimedia information search engines

Setiap *sequence* video disusun dari seri *Group of Picture* (GOP). Sebuah GOP tersusun dari urutan gambar (*frame*) yang setidaknya memiliki satu I-*Encoded frame*. I encoding atau *Intra Encoding* digunakan untuk memberikan *random access* pada *compressed* video file. Setiap *frame* tersusun dari serangkaian *slices* yang saling berputar membentuk serangkaian *macro-block*. Model MPEG *video frame* dapat dilihat pada gambar 2.12.



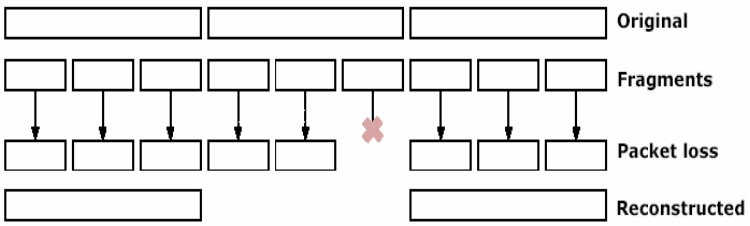
***Gambar 2.12** MPEG video frame*

(Sumber : www.portal.acm.net)

Ada tiga buah type *frame* dari MPEG, yakni:

1. I Frame (*Intra Coded Frame*), merupakan *frame* yang dikodekan secara bebas dari semua *frame* yang lain. *Encoding* ini digunakan sebagai *random access* dari *image* video.
2. P Frame (*Predictively Coded*), merupakan *frame* yang dikodekan berdasarkan *frame* sebelumnya.
3. B Frame (*Bi-directionally predicted frame*), merupakan *frame* yang dikodekan dengan menggunakan antara *frame* yang sebelumnya dan sesudahnya.

Untuk melakukan pengiriman data video akan dilakukan fragmentasi yakni dengan membuat paket-paket besar yang kemudian dijadikan beberapa *slice* yang lebih kecil untuk kemudian dikirimkan melalui jaringan, hal tersebut dilakukan agar *packet* yang besar tersebut dapat di*transmisi*kan melalui jaringan yang memiliki keterbatasan *bandwidth*. Proses fragmentasi dapat dilihat pada gambar 2.13.

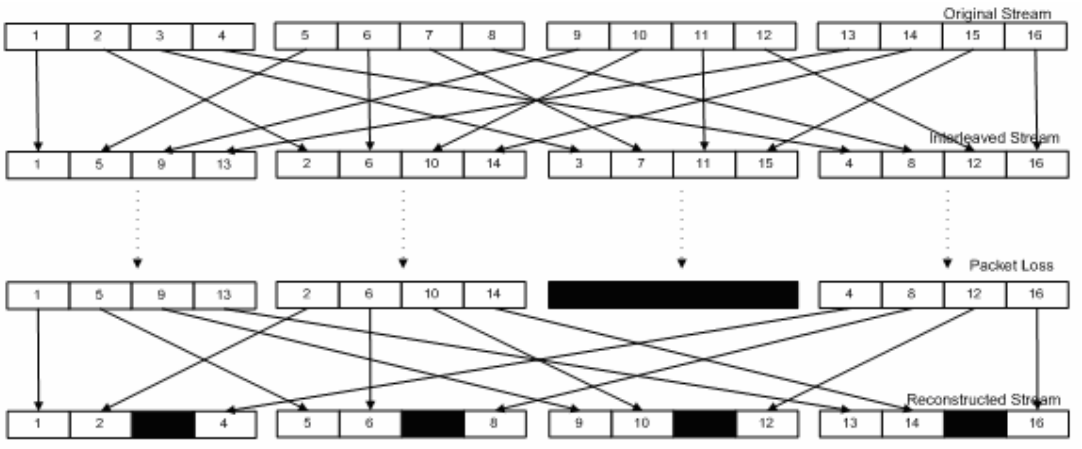


Gambar 2.13 *Proses Fragmentasi*

(Sumber : www.portal.acm.net)

Pada proses fragmentasi tersebut, jika salah satu *slice* dalam satu *packet* mengalami kerusakan, maka *packet* tersebut akan dibuang, sehingga *packet* tersebut akan dianggap sebagai *packet loss*. Untuk mengatasi masalah tersebut,

maka akan dilakukan modifikasi fragmentasi dengan tujuan agar lebih memaksimalkan penanganan terhadap packet *loss*, seperti yang terlihat pada gambar 2.14.



Gambar 2.14 *Modifikasi Fragmentasi*

(Sumber : www.portal.acm.net)

Original stream dibagi-bagi menjadi beberapa *chunk* (potongan) yang lebih kecil, kemudian *chunk* tersebut di-*interleaved* (diselipkan) pada packet-packet yang terpisah, sehingga jika terjadi *packet loss*, masih ada *chunk* lain untuk dilakukan rekonstruksi data.

2.4.3 Standar H.263

H.263 merupakan standar kompresi video untuk *bit rate* rendah sehingga dapat dilewatkan pada modem, antara 15 hingga 20 Kbps. H.263 menggunakan *half-pixel accuracy* pada *motion estimation*-nya sehingga meningkatkan kualitas gambar. Secara umum pola kerja H.263 sebagai berikut, *video frame* akan di enkode dengan video enkoder. Aliran video yang terkompres kemudian di proses

kembali ketika akan ditampilkan dan terjadi proses dekode menggunakan video dekoder, *frame* yang di dekode ini kemudian ditampilkan.

Proses kompresi *frame* terdiri dari dua macam metode yaitu kompresi *Intraframe* dan kompresi *Interframe*.

2.4.4 Kompresi Intra Frame

Kompresi Intraframe dilakukan dengan memanfaatkan redudansi spasial yang terdapat dalam suatu frame. Redudansi ini disebabkan karena adanya kesamaan antara sebuah piksel dengan piksel disekitarnya.

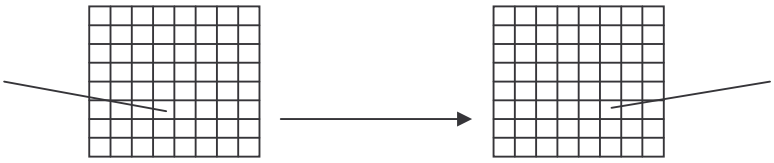
Kompresi Intraframe terdiri dari proses transformasi dan kuantisasi, dalam proses transformasi digunakan *Discrete Cosinus Transform* (DCT) untuk melakukan proses transformasi dari domain ruang ke domain frekuensi. Kuantisasi digunakan untuk memotong hasil transformasi, proses selanjutnya adalah pengkodean dengan menggunakan *Run Length Encoding* (RLE) dan *Variable Length Coding* (VLC).

Tahap paling awal pada kompresi intraframe adalah persiapan blok, yaitu suatu frame dibagi menjadi blok-blok yang tidak saling menindih. Pembagian blok ini diperlukan agar proses kompresi menjadi efisien, karena proses akan dilakukan pada blok-blok yang relatif kecil.

2.4.4.1 Discrete Cosine Transform (DCT)

Prinsip dasar yang dilakukan dengan Descrete Cosine Transform (DCT) adalah mentransformasikan data dari domain ruang ke domain frekuensi. Masukan proses DCT berupa matrik data dua dimensi N x N, keluaran proses DCT juga merupakan matriks data dua dimensi N x N, dimana f (x,y) sama dengan data pada domain ruang dan F (u,v) sama dengan domain frekuensi.

Tiap koefisiensi dari matriks keluaran ini merupakan nilai pada tiap frekuensi spasial dua dimensi. Pada gambar 2.15 di tunjukan proses DCT.



Gambar 2.15 *Discrete Cosine Transform*

Koefisien (0,0) merupakan koefisien pada frekuensi terendah dalam matriks. Koefisien ini disebut sebagai koefisien DC, yang paling menentukan pada blok, karena merupakan nilai rata-rata dari blok. Koefisien lainnya disebut sebagai koefisien AC, yang menerangkan jumlah daya spectral yang terdapat pada masing-masing frekuensi spasial .

Mata manusia lebih peka pada frekuensi rendah (pada kiri atas matriks), terutama frekuensi DC, daripada frekuensi tinggi (pada kanan bawah matriks). Hal ini dikarenakan distorsi yang terjadi pada frekuensi tinggi tidak merusak data secara signifikan. Sifat ini dimanfaatkan dengan memotong data pada frekuensi tinggi yang dilakukan dengan proses kuantisasi.

Pada dekompresi, untuk mentransformasikan kembali data dari domain frekuensi ke domain ruang, digunakan *inverse* dari *discrete cosine transform* atau IDCT.

2.4.4.2 Kuantisasi

Proses kuantisasi merupakan proses untuk mengurangi jumlah bit yang diperlukan untuk menyimpan suatu nilai dengan memperkecilnya. Proses ini diterapkan pada keluaran proses DCT. Kuantisasi dilakukan dengan membagi keluaran proses DCT dengan suatu nilai yang ditetapkan dalam matriks kuantisasi.

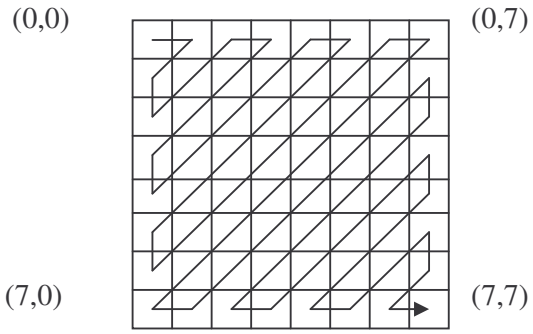
Quantum adalah matriks kuantisasi. Matriks kuantisasi dapat dipilih *uniform* atau *nonuniform*. Pada matriks kuantisasi uniform, semua koefisien mempunyai besar yang sama, sedangkan, pada matriks kuantisasi *nonuniform*, koefisien – koefisien pada matriks meningkat tajam dari titik pusat. Hal ini akan mengakibatkan nilai frekuensi tinggi pada keluaran DCT akan dipotong dengan cepat, sehingga kompresi yang dilakukan lebih efektif.

Hasil proses dekuantisasi cenderung mengalami distorsi dibandingkan nilai aslinya. Hal ini dikarenakan pada proses kuantisasi inilah terjadi error paling besar, yang disebabkan proses pembulatan.

2.4.4.3 Run Length Encoding (RLE)

RLE (Run Length Encoding) adalah proses serangkaian simbol yang berurutan dikodekan menjadi suatu kode yang terdiri dari simbol tersebut dan jumlah perulangannya.

Hasil proses transformasi yang dikuantisasi cenderung nol untuk frekuensi tinggi. Untuk melakukan RLE secara efektif, keluaran proses kuantisasi tadi dibaca secara linier dari frekuensi terendah sampai frekuensi tertinggi. Cara yang digunakan adalah *zig - zag scanning*, yaitu membaca secara zig-zag dimulai dari koefisien DC (0,0), kemudian koefisien (0,1), koefisien (1,0) hingga koefisien (NxN). Urutannya dapat dilihat pada gambar 2.16. Pada keluaran proses DCT yang dikuantisasi, nilai nol cenderung berulang secara berurutan, sedangkan nilai lain jarang muncul berurutan. Oleh karena itu RLE akan dilakukan dilakukan pada data bernilai 0



Gambar 2.16 *Zig-zag Scanning*

Pada dekompresi, proses sebaliknya dilakukan, hasil RLE diuraikan kembali, dan dibaca sebagai blok, selanjutnya diumpankan untuk masukan proses dekuantisasi.

2.4.4.4 Variable Length Encoding (VLC)

VLC digunakan untuk mengkodekan simbol dengan kode–kode tertentu yang mempunyai panjang berlainan. Pengkodean ini menggunakan prinsip entropi, yaitu simbol yang sering muncul dikodekan dengan kode yang pendek

dan simbol yang jarang muncul dikodekan dengan kode yang panjang. Dengan demikian, secara keseluruhan bit yang dibutuhkan menjadi lebih sedikit.

Pada kompresi I*ntraframe*, hasil proses RLE dikodekan dengan VLC, maka jumlah bit yang disimpan atau ditransmisikan menjadi lebih kecil.

2.4.4.5 Huffman Coding

Huffman coding adalah suatu metode kompresi untuk mengkompresi serangkaian karakter menjadi kode biner 0 dan 1. pada dasarnya Huffman coding digunakan untuk mengurangi jumlah bit yang dipakai untuk merepresentasikan sebuah karakter.

Untuk membuat pengkodean dengan metode Huffman, langkah – langkahnya sebagai berikut :

1. Menentukan jenis karakter dan frekuensinya
2. Membuat pohon biner berdasarkan frekuensi setiap jenis karakter
3. Dari pohon biner dibuat kode biner dengan cara jika ke kiri di set “0” dan ke kanan di set “1”

Untuk membuat pohon biner dalam metode Huffman langkah – langkahnya sebagai berikut

1. Frekuensi diurutkan dari yang terkecil sampai yang terbesar
2. Setelah diurutkan, nilai frekuensi pertama dijumlahkan dengan frekuensi kedua, hasil penjumlahan disimpan sebagai parent pertama, frekuensi pertama dan kedua sebagai anak dari parent pertama
3. *Up-date* frekuensi dengan parent pertama sebagai frekuensi pertama
4. Kembali ke langkah 1. 2 dan 3 sampai frekuensi habis menjadi anak

5. Tentukan kode untuk setiap anak dengan aturan set “0” ke kiri dan set “1” untuk ke kanan.

2.4.5 Kompresi Interframe

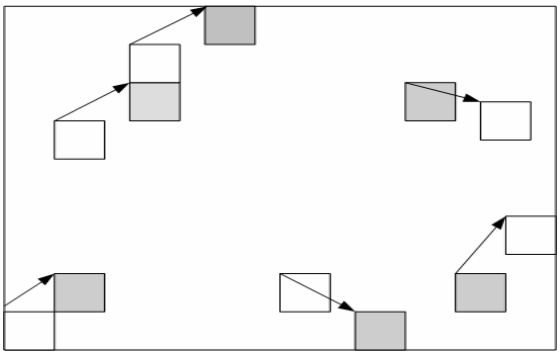
Kompresi *Interframe* dilakukan dengan memanfaatkan redundansi temporal yang terdapat antar frame. Redundansi temporal disebabkan adanya piksel–piksel yang berkorelasi di antara frame-frame tersebut, terutama dikarenakan banyak bagian frame yang tidak berubah dibanding frame sebelum atau sesudahnya.

Proses yang digunakan dalam kompresi *Interframe* adalah estimasi gerak (*motion compensation*) dengan teknik pencocokan blok (*matching block*) untuk mendapatkan vektor gerak (*motion vector*).

2.4.5.1 Estimasi Dan Kompensasi Gerak

Estimasi gerak (*motion estimation)* merupakan teknik kompresi I*nterframe* yang memprediksi sebuah frame dari frame sebelumnya (*reference frame)*, dengan mengestimasi gerakan blok–blok antar frame tersebut. Frame dibagi menjadi blok–blok yang tidak *overlap*. Tiap blok dibandingkan dengan blok–blok berukuran sama, pada frame sebelumnya dengan melakukan pencocokan blok (*block matching)*

Dalam melakukan pencocokan tersebut, lokasi dari blok yang paling mirip atau *match* pada frame referensi (*reference frame*) berbeda dari lokasi blok target (*target block*). Perbedaan relatif posisi ini disebut vektor gerak (*motion vector*), seperti ditunjukkan pada gambar 2.16



Gambar 2.17 *Vektor Gerak*

Jika posisi blok target dan blok yang *match* sama, maka vektor geraknya adalah nol. Vektor gerak inilah yang menunjukkan pergeseran blok – blok antar *frame*. Ketika mengkodekan tiap blok dari *frame* yang diprediksi, vektor gerak yang menunjukkan posisi blok yang *match* pada *frame* referensi, dikodekan pada posisi blok yang *match* pada *frame* referensi, dikodekan pada posisi target blok itu sendiri, maka terjadi kompresi, karena jumlah bit yang diperlukan untuk mengkodekan vektor gerak lebih sedikit daripada untuk mengkodekan suatu blok secara utuh.

Pada dekompresi, dekoder menggunakan vektor gerak untuk menemukan *matching block* pada *frame* referensi dan menyalin *matching block* tersebut ke posisi yang sesuai pada *frame* yang sedang diprediksi. Dengan demikian, suatu *frame* prediksi tersusun atas blok–blok dari *frame* sebelumnya.

Keefektifan teknik kompresi menggunakan kompensasi gerak berbasis blok ini bergantung pada beberapa kondisi berikut

1. Objek bergerak pada bidang datar. Efek dari *zoom* dan rotasi tidak dapat ditangani dengan metode ini
2. Pencahayaan harus konstan dan seragam. Objek yang mengalami perubahan pencahayaan tidak dapat dikenali
3. Objek yang dilewati objek lain tidak dapat ditangani dengan metode ini

Blok–blok yang terletak pada pinggiran frame, diestimasi dan vektor gerak boleh melebihi batas *frame*. Piksel–piksel pinggir digunakan untuk melakukan kompensasi bila vektor gerak menunjuk ke piksel yang berada diluar batas *frame*

2.4.5.2 Block Matching

Pencocokan blok atau *block matching* adalah proses pembandingan blok dengan blok–blok pada *frame* sebelumnya, untuk menemukan *matching block*. *Matching block* merupakan proses yang paling banyak menyita waktu selama *encoding*. *Matching block* cukup dilakukan pada komponen kecerahan (*luminance*) dari *frame*. Hal ini dikarenakan mata manusia lebih peka terhadap kecerahan.

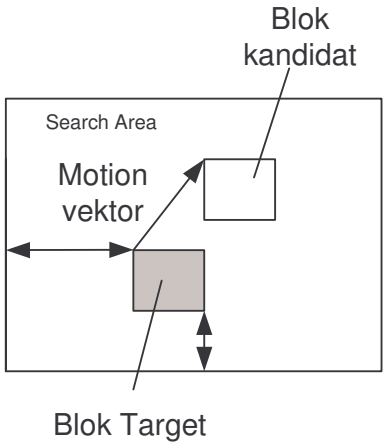
Langkah pertama untuk proses ini adalah membagi *frame* menjadi blok – blok berukuran tertentu. Ukuran blok yang besar mengakibatkan sedikit jumlah vektor gerak yang dihasilkan. Namun, akan sulit menemukan blok yang match dengannya dan *error* yang dihasilkan perbedaan blok relatif besar. Blok yang dibandingkan dengan blok – blok pada *frame* referensi disebut sebagai blok target

Langkah selanjutnya adalah menentukan *search area* atau daerah pencarian pada *frame* referensi. Pencarian blok yang *match* dapat dilakukan pada seluruh daerah *frame* referensi. Namun karena perubahan antar *frame* cenderung

kecil, daerah pencarian cukup dibatasi pada posisi sekitar blok target pada *frame* referensi, maka ditentukan suatu maximum *displacement* yang membatasi jumlah piksel maksimum pada arah vertikal dan horizontal dari posisi blok target pada frame saat ini.

Langkah terakhir adalah menemukan pencocokan blok pada daerah pencarian. Proses ini dilakukan dengan membandingkan target blok dengan blok–blok pada daerah pencarian yang disebut blok kandidat, seperti ditunjukkan pada gambar 2.18. Semakin besar *displacement,* semakin luas daerah pencarian, semakin besar pula peluang untuk mendapatkan pencocokan blok yang bagus. Namun jumlah blok kandidat meningkat secara kuadratik sebanding dengan peningkatan *displacement*, sehingga lebih banyak lagi pembandingan blok yang perlu dilakukan

Pencocokan blok target dengan blok–blok kandidat pada daerah pencarian dilakukan dengan besar *step* tertentu, yang merupakan besar pergeseran dalam pencarian blok. Jumlah blok kandidat, selain ditentukan oleh ukuran daerah pencarian, ditentukan pula oleh besarnya *step*



Gambar 2.18 *Pencocokan Blok*

Setelah diperoleh *matching* blok, maka perbedaan posisinya dengan target blok disebut vektor gerak (*motion vector*), Proses ini menghasilkan vektor gerak pada arah horizontal MVx dan vektor gerak pada arah vertikal MVy.

2.5 *3rd Generation Partnership Project* (3GPP)

IPCC adalah sebuah asosiasi industri internasional yang mendedikasikan diri pada percepatan pengembangan dari teknologi dan layanan VoIP, video over IP, dan lainnya melalui jaringan yang terkonvergensi. 3GPP merupakan kolaborasi antara ETSI (Eropa), ARIB/TTC (Jepang), CCSA (Cina), ATIS (Amerika Utara), dan TTA (Korea Selatan). Tujuan 3GPP adalah untuk membuat spesifikasi sistem mobile phone generasi ketiga yang aplikabel secara global, dalam hubungannya dengan proyek IMT-2000 (International Mobile Telecommunications- 2000) dari International Telecommunication Union (ITU). 3GPP2 yang bertujuan menstandarisasi Code Division Multiple Access – 2000 (CDMA-2000), berbeda dengan 3GPP yang menstandarisasi sistem Global System for Mobile communications – Groupe Special Mobile (GSM) yang kini dikenal sebagai Universal Mobile Telecommunications System (UMTS).

2.6 Bluetooth

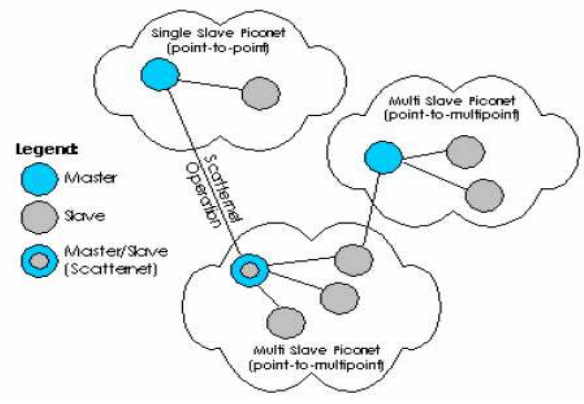
2.6.1 Pengertian

Bluetooth berasal dari nama seorang raja Denmark 940-981 yaitu Harald Blatand (dalam bahasa inggris berarti Bluetooth), yang hidup pada pertengahan abad ke sepuluh. Bluetooth merupakan spesifikasi terbuka (*open specification*) untuk data dan *voice* (suara) yang berbasis pada *short-range radio link*. Bluetooth

didisain untuk beroperasi di lingkungan yang memiliki frekuensi radio yang cukup ribut atau berisik (*noisy radio frequency environment*). Bluetooth menggunakan "*fast acknowledgement*" dan mekanisme "*frequency hopping*" untuk membuat *link* menjadi *robust*. Bluetooth radio modules menghindari *interferensi* dengan sinyal lain dengan cara *hopping* ke frekuensi baru setelah mengirim atau menerima paket. Bluetooth radio beroperasi pada ISM band 2.4 GHz. Bluetooth merupakan *chip radio* yang dirancang untuk menggantikan kabel. Informasi yang biasanya dibawa oleh kabel, dengan bluetooth di*transimisi*kan pada frekuensi tertentu yang kemudian diterima oleh chip bluetooth kemudian informasi tersebut diterima oleh komputer, printer, handphone, dan sebagainya.

2.6.2 Cara Kerja Bluetooth

Bluetooth device dibagi dua, yakni *master device* dan *slave device*. Bluetooth mendukung tiga jenis koneksi dalam membentuk jaringan, yakni: *point-to-point*, *piconet*, dan *scatternet*. Dalam operasinya, bluetooth *device* melakukan koneksi ke dalam *piconet*. *Piconet* terdiri dari sebuah *master device* dan *active slave device* dengan jumlah maksimum *active slave device* adalah 7. Kumpulan dari beberapa piconet yang saling berhubungan disebut *scatternets*. Bluetooth device mempunyai 4 *basic states*, yaitu: *master* (mengontrol sebuah piconet), *active* slave (terhubung dan secara aktif memonitor piconet), *parked* slave (secara logik masih bagian dari *piconet* tetapi *low power*), dan *standby* (tidak terhubung dengan *piconet*). Berikut merupakan gambar sebuah node *master piconet* menjadi *node slave* untuk *piconet* yang lain. Model jaringan Piconet dan Scatternet Bluetooth dapat dilihat pada gambar 2.19.



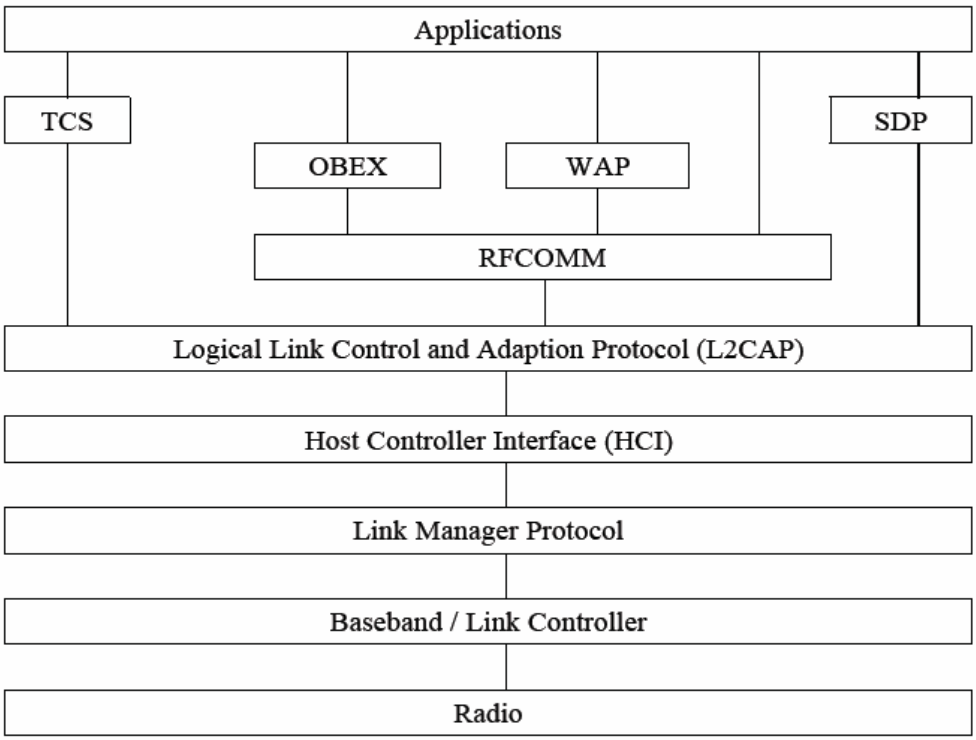
onet dan Scatternet Bluetooth

(Sumber : www.materikuliah.com)

Pada mulanya, bluetooth *device* hanya mengetahui sekitarnya saja dan berada pada *mode standby*, dalam mode ini bluetooth *device* dapat mengurangi kekuatannya sekitar 98%. *Inquiry* merupakan proses bagaimana bluetooth *device* mempelajari bluetooth *device* lainnya yang ada dalam jangkauan. Selama proses *inquiry*, *master device* secara terus-menerus melakukan *broadcast page command* dengan menggunakan *inquiry* ID, yang kemudian setiap *standby device* dalam jangkauannya menerima *inquiry page*. Dengan melakukan persetujuan, node-node ini akan merespon dengan sebuah standar FHS *packet* yang menyediakan BT ID yang unik *clock offset*-nya. Setelah proses *inquiry*, dilanjutkan proses *paging*, yakni akan dibangun hubungan antar device. Setelah sebuah *slave device* membalas *master device*, maka *master device* akan mengirim sebuah FHS *packet* dan menetapkan sebagai *Active Member Address* pada *piconet*.

2.6.3 Protokol Stack

Tidak seperti *wireless* standar yang lain, spesifikasi bluetooth terdiri dari *physical layer*, *link layer*, dan *application layer* yang mendukung data, *voice*, dan *aplikasi content-centric*. Protokol *stack* bluetooth dibagi menjadi dua bagian, yaitu: Bluetooth *controller* yang biasa diimplementasikan pada *hardware*, dan Bluetooth *host* yang berhubungan dengan aplikasi dan layanan-layanan. Gambar 2.20 merupakan proses yang merepresentasikan protokol *stack* bluetooth.



Gambar 2.20 *Bluetooth Protocol Stack*

(Sumber : www.materikuliah.com)

Bluetooth *stack* memiliki beberapa *layer*. HCI merupakan *layer* yang memisahkan *hardware* dari *software* dan mengimplementasikan sebagiannya pada *software* dan *hardware*. Layer di bawah HCI mengimplementasikan *hardware*

sedangkan layer di atas HCI mengimplementasikan *software*. Tabel 2.2 di bawah akan dideskripsikan masing-masing layer.

Tabel 2.3 *Deskripsi layer protokol bluetooth*

Layer	**Deskripsi**
Applications	*Profile* bluetooth yang mengendalikan *developer* dalam menentukan bagaimana sebuah aplikasi harus menggunakan protokol *stack*
Telephony Control System (TCS)	Menyediakan *Telephony service*
Service Discovery Protocol (SDP)	Digunakan untuk mencari *service* yang disediakan pada *remote* bluetooth *device*
WAP and OBEX	Menyediakan *interface* pada bagian layer yang lebih tinggi pada protokol komunikasi yang lain
RFCOMM	Menyediakan keterhubungan melalui bluetooth seperti yang digunakan pada standard serial (COM) port.
L2CAP	Merupakan layer yang menangani *packet segmentation and reassembly* (SAR). *Multiplexes* data dari layer yang lebih tinggi dan merubah merubah antar ukuran packet yang berbeda.
Host Control Interface (HCI)	Menangani komunikasi antara *host* dan bluetooth *module*
Link Manager Protocol	Mengontrol dan konfigurasi *link* kepada *device* lain
Baseband and Link Controller	Mengontrol akses dan mengirim paket data ke media fisik yang melewati bluetooth radio. Layanan-layanan yang

	disediakan pada layer ini antara lain: prosedur inquiry dalam menemukan *device*, fungsionalitas *device* dalam sinkronisasi *clock* dan membangun *link* pada frekuensi *hopping* yang tepat, *error correction*, *data whitening*, *flow control*, fungsi untuk men-*generate* kunci enkripsi
Radio	*Modulate* dan *demodulate* data dari transmisi dan menerima di udara. Bluetooth Radio merupakan *layer* fisik yang menggunakan *frequency hopping spead spectrum* untuk memperkecil interferensi dari device lain yang menggunakan band ini. Bluetooth membagi frekuensi band 2.4 GHz menjadi 79 *channel* pada 1 MHz (dari 2.402 sampai 2.480 GHz) dan frekuensi hop 1600 kali per detik.

2.6.4 Bluetooth Link

Dua type *link* fisik yang didefinisikan versi 1.1 pada spesifikasi bluetooth yaitu: *Synchronous Connection Oriented* (SCO) link dan *Asynchronous ConnectionLess* (ACL) link. SCO dan ACL merupakan bagian dari spesifikasi *baseband*.

Link SCO diperuntukkan untuk *transmisi* audio. Ketika menyeting SCO *link*, ada slot waktu yang dicadangkan untuk *transmisi* data, sehingga harus menyediakan jaminan *Quality of Service* (QOS). Kerusakan dan *error packet* tidak dikirim kembali. Semua link SCO beroperasi pada 64 kbps. Sebuah *master device* dapat memiliki sampai tiga link SCO pada waktu yang sama, semua pada

slave yang sama atau *slave* yang berbeda. *Slave device* dapat memiliki sampai tiga link SCO pada master *device*.

Link ACL diperuntukkan untuk *transmisi* data. ACL *link* menyediakan pengiriman *error-free* pada data yang berarti kerusakan atau *error packet* akan dikirim kembali. *Maximum data rate* pada level aplikasi berkisar 650 kbps untuk link ACL. Sebuah master *device* dapat memiliki sebuah nomor *link* ACL ke nomor *device* yang berbeda, tetapi hanya satu *link* ACL yang dapat *exist* diantara dua *device*. Data biasanya di*tansfer* dari dan untuk *layer Logical Link Control and Adaptation Protocol* (L2CAP) pada bluetooth stack. Para developer aplikasi biasanya menunjuk L2CAP dan RFCOMM ketika membicarakan tentang *link* bluetooth. L2CAP dan RFCOMM merupakan *layer* terpisah pada bluetooth *stack* yang dipercayakan pada link ACL untuk mengirim data.

2.7 Video Streaming

Dalam dunia Internet, *streaming* merupakan sebuah teknologi yang mampu mengkompresi atau menyusutkan ukuran file audio dan video agar mudah ditransfer melalui jaringan Internet. Pen*transfer*an file audio dan video tersebut dilakukan secara terus-menerus. Dari sudut pandang prosesnya, *streaming* berarti sebuah teknologi pengiriman file dari server ke client melalui jaringan *packet-based*. *Streaming* merupakan sebuah metode untuk membuat audio, video, dan multimedia yang lain yang tersedia untuk real-time pada type jaringan yang berbeda. Data pada file *streaming* dibagi-bagi ke dalam beberapa paket kecil yang dikirim ke sebuah aliran secara terus menerus ke komputer *end-user* atau *mobile phone*.

Aplikasi dalam layanan *streaming* dibagi menjadi dua, yaitu "*on-demand*" dan "*live*". Sebagai contoh yang termasuk dalam kategori "*on-demand*" yakni musik dan video, sedangkan yang termasuk "*live*" yakni acara radio atau televisi yang disiarkan secara *broadcasting* pada saat itu juga.

Ide dasar dari video streaming adalah untuk membagi-bagi video asli menjadi beberapa *packet* yang kemudian dikirim secara berurutan, dan memungkinkan *receiver* melakukan *decode* dan *playback* video berdasarkan *packet* tersebut tanpa harus menunggu seluruh video terkirim. Ada beberapa masalah dasar dalam *video streaming*, yakni:

1. *Bandwidth*

   Jika pengirim mengirimkan data lebih cepat dibandingkan dengan ketersediaan *bandwdith* yang ada, maka akan terjadi *congestion* (kemacetan) pada jaringan, *packet loss*, dan kualitas video yang jelek. Tapi apabila pengirim mengirimkan paket data lebih lambat dari *bandwidth* yang tersedia, maka kulitas video yang sampai ke penerima akan kurang optimal. Salah satu cara untuk mengatasinya adalah dengan melakukan estimasi terhadap ketersedian *bandwidth,* yang kemudian mencocokan dengan *bit rate* video yang akan ditransmisikan.

2. *Delay Jitter*

   Delay jitter merupakan masalah yang menyebabkan penerima harus menerima/*decode*/menampilkan *frame* dengan *rate* yang konstan, dan setiap *frame* yang terlambat datang akan menyulitkan rekonstruksi video yang diterima.

3. *Loss Rate*

   Pada jaringan *wireless*, *loss rate* dapat disebabkan oleh *bit error* dan *bursh error. Loss rate* ini dapat menimbulkan kerusakan pada kualitas video hasil rekonstruksi. Untuk mengatasi *loss rate* ini, system *video streaming* dapat didesain dengan fasilitas *error rate control.*