BAB II

LANDASAN TEORI

2.1 Teknologi Selular

Dengan adanya konvergensi atau pemusatan antara teknologi informasi dengan teknologi telekomunikasi, membuat teknologi telah menjadi segalanya bagi manusia, Teknologi komunikasi khususnya teknologi selular telah berkembang pesat di Indonesia, hal ini dimungkinkan dengan penetrasi pasar yang besar terhadap kebutuhan telekomunikasi khususnya yang sifatnya bergerak *(mobile)*, saat ini menurut statistik pengguna selular di Indonesia telah mencapai angka lebih dari 8 juta. Masyarakat Indonesia secara tidak langsung telah menggunakan teknologi informasi khususnya dibidang komunikasi. Mobilitas dan *trend* mungkin yang menjadi faktor utama dari suksesnya teknologi ini, mobilitas merupakan keunggulan utama teknologi seluler dibandingkan dengan telpon tetap (*Fixed Phone*). Setiap pelanggan dapat mengakses dimana saja dan kapan pun ia berada, Komunikasi suara pada saat ini tidak lagi hanya mengandalkan jaringan kabel yang besifat tetap (*fixed line*), selain itu juga komunikasi tidak hanya berupa suara namun dapat juga berupa data dan gambar yang berujung pada multimedia. Saat ini kita mengenal berbagai jenis perangkat komunikasi, seperti perangkat komunikasi tetap *(fixed phone),* komunikasi bergerak terbatas *(fixed mobile phone)* dan komunikasi bergerak selular *(cellular mobile phone).*

Ada beberapa teknologi tanpa kabel untuk teknologi selular ini, diantaranya adalah :

1. **CDMA (*Code Division Multiple Access*),** menggunakan teknologi *spread-spectrum* untuk mengedarkan sinyal informasi yang melalui *bandwith* yang lebar (1,25 MHz). Teknologi ini asalnya dibuat untuk kepentingan militer, menggunakan kode digital yang unik, lebih baik daripada *channel* atau frekuensi RF.

2. **AMPS (*Advanced Mobil Phone Service*),** merupakan teknologi analog yang menggunakan FDMA (*Frequency Division Multiple Access*) untuk membagi-bagi *bandwith radio* yang tersedia kepada sejumlah *channel* diskrit yang tetap. Dengan AMPS, bandwith 1,25 MHz yang diberikan untuk penggunaan selular dibagi menjadi *channel* dengan lebar 30 KHz, masing-masing hanya dapat melayani satu pelanggan (*subscriber*) pada satu waktu. Satu *subscriber* mengakses sebuah *channel* maka tidak satupun *subscriber* lainnya dapat mengakses *channel* tersebut sampai panggilan pertama itu berhenti atau *handed-off* ke *base station* lainnya.

3. **TDMA (*Time Division Multiple Data*),** merupakan sebuah teknologi digital, sama halnya yaitu dengan membagi-bagi spektrum yang tersedia kepada sejumlah *channel* diskrit yang tetap, meskipun masing-masing *channel* merepresentasikan *time slot* yang tetap daripada band frekunesi yang tetap. Sebagai contoh yang mengimplementasikan

teknologi TDMA adalah GSM, yang membagi *carriers* berlebar 2300 KHz menjadi delapan *time-division* channel. GSM (*global sistem for mobile*) adalah teknologi yang berbasis TDMA.

4. **UMTS (*Universal Mobile Telecomunication Access*)** merupakan salah satu sistem generasi ketiga yang dikembangkan di Eropa. dirancang sehingga dapat menyediakan bandwith sebesar 2 Mbits/s. Layanan yang dapat diberikan UMTS diupayakan dapat memenuhi permintaan pemakai dimanapun berada, artinya UMTS diharapkan dapat melayani area yang seluas mungkin, jika tidak ada *cell* UMTS pada suatu daerah dapat di *route*-kan melalui satelit.

2.1.1 Perkembangan Aplikasi Pada Telepon Selular (Ponsel)

Ponsel pada saat ini tidak hanya digunakan untuk komunikasi suara, tetapi juga dapat dapat digunakan untuk membeli tiket, mencari berita, perbankan dan menjelajah *Internet*, bahkan untuk mengoperasikan peralatan tertentu. Boleh dikatakan suatu saat nanti kemampuan telepon selular akan dapat menyamai kemampuan komputer yang ada saat ini. Dengan kemampuan CPU dan *memory* yang makin meningkat, semakin dimungkinkan bagi *developer* untuk mengembangkan aplikasi yang bersifat *multimedia*.

Suatu saat nanti telepon selular betul-betul merupakan suatu peralatan multifungsi yang selain untuk komunikasi juga untuk mengendalikan peralatan,

mengkoordinasi peralatan dan lain sebagainya. Semua perkembangan ini tentu tidak terlepas dari teknologi informasi dan teknologi komunikasi yang sangat erat hubungannya dengan jaringan komputer dan komunikasi data.

2.2 Jaringan Komputer

Jaringan komputer (*computer network*) adalah hubungan dua buah simpul atau *node* (umumnya berupa komputer) atau lebih ditujukan untuk melakukan pertukaran data atau untuk berbagi pemakaian perangkat lunak, perangkat keras dan bahkan berbagi kekuatan proses. Berikut merupakan manfaat penggunaan jaringan komputer :

1. Berbagi perangkat keras.

   Perangkat keras dapat digunakan oleh sejumlah komputer tanpa perlu melepas dan memasang kembali, cukup dipasang ke komputer atau dihubungkan ke suatu peralatan khusus dan semua komputer dapat mengaksesnya.

2. Berbagi program atau data.

   Program atau data dimungkinkan untuk disimpan pada sebuah komputer yang bertindak sebagai *server*. Cara ini memungkinkan perusahaan membeli sebuah perangkat lunak dan dipasang di *server*, kemudian orang yang memerlukannya dapat mengakses program tersebut.

3. Mendukung kecepatan komunikasi.

   Dengan adanya dukungan jaringan komputer, komunikasi dapat dilakukan lebih cepat. Misalnya saja rapat lewat telekonferensi.

4. Memudahkan pengaksesan informasi.

   Jaringan komputer memudahkan pengaksesan informasi. Seseorang dapat bepergian ke mana saja dan tetap mengakses data yang terdapat pada *server* ketika ia membutuhkannya.

2.2.1 Tipe – Tipe Jaringan Komputer

Jaringan komputer jika dilihat dari rentang geografisnya maka akan dibagi menjadi tiga jenis :

2.2.1.1 *Local Area network (LAN)*

LAN digunakan untuk menghubungkan komputer yang berada di dalam suatu area yang kecil, misalnya di dalam suatu gedung perkantoran atau kampus. Jarak antar komputer yang dihubungkannya bisa mencapai 5 sampai 10 km. Suatu LAN biasanya bekerja pada kecepatan mulai 10 Mbps sampai 100 Mbps. LAN menjadi populer karena memungkinkan banyak pengguna untuk memakai sumber daya secara bersama-sama. Contoh dari sumber daya yang dapat digunakan itu misalnya suatu *mainframe*, *file server*, *printer*, dan sebagainya. LAN pada umumnya menggunakan media transmisi berupa kabel. Namun juga ada yang nirkabel yang disebut sebagai *Wireless* LAN (WLAN).

Menurut tipenya, LAN terdiri dari :

1. *Client / server*, adalah suatu model jaringan yang memiliki *client* dan *server*. *Client* adalah komputer yang meminta layanan sedangkan *server* adalah komputer yang bertindak untuk melayani permintaan *client*. Fungsi *server* sendiri sebenarnya merupakan perangkat lunak yang dijalankan pada perangkat keras yang pada umumnya adalah komputer. Beberapa contoh fungsi server adalah sebagai berikut :

    a. *File server*, *server* yang menangani berkas yang dapat diakses oleh *client*.

    b. *Print server*, *server* yang bertindak sebagai pengontrol *printer* yang dapat digunakan oleh *client*.

    c. *Web server*, server yang menangani halaman – halaman *Web* yang dapat diakses oleh *browser*.

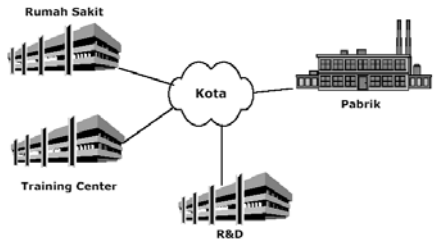
    d. *Mail server*, *server* yang menangani surat elektonik.

2. *Peer-to-Peer*, adalah model jaringan yang memberikan kedudukan yang sama terhadap semua komputer. Tak ada yang bertindak sebagai *server* atau *client* secara eksplisit. Pada model ini komputer dapat berhubungan langsung tanpa bergantung pada *server*, jaringan model ini tidak efektif untuk jaringan skala besar.

3. *Hybrid*, jaringan ini merupakan gabungan dari model *Peer-to-peer* dan *Client / server*.

2.2.1.2 *Metropolitan Area Network* (MAN).

MAN merupakan suatu jaringan yang cakupannya meliputi suatu kota. MAN menghubungkan LAN-LAN yang lokasinya berjauhan. Jangkauan MAN bisa mencapai 10 km sampai beberapa ratus km dengan kecepatan berkisar antara 1,5 Mbps sampai 150 Mbps. Jaringan seperti ini biasanya menggunakan media transmisi dengan gelombang mikro (*microwave*) atau gelombang radio. Namun, ada juga yang menggunakan jalur sewa (*leased line*). Gambar 2.1 berikut adalah suatu ilustrasi suatu MAN.

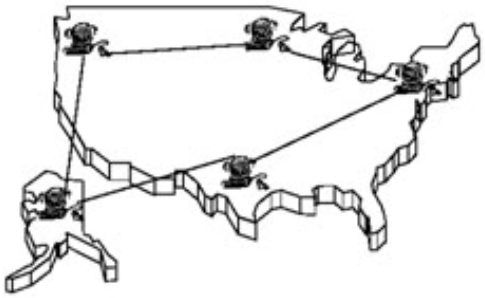


Gambar 2.1 Contoh MAN

2.2.1.3 *Wide Area Network* (WAN).

WAN dirancang untuk menghubungkan komputer-komputer yang terletak pada suatu cakupan geografis yang luas, seperti hubungan dari satu kota ke kota lain di dalam suatu negara. Cakupan WAN bisa meliputi 100 km sampai 1.000 km, dan kecepatan antar kota bisa bervariasi antara 1,5 Mbps sampai 2,4 Gbps. Dalam WAN, biaya untuk peralatan transmisi sangat tinggi, dan biasanya jaringan WAN dimiliki dan dioperasikan sebagai suatu jaringan publik. Para pelaku bisnis

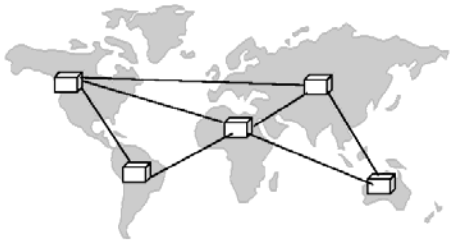
dapat menyewa sistem transmisi tersebut untuk menghubungkan kantor-kantor cabang yang dimilikinya. Gambar 2.2 berikut adalah suatu ilustrasi dari salah satu contoh WAN.



Gambar 2.2 Contoh WAN

2.2.1.4 *Global Area Network* (GAN).

GAN merupakan suatu jaringan yang menghubungkan negara-negara di seluruh dunia. Kecepatan GAN bervariasi mulai dari 1,5 Mbps sampai dengan 100 Gbps dan cakupannya mencakupi ribuan kilometer. Contoh yang sangat baik dari GAN ini adalah *Internet.* Gambar 2.3 berikut memperlihatkan contoh suatu GAN.



Gambar 2.3 Contoh GAN

Jaringan-jaringan tersebut diatas seperti LAN, MAN, WAN, dan GAN dapat melakukan interaksi antara jaringan satu dengan jaringan yang lain dengan *interface* yang digunakan oleh jaringan-jaringan tersebut adalah *interface* yang sudah ditentukan oleh standar internasional maupun regional, standar-standar ini memungkinkan peralatan-peralatan yang berasal dari vendor yang berbeda dapat dihubungkan satu sama lain.

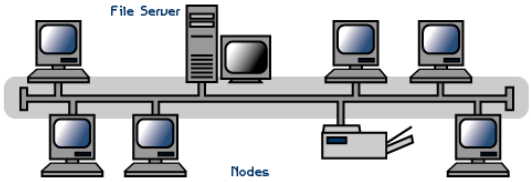
2.2.2 Arsitektur Jaringan Komputer

Susunan atau arsitektur jaringan komputer secara fisik dalam suatu jaringan disebut dengan topologi jaringan, merujuk pada konfigurasi kabel, komputer, dan perangkat lainnya. Sedangkan arsitektur logik atau biasa disebut protokol jaringan adalah cara kerja atau metode yang digunakan untuk mengantarkan informasi diantara *host / workstation*.

Dibawah ini adalah beberapa penjelasan dari topologi fisik yang digunakan di dalam jaringan komputer.

2.2.2.1 *Linear Bus* (Garis Lurus)

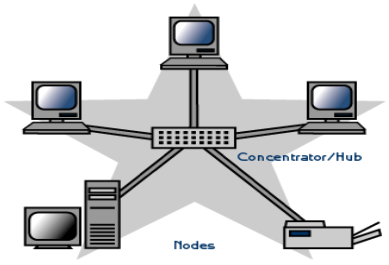
Sebuah topologi *Linear Bus* / garis lurus terdiri dari satu jalur kabel utama dimana pada masing-masing ujungnya diberikan sebuah terminator. Semua *nodes* pada jaringan (*File server*, *workstation*, dan perangkat lainnya) terkoneksi pada sebuah kabel utama. Jaringan-jaringan *Ethernet* dan *Localtalk* menggunakan topologi linear ini. Gambar 2.4 berikut adalah ilustrasi dari topologo *Linear Bus.*



Gambar 2.4 Topologi *Linear Bus*

2.2.2.2 *Star* (Bintang)

Topologi model ini didesain dimana setiap *node* (*file server*, *workstation*, dan perangkat lainnya) terkoneksi ke jaringan melewati sebuah hub atau *concentrator*. Data yang terkirim ke jaringan akan melewati hub/ *concentrator* sebelum melanjutkan ke tempat tujuannya. Hub ataupun *concentrator* akan mengatur dan mengontrol keseluruhan fungsi jaringan. Selain itu juga bisa bertindak sebagai *repeater*/penguat aliran data. Konfigurasi pada jaringan model ini menggunakan kabel *Tweisted pair*, dan dapat digunakan bersama kabel koaksial atau kabel fiber optik. Gambar 2.5 berikut adalah ilustrasi dari Topologi *Star* (Bintang)

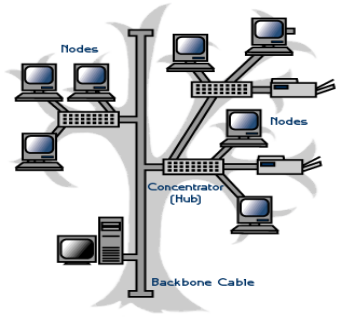


Gambar 2.5 Topologi *Star* (Bintang)

2.2.2.3 *Tree* (Pohon)

Topologi ini merupakan perpaduan antara topologi Garis Lurus dan Bintang, yang mana terdiri dari kelompok-kelompok dari *workstation* konfigurasi

bintang yang terkoneksi ke kabel utama yang menggunakan topologi garis gurus/ *linear bus*. Topologi ini memungkinkan untuk pengembangan jaringan yang telah ada, dan memungkinkan sebuah perusahaan mengkonfigurasi jaringan sesuai dengan kebutuhannya. Gambar 2.6 berikut adalah ilustrasi dari topologi *tree*.



Gambar 2.6 Topologi *Tree* (Pohon)

2.2.3 Protokol Jaringan Komputer

Sedangkan arsitektur logik atau biasa disebut protokol jaringan sebagaimana dijelaskan sebelumnya, adalah cara kerja atau metode yang digunakan untuk mengantarkan informasi diantara *workstation.* Berikut adalah beberapa protokol jaringan yang biasa dipakai pada jaringan lokal atau *Local Area Network* (LAN).

1. *Ethernet*

   Protocol *Ethernet* sejauh ini adalah yang paling banyak digunakan, *Ethernet* menggunakan metode akses yang disebut CSMA/CD (*Carrier Sense Multiple Access/Collision Detection*). Sistem ini menjelaskan bahwa setiap komputer memperhatikan ke dalam kabel dari *network* sebelum mengirimkan sesuatu kedalamnya. Jika dalam

jaringan tidak ada aktifitas atau bersih komputer akan mentransmisikan data, jika ada transmisi lain di dalam kabel, komputer akan menunggu dan akan mencoba kembali transmisi jika jaringan telah bersih. kadangkala dua buah komputer melakukan taransmisi pada saat yang sama, ketika hal ini terjadi, masing-masing komputer akan mundur dan akan menunggu kesempatan secara acak untuk mentransimisikan data kembali. Metode ini dikenal dengan koalisi, dan tidak akan berpengaruh pada kecepatan transmisi dari *network*. Protokol *Ethernet* dapat digunakan untuk pada model jaringan Garis lurus, Bintang, atau Pohon . Data dapat ditransmisikan melewati kabel *twisted pair*, Koaksial, ataupun kabel *fiber optic* pada kecepatan 10 Mbps.

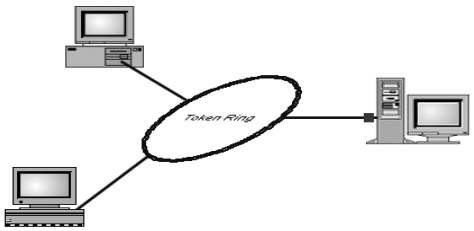
2. *Local Talk*

   Local Talk adalah sebuh protokol *network* yang di kembangkan oleh *Apple Computer Inc*. untuk mesin -mesin komputer *Macintosh*. Metode yang digunakan oleh *LocalTalk* adalah CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*), hampir sama dengan CSMA/CD, a*dapter LocalTalk* dan *cable twisted pair* khusus dapat digunakan untuk menghubungkan beberapa komputer melewati *serial port*. *Sistem Operasi Macintosh* memungkinkan koneksi secara jaringan *peer-to-peer* tanpa membutuhkan tambahan aplikasi khusus. Protokol *LocalTalk* dapat digunakan untuk model jaringan Garis Lurus ,Bintang , ataupun model Pohon dengan menggunakan kabel

twisted pair . Kekurangan yang paling mencolok yaitu kecepatan transmisinya yang hanya 230 Kbps.

3. *Token Ring*

Protokol Token di kembangkan oleh IBM pada pertengahan tahun 1980. Metode Aksesnya melalui lewatnya sebuah *token* dalam sebuah lingkaran seperti cincin . Dalam lingkaran *token*, komputer-komputer di hubungkan satu dengan yang lainnya seperti sebuah cincin. Sebuah sinyal *token* bergerak berputar dalam sebuah lingkaran (cincin) dalam sebuah jaringan dan bergerak dari sebuah komputer-menuju ke komputer berikutnya, jika pada persinggahan di salah satu komputer ternyata ada data yang ingin di transmisikan, *token* akan mengangkutnya ke tempat dimana data itu ingin ditujukan, *token* bergerak terus untuk saling mengkoneksikan diantara masing-masing komputer. Gambar 2.7 berikut adalah ilustrasi dari sebuah *token ring*



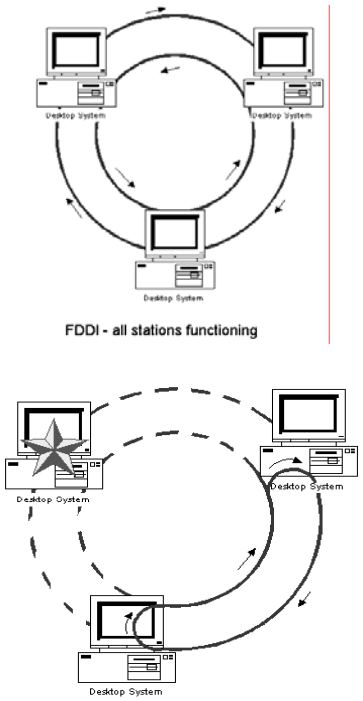
Gambar 2.7 *Token Ring*

Protokol *Token Ring* membutuhkan model jaringan Bintang dengan menggunakan kabel *twisted pair* atau kabel *fiber optic* dan dapat melakukan keceptan transmisi 4 Mbps atau 16 Mbps. Sejalan dengan

perkembangan *Ethernet*, penggunaan *Token Ring* makin berkurang sampai saat ini.

4. *Fiber Distributed Data Interface* (FDDI)

   Adalah sebuah protokol jaringan yang menghubungkan antara dua atau lebih jaringan bahkan pada jarak yang jauh. FDDI menggunakan dua buah topologi *ring* secara fisik. Proses transmisi baiasanya menggunakan satu buah *ring*, namun jika ada masalah ditemukan akan secara otomatis menggunakan *ring* yang kedua. Gambar 2.8 berikut adalah ilustrasi dari sebuah *token* FDDI



Gambar 2.8 FDDI

Sebuah keuntungan dari FDDI adalah kecepatan dengan menggunakan kabel *fiber optic* pada kecepatan 100 Mbps.

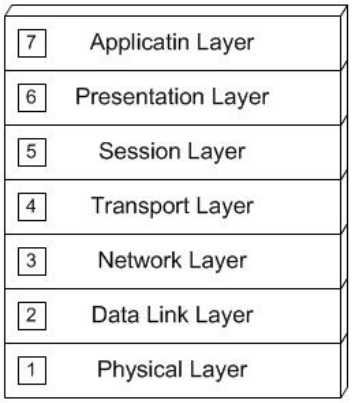
5. *Asynchronous Transfer Mode* (ATM)

   yaitu sebuah protokol jaringan yang mentransmisikan data pada kecepatan 155 Mbps atau lebih. ATM mentransmisikan data kedalam satu paket dimana pada protokol yang lain men-*transfer* pada besar-kecilnya paket. ATM mendukung variasi media seperti *video, CD-audio,* dan gambar. ATM bekerja pada model topologi Bintang , dengan menggunakan Kabel *fiber optic* ataupun kabel *twisted pair* . ATM pada umumnya digunakan untuk menghubungkan dua atau lebih LAN . selain itu juga banyak dipakai oleh *Internet Service Providers* (ISP) untuk meningkatkan kecepatan akses *Internet* untuk klien mereka.

Dalam tiap topologi memiliki perbedaan cara dalam pengiriman data. Hal tersebut merupakan karakteristik dari tiap-tiap topologi jaringan. Berbeda dengan pertukaran data antar sistem yang berbeda dalam jaringan, karena memang memiliki sistem yang berbeda tetapi dalam jaringan yang sama, jadi untuk menyampaikan suatu data ke sistem lain harus menggunakan tatacara yang biasa digunakan dalam pelaksanaan pertukaran data antara dua buah sistem dalam jaringan. Tatacara ini sering disebut dengan protokol komunikasi.

2.2.4 Protokol Komunikasi

Standar protokol komunikasi yang terkenal adalah OSI (*Open System Interconnecting*) yang ditentukan oleh ISO (*International Standard Organization*). Pada tahun 1978 ISO menyusun standart OSI yang dapat dipakai untuk mengembangkan sistem terbuka dan sebagai referensi dengan sistem komunikasi data yang berbeda. Sistem jaringan yang dirancang menurut kerangka kerja dan spesifikasi OSI maka akan memiliki metode komunikasi yang saling kompatibel. Standar OSI ini mendefinisikan tujuh lapisan.



Gambar 2.9 Lapisan pada Model OSI

Pada gambar 2.9 diatas dapat dilihat terdapat 7 lapisan (*layer*) yang bekerja dari mulai lapisan teratas menuju lapisan terbawah sesuai urutan. ketujuh lapisan tersebut disusun berdasarkan lima prinsip yang harus diikuti untuk menentukan *layer* dalam komunikasi, yaitu :

a. *Layer* dibuat jika ketika diperlukan pemisahan *level* yang secara teori diperlukan.

b. Masing-masing *layer* memiliki fungsi yang jelas.

c. Setiap fungsi dari masing-masing *layer* telah ditentukan agar sesuai dengan standart protokol secara internasional.

d. Batas kedua *layer* telah ditentukan untuk mengurangi informasi menerobos antarmuka *layer.*

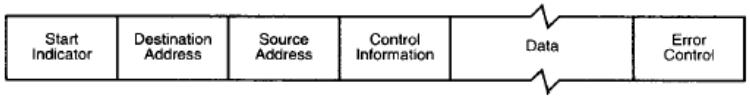
e. Setiap *layer* ditentukan dengan jelas fungsinya, tetapi jumlah *layer* sebaiknya sekecil mungkin untuk menghindari arsitektur yang luas.

Berikut adalah fungsi dari masing masing lapisan model OSI :

1. *Physical Layer*, dipergunakan untuk mengtransmisikan data per *bit* melewati saluran komunikasi. Susunan *bit* tersebut mungkin mewakili pengiriman *file* atau *record* dari *file database*, *physical-layer* mengabaikan arti (melupakan) susunan bit tersebut, untuk selanjutnya akan dikodekan menjadi digit **1** dan **0** atau menjadi bentuk analog. *Physical-layer* menangani proses mekanis, elektris dan prosedur antar muka dalam media fisik (saluran transmisi, *driver*, sensor, pencatu, dll.).

2. *Data Link Layer* , *Data-link-layer* dibentuk berdasarkan kemampuan transmisi dari *physical layer*. Susunan *bit* yang akan dikirim atau diterima dikumpulkan dalam kelompok yang disebut *frame*. Dalam konteks LAN, *frame* dapat berarti *token-ring* atau *ethernet frame*. Awal dan akhir *frame* di tandai dengan susunan *bit* khusus, sehingga *frame* tersusun dalam susunan bit yang terdiri

atas *address-field, control-field, data-field,* dan *errorcontrol-field*, yang masing-masing memiliki fungsi tertentu. Gambar 2.10 berikut ini adalah ilustrasi bentuk frame Data Link Layer.



Gambar 2.10 Bentuk *frame Data Link Layer*

Address-field berisi alamat node pengirim *(source)* dan penerima *(destination). Control-field* dipakai untuk menandai adanya perbedaan jenis dari *data-link-frame,* termasuk frame data dan frame yang dipakai untuk mengatur *data-link-channel*. *Data-field* berisi data asli yang dikirimkan bersama dalam frame. *Error-control-field* dipakai untuk mendeteksi adanya pada *data-link-frame*. *Data-link-layer* merupakan layer pertama yang terlihat memiliki perhatian kepada pendeteksian *error*. *Error-control-field* umumnya berisi hasil pengecekan secara hardware yang dipergunakan untuk mendeteksi adanya *error.*

3. *Network Layer*, dibentuk berdasarkan hubungan *node-to-node* yang disediakan oleh *data-link-layer*. Pelayanan *data-link* secara *node-to-node* menuju jaringan akan menjadi meningkat dengan adanya *layer* ini, sehingga *data-link-layer* dapat menambah pelayanan untuk rute lintasan sejumlah *packet* (bagian dari informasi yang berada pada *network-layer*) diantara beberapa *node* dihubungkan

melewati jaringan yang kompleks secara berubah-ubah. Disamping melayani proses *routing*, *network-layer* membantu menghilangkan kemacetan dengan cara mengatur aliran data. Disamping itu *network-layer* dapat membuat kemungkinan agar dua jaringan dapat dihubungkan menerapkan *uniform-addresing-mecanism* (suatu mekanisme untuk pengalamatan sejenis). Sebagai contoh jaringan lokal *Ethernet* dan *Token-ring* memiliki alamat *datalink* yang berbeda tipenya, untuk menghubungkan dua jaringan tersebut maka *diperlukan uniform-addressing-mechanism* yang dapat dimengerti oleh *Ethernet* maupun *Token-ring*. Untuk jaringan yang berbasis *Novel-Netware* maka digunakan *internet-packet-exchange* (IPX) sebagai protokol *network-layer*, sedangkan jaringan berbasis TCP/IP digunakan *internet-protocol* (IP).

4. *Transport Layer*, *Transport-layer* menyediakan perbaikan untuk melayani *network- layer*. Lapisan ini membantu dalam meyakinkan pengiriman data dapat diandalkan dan menggabungkan data yang telah dikirim dari ujung ke ujung. Pada *layer* ini merupakan kesempatan terakhir untuk mengatasi *error*, tetapi pada kenyataannya *transport-layer* menyediakan pengiriman yang bebas error. *Transport-layer* adalah *layer* menengah dalam model OSI, tiga layer dibawahnya menyatakan bagian *subnet* dari model jaringan, sedangkan tiga layer diatasnya biasanya dipergunakan untuk proses *softwering* pada *node*. Karena adanya *multiplexing*,

beberapa elemen *software* atau pada OSI disebut dengan *protocol-entity* untuk membagi alamat (*address*) *network-layer* yang sama. Untuk mengidentifikasi setiap elemen *software* didalam *transport-layer* diperlukan bentuk umum dalam pengalamatan, yang disebut dengan *transport-address* yang biasanya merupakan kombinasi alamat *network-address* dan nomor *transport* dari *service-access-point*. Kadangkala untuk mengidentifikasi alamat *tranport* disebut dengan *socket* atau *port-number*.

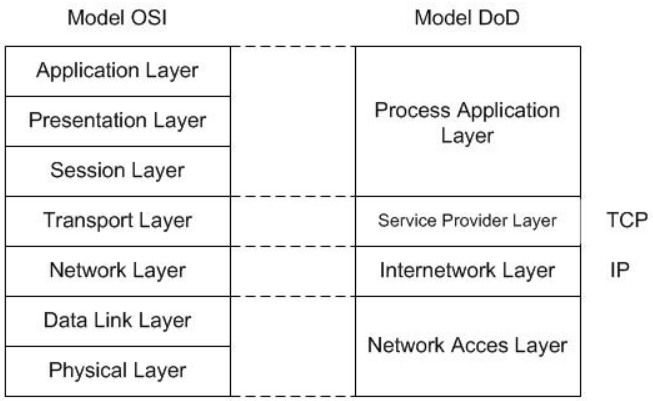
5. *Session Layer*, *Session-layer* menggunakan *transport-layer* untuk menyediakan perbaikan layanan pada *session*. Contoh dari *session* termasuk pencatatan pada *host* saat *user* sedang pada jaringan atau *session* sedang menyusun untuk kegunaan *transfer file*. *Session* pada saat tersambung secara umum dapat menyediakan komunikasi dua arah *(full-duplex)*, tetapi pada beberapa aplikasi kadangkala hanya memerlukan komunikasi satu arah *(half-duplex).*

6. *Presentation Layer*, mengatur cara untuk mewakili data, beberapa cara untuk mewakili data untuk *file* teks dan angka adalah menggunakan ASCII atau EBDIC. Jika dua sisi yang terlibat dalam komunikasi menggunakan metoda berbeda untuk mewakili data, maka tidak akan saling mengerti satu sama lain. *Presentation-layer* menggunakan *syntax* dan *semantic* yang umum untuk mewakili data. Jika semua node memakai dan mengerti bahasa yang umum

dipakai maka kesalahpahaman dalam menginterprestasikan data dapat dihilangkan. Contoh bahasa yang umum dipakai sesuai rekomendasi OSI adalah *abstract-syntax-representaion.*

7. *Application layer*, berisi beberapa protokol dan fungsi yang diperlukan oleh pemakai aplikasi untuk melakukan jenis komunikasi yang diinginkan.

Jenis protokol lainnya adalah TCP/IP *(Transmission Control Protocol/ Internet Protoco*l) berkedudukan sebagai suatu arsitektur yang kerap kali disebut arsitektur *internet*. Karena TCP/IP memiliki pertalian dengan *Departement of Defense*, sehingga lapisan protokol TCP/IP disebut dengan model DoD. Adapun perbandingannya model OSI dan TCP/IP dapat dilihat pada gambar 2.11 berikut ini



Gambar 2.11 Model OSI dan Model DoD

TCP/IP merupakan dua buah protokol yang memiliki fungsi masing – masing. IP merupakan protokol utama pada model OSI pada lapisan jaringan, sedangkan TCP merupakan salah satu protokol yang paling umum digunakan. Selain itu TCP/IP dikenal pula sebagai suatu paket yang berisi sejumlah program.

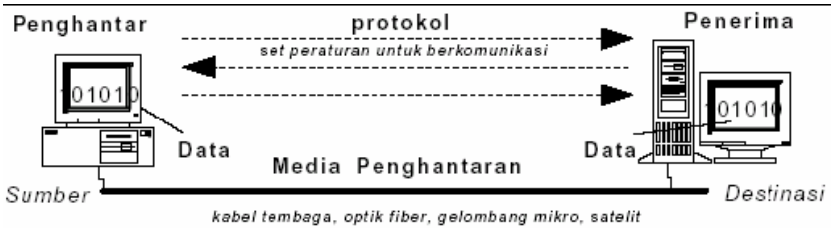
2.3 Komunikasi Data

Komunikasi data adalah merupakan bagian dari telekomunikasi yang secara khusus berkenaan dengan transmisi atau pemindahan data dan informasi diantara komputer komputer dan piranti-piranti yang lain dalam bentuk digital yang dikirimkan melalui media komunikasi data. Komunikasi data merupakan bagian vital dari suatu masyarakat informasi karena sistem ini menyediakan infrastruktur yang memungkinkan komputer-komputer dapat berkomunikasi satu sama lain.

2.3.1 Komponen Komunikasi Data

Terdapat beberapa komponen penting dalam komunikasi data, komponen-komponen tersebut adalah :

1. Sumber / pengirim, adalah piranti yang mengirimkan data.
2. Penerima, adalah piranti yang menerima data.
3. Data, informasi yang akan dipindahkan atau dikirimkan.
4. Media Transmisi, adalah media atau saluran yang akan digunakan untk pengiriman data.
5. Protokol, adalah aturan-aturan yang berfungsi untuk menyelaraskan hubungan.



Gambar 2.12 Komunikasi Data

Gambar 2.12 diatas memperlihatkan bagaimana komponen-komponen komunikasi data saling bekerja sama sehingga terbentuk sebuah komunikasi data diantara dua computer melalui media transmisi.

2.3.2 Media Transmisi

Dalam perkembangannya, media transmisi dalam komunikasi data khususnya komunikasi data antara *mobile device* dengan komputer *desktop* tidak hanya menggunakan media kabel (*wire*), melainkan terdapat pula media transmisi tanpa kabel yang biasa disebut *wireless.*

2.3.2.1 Media Transmisi Kabel (*Wire*)

Sebagai alat komunikasi antara mobile device dengan computer, komunikasi terjadi menggunakan kabel data. Kabel data ini biasanya sudah termasuk dengan paket atau *accessories* sesuai jenis dan seri *mobile device* tersebut. Tipe koneksi kabel data tersebut dengan *Unuversal Serial Bus* (usb) atau menggunakan *communication port.*

2.3.2.2 Media Transmisi Tanpa Kabel (*Wireless*)

Penggunaan media kabel untuk berkomunikasi tidak bersifat *mobile* dan fleksibilitas, serta tidak efisien. Beberapa media komunikasi *wireless* adalah :

a. Infrared

Teknologi *Infrared* adalah teknologi pertama dan bersifat umum, seperti *remote* televisi. Prinsip kerjanya berupa prosesor kecil pada *remote* yang akan melakukan penerjemahan penekanan tombol menjadi instruksi bahasa mesin (bilangan biner) yang dikirimkan melalui *infrared* ke televisi, lalu data diubah kembali menjadi instruksi yang dikenali oleh televisi.

Dalam *mobile device* media transmisi yang memang sering didapat yaitu *infrared*. Terutama bagi beberapa ponsel seri lama yang belum memiliki media transmisi jenis baru.

IrDA (*Infrared Data Association*) memiliki panjang gelombang sekitar 875 nm. Sinar yang dihasilkan dan dipancarkan didapat dari sebuah lampu *LED.* Terdapat dua versi yang telah dikembangkan yaitu versi 1.0 yang memiliki kecepatan dari 0,576 hingga 115,2 kbps. Dan versi kedua adalah versi 2.0 memiliki kecepatan 0,576 hingga 1,152 Mbps.

b. *Bluetooth*

Transmisi *Bluetooth* bias mencapai 10 meter dan tidak terhalang fleksibilitas media, berbeda dengan media lainnya seperti *infrared* atau *Wi-Fi*, *Bluetooth* memungkinkan koneksi antar piranti elektronik apa saja dan bukan hanya antar komputer. *Bluetooth* dapat dibuat membentuk *Personal Area Network* (PAN) antar perangkat seperti komputer, ponsel, Personal Digital Assistant (PDA), kamera, perangkat *audio-video*, bahkan sampai peralatan rumah tanga Hampir seluruh *mobile device* generasi terkini sudah memiliki media transmisi *Bluetooth*, hal itu dikarenakan kecepatan dan kemudahan *transfer* data antar *mobile device* sangat mudah.

c. Wi-Fi (*Wireless Fidelity*)

Wi-Fi bekerja pada frekuensi yang sama dengan *Bluetooth*, yaitu pada 2,4 Ghz. Namun bedanya *Bluetooth* menggunakan *spread spectrum Frequency Hopping (SSFH)* sedangkan Wi-Fi menggunakan *Direct Sequence Spread Spectrum* (DSSS).

Intinya *spread* pada *Wi-Fi* akan lebih stabil dan tentunya akan lebih cepat dibandingkan dengan *Bluetooth*. Namun *W-Fi* memiliki kelemahan yang sangat menggangu seperti masalah keamanan pengiriman data dan rentan terhadap konflik dengan perangkat lain dalam waktu yang bersamaan.

d. GPRS (*General Packet Radio Access*)

GPRS merupakan standar konektivitas dalam setiap *mobile device*. GPRS menjadi media koneksi *download* dan *upload* data dengan *internet.*

2.4 *Bluetooth*

Bluetooth adalah sebuah teknologi komunikasi nirkabel (*wireless*) yang beroperasi dalam pita frekuensi 2,4 GHz *unlicensed* ISM *(Industrial, Scientific and Medical*) dengan menggunakan sebuah *frequency hopping tranceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real-time* antara *host-host Bluetooth* dengan jarak jangkauan layanan yang terbatas (sekitar 10 meter). *Bluetooth* sendiri dapat berupa *card* yang bentuk dan fungsinya hampir sama dengan *card* yang digunakan untuk *wireless local area network* (WLAN) dimana menggunakan frekuensi radio standar IEEE 802.11, hanya saja pada *Bluetooth* mempunyai jangkauan jarak layanan yang lebih pendek dan kemampuan transfer data yang lebih rendah.

Bluetooth menggunakan salah satu dari dua jenis frekuensi *Spread Specturm Radio* yang digunakan untuk kebutuhan *wireless*. Jenis frekuensi yang digunakan adalah FHSS (*Frequency Hopping Spread Spedtrum*), sedangkan yang satu lagi yaitu DSSS (*Direct Sequence Spread Spectrum*) digunakan oleh IEEE802.11xxx atau lebih dikenal Wi-Fi. *Transceiver* yang digunakan oleh *Bluetooth* bekerja pada frekuensi 2,4 GHz *unlicensed* ISM (*Industrial, Scientific, and Medical*).

Protokol *Bluetooth* menggunakan sebuah kombinasi antara *circuit switching* dan *packet switching*. *Bluetooth* dapat mendukung sebuah kanal data asinkron, tiga kanal suara sinkron simultan atau sebuah kanal dimana secara bersamaan mendukung layanan data asinkron dan suara sinkron. Setiap kanal suara mendukung sebuah kanal suara sinkron 64 kb/s. Kanal asinkron dapat mendukung kecepatan maksimal 723,2 kb/s asimetris, dimana untuk arah sebaliknya dapat mendukung sampai dengan kecepatan 57,6 kb/s. Sedangkan untuk mode simetris dapat mendukung sampai dengan kecepatan 433,9 kb/s.

Sebuah perangkat yang memiliki teknologi *wireless Bluetooth* akan mempunyai kemampuan untuk melakukan pertukaran informasi dengan jarak jangkauan sampai dengan 10 meter ($\pm$30 feet), bahkan untuk daya kelas 1 bisa sampai pada jarak 100 meter.

2.4.1 *Bluetooth Protocol Stack*

Bluetooth Protocol Stack adalah suatu perangkat lunak atau komponen *firmware* yang mempunyai akses langsung terhadap perangkat *Bluetooth*. *Bluetooth protocol stack* mampu mengontrol beberapa hal seperti setingan perangkat, parameter-parameter komunikasi, dan aras daya untuk alat *Bluetooth*. *Bluetooth stack* terdiri dari beberapa *layer* dimana masing-masing *layer* mempunyai tugas yang spesifik. Beberapa *layer* pada *Bluetooth stack* yang paling sering digunakan pada hampir setiap perangkat *Bluetooth* diantaranya :

a. *Host Protocol Interface (HCI)*

   Layer ini adalah sebagai antarmuka antara radio dan komputer *Host.*

b. *Logical Link Controller Adaptation Protocol (L2CAP)*

   Layer ini adalah *multiplexer* dari semua data yang melalui unit, *layer* ini mempunyai akses langsung kepada HCI.

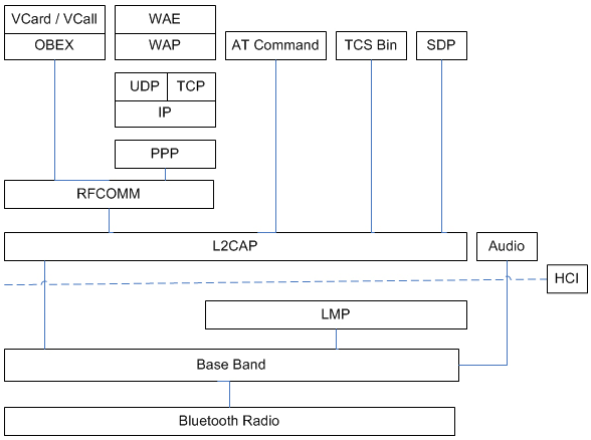
c. *Service Discovery Protocol (SDP)*

   Layer ini digunakan untuk menemukan service yang tersedia untuk perangkat *Bluetooth* yang terhubung.

d. *RFCOMM*, adalah lapisan protokol yang dipakai oleh profil *serial port* untuk bisa berkomunikasi.

e. *Object Exchange Protocol (OBEX)*, adalah lapisan protokol yang dipakai untuk saling tukar menukar atau transfer data antara perangkat *Bluetooth*.

Untuk lebih jelasnya mengenai *layer-layer* pada sistem *Bluetooth* bisa dilihat pada gambar 2.13 berikut :



Gambar 2.13 *Layer-layer* pada sistem *Bluetooth*.

Tiga buah lapisan fisik yang sangat penting dalam protokol arsitektur *Bluetooth* ini adalah :

1. *Bluetooth Radio,* adalah lapis terendah dari spesifikasi *Bluetooth*. Lapis ini mendefinisikan persyaratan yang harus dipenuhi oleh perangkat *tranceiver* yang beroperasi pada frekuensi 2,4 GHz ISM.

2. *Base Band,* lapisan yang memungkinkan terjadinya komuniksasi antara beberapa unit *Bluetooth*. Sistem Radio Frekuensi dari *Bluetooth* ini menggunakan FHSS (*Frequensy Hopping Spread Spectrum*) yang mengirimkan data dalam bentuk paket pada *time slot* dan frekuensi yang telah ditentukan, lapisan ini melakukan prosedur pemeriksaan dan *paging* untuk sinkronisasi transmisi frekuensi *hopping* dan *clock* dari perangkat *Bluetooth* yang berbeda.

3. *Link Manager Protocol* (LMP), bertanggung jawab terhadap *link set-up* antar perangkat *Bluetooth*. Hal ini termasuk aspek securiti seperti autentifikasi dan enkripsi dengan pembangkitan, penukaran dan pemeriksaan ukuran paket dari lapisan *base band*.

2.4.2 Profil – Profil *Bluetooth*

Profile – profil *Bluetooth* memungkinkan perangkat *Bluetooth* untuk bisa berkomunikasi dengan perangkat *Bluetooth* yang tidak sejenis seperti PDA *Bluetooth Enabled* dengan ponsel yang dilengkapi dengan *Bluetooth*, kedua perangkat ini mempunyai *Bluetooth stack*, dan profil *Bluetooth* memungkinkan

kedua perangkat ini untuk dapat saling berkomunikasi seperti *transfer* data diantara keduanya, dan menjelajah *internet* dengan memakai ponsel sebagai *wireless modem*-nya.

Sebuah *Bluetooth* profil adalah satu set fungsional yang dirancang untuk berangkat *Bluetooth*, untuk contoh tersebut diatas misalnya, ponsel dan PDA kedua-duanya harus didukung profil sinkronisasi agar bisa *transfer* data antara keduanya. Untuk mengirim data *object* seperti file a.vcf dari PDA ke ponsel, keduanya harus mempunyai profile *Object Push Profile* yang sudah diterapkan. Dan untuk *browsing* internet menggunakan PDA dengan ponsel sebagai *wireless modem*nya, keduanya harus didukung *Dial Up Networking Profiles*.

2.4.3 Konsep Aplikasi JAVA *Bluetooth*

Konsep dasar aplikasi *Bluetooth* (dengan Java atau yang lainnya) terdiri dari beberapa konsep, diantaranya :

a. Inisialisasi *Stack*

Sebelum kita melakukan berbagai operasi Java *Bluetooth* kita perlu menginisialisasikan *stack*, seperti telah tersebut diatas bahwa *stack* adalah perangkat lunak yang mengontrol perangkat *Bluetooth*. Inisialiasi *stack* terdiri dari sejumlah hal, akan tetapi tujuan utama dari inisialisasi *stack* adalah untuk menemukan perangkat *Bluetooth* yang siap berkomunikasi tanpa kabel. Setiap produsen perangkat *Bluetooth* menangani inisialisasi dengan cara yang berbeda-beda.

b. Manajemen Perangkat (*Device Management*)

LocalDevice dan *RemoteDevice* adalah dua kelas utama di dalam spesifikasi Java *Bluetooth* yang mengijinkan *user* untuk bisa melakukan pengaturan perangkat (*Device Management*). Kelas-kelas ini memberi user kemampuan untuk mengetahui informasi statistik sekitar perangkat *Bluetooth* (*LocalDevice*) dan informasi perangkat di sekitar area (*RemoteDevice*). Metode statis *LocalDevice.getLocalDevice* memberikan satu obyek *LocalDevice* yang siap untuk anda pergunakan. Untuk mendapatkan alamat unik dari perangkat *Bluetooth*, cukup panggil *getBluetoothAddress()* pada perangkat lokal *Bluetooth*. Setiap perangkat *Bluetooth* mempunyai satu alamat yang unik. Jika *user* ingin agar perangkat *Bluetooth* bisa ditemukan oleh perangkat *Bluetooth* lain di dalam area, cukup pakai metoda *setDiscoverable()* pada objek di dalam *LocalDevice.*

c. Pencarian Perangkat (*Device Discovery*)

Disekitar perangkat *Bluetooth* kita bisa terdapat perangkat *Bluetooth* yang lainnya seperti laptop, komputer *desktop*, *printer*, ponsel, atau mungkin juga PDA, untuk mengetahui dengan pasti tentang perangkat *Bluetooth* apa saja yang terdapat disekitar perangkat *Bluetooth* kita, perangkat *Bluetooth* akan menggunakan *class – class Device Discovery* yang telah disediakan untuk *API Java Bluetooth*.

d. Pencarian Layanan (*Service Discovery*)

Setelah menemukan perangkat *Bluetooth* diarea kita, maka selanjutnya perangkat *Bluetooth* akan mencari pelayanan apa saja yangditawarkan oleh perangkat *Bluetooth* yang telah kita temukan. Jika perangkat yang telah kita temukan (*remote device*) adalah sebuah *printer*, maka kita bisa mendapatkan layanan percetakan melalui *printer* tersebut. Tetapi jika *remote device* yang kita temukan adalah sebuah komputer, maka ada kemungkinan kita bisa melakukan berbagai operasi didalam komputer tersebut.

e. Pendaftaran Pelayanan (*Service Registration*)

Sebelum perangkat *Bluetooth* klien melakukan pencarian layanan pada sebuah perangkat *Bluetooth server*, perangkat *Bluetooth server* harus mendaftarkan pelayanan secara internal pada *database* pencarian layanan atau *Service Discovery Database* (SDDB). Proses ini disebut pendaftaran layanan (*Service Registration*). Pada aplikasi *peer-to-peer* seperti *transfer* data atau aplikasi *chat*, perlu di ingat bahwa setiap perangkat bisa berperan sebagai klien ataupun sebagai *server*, maka perlu dibedakan antara klien yang melakukan pencarian layanan dan *server* yang melakukan pendaftaran layanan.

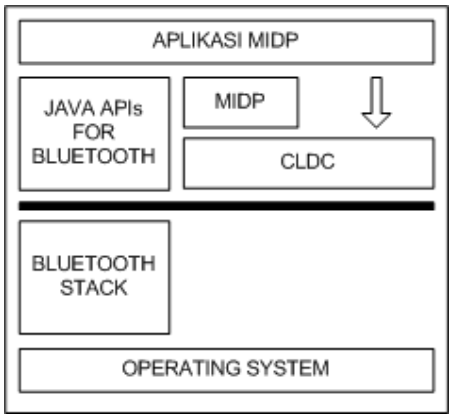
f. Komunikasi

Bluetooth adalah sebuah protokol komunikasi, *Java Bluetooth* API memberikan kita tiga cara untuk mengirim dan menerima data.

2.4.4 Pemrograman Java Untuk Teknologi *Bluetooth*

JSR-82 adalah spesifikasi standar pertama yang digunakan untuk membangun aplikasi *Bluetooth* dengan bahasa Java. *JSR-82* menyediakan kemampuan *Bluetooth* kedalam perangkat *virtual machine* J2ME. Sekumpulan Java API memisahkan kompleksitas dari *Bluetooth Protocol Stack* kedalam beberapa *Class, Interface,* dan beberapa Metode. API pada JSR-82 memungkinkan user untuk mencari dan mendaftarkan perangkat dan layanan, menetapkan, mengatur dan mengendalikan koneksi RFCOMM, L2CAP dan OBEX yang digunakan untuk komunikasi data. Selain itu JSR-82 juga menyediakan keamanan untuk semua aktivitas tersebut diatas.

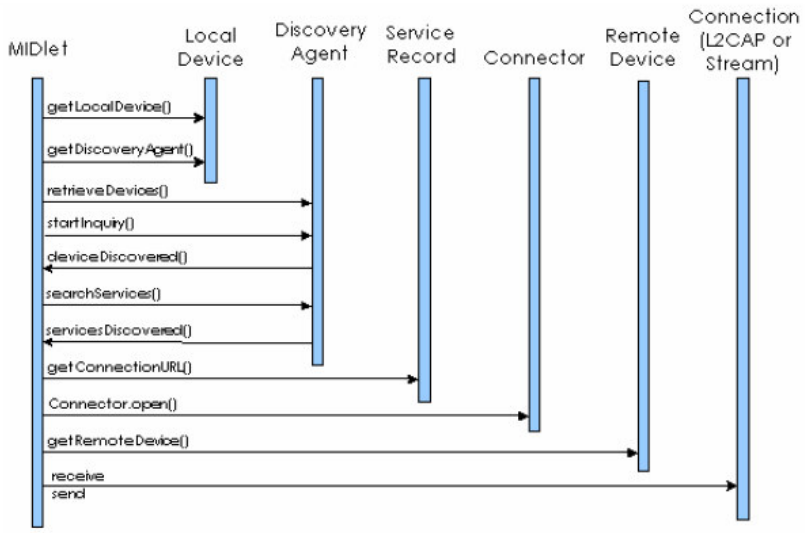
Gambar 2.14 berikut adalah gambaran dari penyusunan CLDC, MIDP, dan API dalam pengembangan aplikasi Java *Bluetooth*.



Gambar 2.14 penyusunan CLDC, MIDP , dan API

Aplikasi *Midlet* membutuhkan suatu mekanisme untuk menemukan perangkat dan pelayanan disekitar area. Pelayanan ini disediakan oleh *class – class discoveryAgent, discoveryListener, RemoteDevice* dan *ServiceRecord* yang

terdapat didalam inti dari *Bluetooth API*, yang memungkinkan *MIDlet* untuk melakukan pencarian perangkat dan pencarian layanan. Pertama *MIDlets* menggunakan metode *DiscoveryAgent.startInquiry* untuk menempatkan perangkat pada mode pencarian. Perangkat akan memanggil *Midlet* kembali ketika sebuah perangkat atau layanan ditemukan dengan menggunakan metode *deviceDiscovered()* atau *serviceDiscovered().* Metode *retrieveDevices()* dan *searchServices()* mencari layanan yang telah didapatkan sebelumnya pada *cache* lokal. Gambar 2.15 berikut adalah gambaran dari proses tersebut diatas :



Gambar 2.15 Proses *Running Midlets*

Dari beberapa penjelasan di atas, terlihat bahwa *Bluetooth* mampu menawarkan solusi yang cukup efektif dan efisien di dalam memberikan layanan kepada user untuk melakukan *transfer* data dengan kecepatan kurang dari 1 Mbit/s dan jangkauan yang relatif pendek. Teknologi *Bluetooth* masih memungkinkan untuk terus berkembang menuju kematangan baik dari sisi

standarisasi maupun aplikasi yang dapat diterapkan. Dengan pertimbangan bahwasannya *Bluetooth* mampu menyediakan berbagai macam aplikasi dan layanan dan dengan biaya yang relatif murah, mudah dalam pengoperasian, *interoperability* yang menjanjikan serta didukung oleh berbagai vendor besar di bidang telekomunikasi maupun komputer, dan lebih dari 1800 perusahaan telah bergabung sebagai *adopter* teknologi ini, maka tidak mustahil teknologi *Bluetooth* suatu saat akan menjadi salah satu primadona untuk digunakan baik untuk keperluan rumah tangga atau perkantoran/bisnis.

2.5 Teknologi JAVA.

Teknologi Java merupakan sebuah teknologi yang berkembang sangat pesat akhir-akhir ini. Bahkan belakangan ini dikabarkan berusaha mengalahkan *Microsoft* yang terkenal sebagai kampiun dari produsen *operating system* dimuka bumi ini. Teknologi Java yang pada awalnya dikenal untuk aplikasi pada *desktop* J2SE (*Java 2 Standard Edition*) ataupun aplikasi pada *server* J2EE (*Java 2 Enterprise Edition*), kini hadir dengan teknologi terbarunya, J2ME (*Java 2 Micro Edition*) *Platform*, untuk pembangunan aplikasi pada *mobile device* seperti *mobile phone* dan PDA. Selain J2ME™ Platform yang termasuk baru, terdapat pula sebuah *platform* teknologi Java yang termasuk baru pula yaitu *Java Card*. *Java Card* merupakan seperangkat perkakas (*tools*) untuk membangun aplikasi pada sebuah *card electronic* seperti SIM *Card* pada ponsel kita. SIM *Card* yang dipergunakan oleh *Mobile Banking* BCA (layanan operator selular Excelcomindo) menjadi salah satu contoh dari teknologi *Java Card*.

Java adalah teknologi berstandar terbuka yang memungkinkan dikembangkannya aplikasi terbuka di atasnya. Mekanisme ini membuat Java disebut sebagai teknologi *platform* terbuka. Pengembangan Java menggunakan bahasa Java sebagai basis pengembangannya yang tidak bergantung pada sistem operasi yang digunakan. Java sebagai bahasa pemrograman memungkinkan para programmer untuk :

1. Menulis sebuah aplikasi dan menjalankannya ditempat lain.
2. Membuat aplikasi yang berjalan di *browser*.
3. Mengembangkan aplikasi di sisi *server (server side)* seperti aplikasi perbankan, forum diskusi.
4. Membuat aplikasi untuk perangkat bergerak (*mobile*)seperti ponsel atau PDA ataupun perangkat lainnya.

Aplikasi Java yang dikembangkan didalam sebuah mekanisme yang melakukan interpretasi yang disebut *Java Virtual Machine* (JVM). Aplikasi Java tidak berinteraksi langsung dengan sistem operasi, tetapi berinteraksi dengan JVM, dimana JVM yang berinteraksi dengan sistem operasi.

2.5.1 *Java Virtual Machine* (JVM)

Java Virtual Machine adalah *software* yang berfungsi untuk menjalankan program Java supaya dapat dimengerti oleh komputer. Hal ini sangat penting karena Java *platform* adalah sebuah fondasi berbasis "hanya" perangkat lunak yang berjalan diatas sistem operasi, dimana sistem operasi sangat terikat dengan

perangkat keras. Alhasil, didapatkan bahwa setiap perangkat keras adalah bermacam-macam, *Java platform* diciptakan untuk mengatasi perbedaan ini.

Java sebagai teknologi *platform* telah berkembang menjadi berbagai macam solusi yaitu J2SE, J2EE, J2ME, dan JavaCard. Dimana semua solusi tersebut berjalan diatas JVM yang diciptakan berdasarkan lingkungan yang dijalankannya. Misalnya JVM Windows di Intel processor, JVM Linux di Itanium, atau JVM Sparc di Solaris. Bilamana dikembangkan sebuah aplikasi *desktop* menggunakan spesifikasi J2SE, maka pengembangan aplikasi tersebut dilakukan dengan bahasa Java dan menjalankannya memerlukan JVM.

Teknologi Java, XML, dan IP telah membuat Java ada dimana-mana, *multi device computing*. Bahasa Java telah memberikan cara yang konsisten bagi *programmer* agar dapat menulis kode yang dapat dieksekusi pada berbagai *device*. Untuk *client-side programming, Visual Basic* masih mendominasi pasar dunia, sedangkan untuk pasar *application server* hampir seluruh pasar telah diambil oleh Java, dan kinipun teknologi Java telah diadopsi oleh berbagai *vendor mobile devices* seperti ponsel maupun PDA.

2.5.2 Teknologi J2ME (*Java 2 Micro Edition*)

Java ME *platform* adalah sebuah kumpulan dari teknologi dan spesifikasi yang bisa digabungkan untuk membangun suatu *Java Runtime Environment* yang lengkap terutama untuk memenuhi kebutuhan dari berbagai perangkat elektronik yang telah banyak beredar. Teknologi Java *ME* terdiri berdasarkan tiga elemen utama, yaitu :

1. *Configuration*, yang menyediakaan *set library* yang paling mendasar dan kemampuan *virtual machine* untuk jangkauan yang luas dari perangkat.
2. *Profile*, adalah satu set dari *Aplication Programming Interface* (API) yang mendukung cakupan yang lebih dangkal dari perangkat, dan
3. Paket opsional (*optional package*), adalah satu set API dari teknologi yang lebih spesifik.

2.5.3 Konfigurasi (*Configuration*)

Konfigurasi adalah sekumpulan API level rendah dan mesin maya yang dioptimasi untuk sekumpulan perangkat. Saat ini, *platform Java ME* terbagi menjadi dua konfigurasi dasar, salah satu yang dibuat untuk memenuhi kebutuhan perangkat *mobile* kecil seperti ponsel disebut *Connected Limited Device Configuration* (CLDC), sedangkan yang lainnya dibuat untuk perangkat *mobile* yang mempunyai kemampuan lebih besar seperti *smartphone* disebut *Connected Device Configuration* (CDC).

Connected Limeted Device Configuration (CLDC) adalah suatu konfigurasi yang diarahkan untuk perangkat dengan sumber daya terbatas seperti ponsel, konfigurasi CLDC didesain khusus untuk memenuhi kebutuhan perangkat untuk menjalankan teknologi Java pada perangkat yang memiliki keterbatasan *memory*, kecepatan processor, dan keterbatasan dalam kemampuan grafik. Sedangkan *Connected Device Configuration* (CDC) adalah suatu konfigurasi yang

diarahkan untuk perangkat *mobile* yang mempunyai kemampuan yang lebih besar dengan koneksi jaringan seperti *Personal Digital Assistant* (PDA) dan *set-top boxes.*

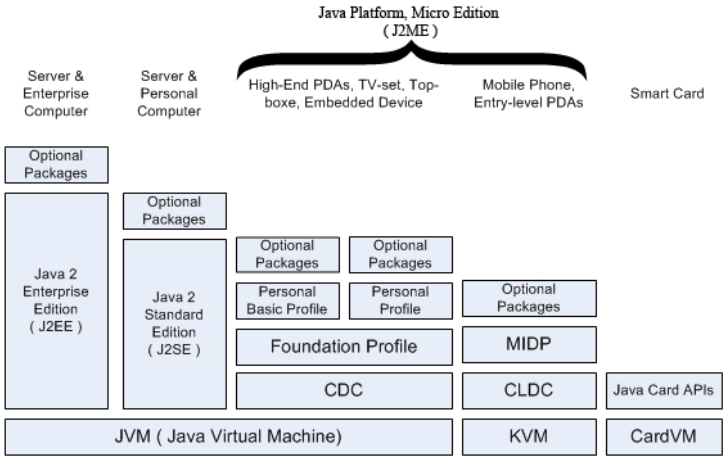
Tabel 2.1 Perbandingan CLDC dan CDC

CLDC (Connected Limited Device Configuration)	CDC (Connected Device Configuration)
Mengimplementasikan subset dari J2SE.	Mengimplementasikan seluruh fitur dari J2SE.
JVM yang digunakan adalah KVM.	JVM yang digunakan adalah CVM.
Digunakan pada perangkat handheld (handphone, PDA, twoway pager) dengan *memory* terbatas(160-512 kb).	Digunakan pada perangkat handheld (internet TV, Nokia Communicator, car TV) dengan *memory* minimal 2 Mb.
Prosesor : 16/ 32 bit.	Prosesor : 32 bit

2.5.4 *Profile*

Profile merupakan kebalikan dari *configuration,* yaitu mengatur hal-hal yang spesifik untuk sebuah *device* atau tipe market. Misalkan pada *profile* ini diatur tentang *persistent storage* dan *user interface* (UI). Saat ini *Java Community Process (*JCP*)* telah mendefinisikan lima buah profile, salah satunya yaitu MIDP (*Mobile Information device Profile*), MIDP adalah profile yang digunakan pada banyak *mobile devices.*

Gambar 2.16 berikut ini memperlihatkan lebih jelas bagaimana komponen dari Java ME berhubungan dengan teknologi Java lainnya.



Gambar 2.16 Berbagai platform Java dengan J2ME didalamnya (CDC dan CLDC).

Aplikasi yang diprogram dengan menggunakan *platform* J2SE tidak dapat dijalankan pada *mobile devices,* karena ada perbedaan terutama pada hal *prosesor computing* dan *memory*-nya. Pada J2SE menggunakan JVM sebagai *interpreter*-nya, sedangkan pada *platform* J2ME menggunakan *Kilo Virtual Machine* (KVM) yang dipasang didalam *handheld devices*. J2ME merupakan *superset* dari J2SE, sebaliknya J2SE merupakan *superset* bagi J2ME. Artinya API yang ada pada J2ME sebagian mengadopsi yang ada pada API J2SE, selain juga meng-implementasikan API spesifik untuk teknologi J2ME yaitu *javax.microedition.*

Keuntungan yang paling menonjol dari J2ME dibandingkan dengan teknologi *wireless* sebelumnya adalah *security* dan *disconnected access* and *synchronization.*

Security menjadi sebuah isu yang sangat penting dewasa ini. Dalam teknologi WAP, *device* me-*retrieve* dan menampilkan *via microbrowser* dan dibutuhkan suatu *gateway* untuk jembatan antara jaringan *internet* dan jaringan *wireless* itu sendiri. Maka wilayah jaringan *internet* menjadi salah satu hal potensial terhadap gangguan keamanan data.

Disconnected access and synchronization maksudnya adalah *wireless device* tetap dapat menjalankan aplikasinya secara *stand alone* meskipun telah putus dari jaringan *wireless*-nya atau berada diluar jangkauan jaringan. Pada aplikasi WAP, *device* harus selalu terkoneksi pada jaringan *wireless*-nya untuk mengakses aplikasi berbasis WML tersebut.

Selain dari kedua keuntungan diatas juga terdapat keuntungan yang secara umum dimiliki oleh aplikasi berbasis Java lainnya yaitu:

1. *Dynamic delivery of application and services*. Aplikasi berbasis Java dapat di-*download* secara *real time over the air* (OTA) dan me-*running*-nya dalam berbagai *device.*
2. *Cross platform compatibility.* Aplikasi berbasis Java mampu berjalan pada berbagai *platform* yang berbeda.
3. *Enhances user experience*. Teknologi Java kaya akan grafik dengan interaksi yang cepat pula serta grafik di-*generate* secara lokal sehingga kebutuhan terhadap *network bandwidth* menjadi tereduksi.
4. *Scalability and performance*. Skalabilitas dan *performance* dapat ditingkatkan karena aplikasi Java dapat berjalan secara *stand alone mode*.

Saat ini seluruh produsen *mobile handset* besar didunia telah memproduksi Java *Technology-enabled mobile phones* atau ponsel yang telah dilengkapi dengan teknologi Java. Pada saat ini telah diproduksi lebih dari 94 juta *mobile handset Java enabled* di dunia ini yang diproduksi oleh kurang lebih 20 produsen atau manufaktur (dalam 150 model) seperti Nokia, Siemens, Sony Ericsson, Samsung, Motorola, dan lain-lain. Sehingga dapat diambil kesimpulan bahwa perangkat elektronik khususnya yang bersifat *mobile* yang telah dilengkapi dengan teknologi Java dan *Bluetooth* akan terus berkembang dan menjadi salah satu primadona untuk digunakan baik untuk keperluan rumah tangga atau perkantoran/bisnis.

2.6 Pemrograman Berorientasi Objek (*Object Oriented Progeramming*)

Pemrograman berorientasi objek adalah suatu metode pemrograman dimana program diorganisasikan sebagai suatu kumpulan objek yang bekerja sama, masing masing objek merepresentasikan *instant* dari kelas, dan kelas itu anggota dari hirarki kelas-kelas yang disatukan lewat keterhubungan pewarisan.

Model data berorientasi objek dikatakan dapat member fleksibilitas yang tinggi, kemudian mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung *Object Oriented Programming* (OOP) mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat. Pemrograman berorientasi objek ini terdiri dari beberapa konsep sebagai berikut :

1. Kelas (*Class*)

   Kumpulan definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebuah *class* secara umum sebaiknya dapat dikenali oleh seorang *non-programmer* sekalipun terkait dengan *domain* permasalahan yang ada, dan kode yang terdapat dalam suatu *class* sebaiknya (*relative*) bersifat mandiri dan independen.

2. *Object (*objek*)*

   Membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program computer berorientasi objek.

3. Abstraksi (*Abstraction*)

   Abstaksi adalah kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari “pelaku” abstrak yang dapat melakukan kerja, laporan dan perubahan keadaan, dan berkomunikasi dengan objek lainnya dalam sistem tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

4. Enkapsulasi (*Encapsulation*)

   Memastikan pengguna tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak, hanya metode dalam

objek tersebut yang diberi ijin untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

5. Pewarisan (*Inheritance*)

Suatu kelas dapat diciptakan berdasarkan kelas lain. Kelas baru ini mempunyai sifat-sifat yang sama dengan kelas pembentuknya, ditambah sifat-sifat khusus lainnya. Dengan kata lain kelas baru ini mewarisi sifat-sifat kelas induknya. Kelas baru ini disebut kelas turunan, sedangkan kelas yang menurunkan kelas turunan disebut kelas induk atau kelas dasar. Dengan konsep pewarisan seperti ini kita dapat menciptakan kelas baru yang mempunyai sifat sama dengan kelas lain tanpa harus menulis ulang bagian-bagian yang sama. Disamping *level* kenampakan anggota kelas *public* dan *private*, dalam kelas dasar seringkali terdapat *level* kenampakan anggota kelas *protected*. Anggota kelas yang mempunyai kenampakan *protected* artinya bahwa anggota kelas ini selain dapat diakses di dalam kelas itu sendiri (seperti *private*), tetapi juga dapat diakses di dalam kelas turunannya.

6. Polimorfisme

Polimorfisme merupakan fitur pemrograman berorientasi objek yang penting setelah pengkapsulan dan pewarisan. Polimorfisme

berasal dari bahasa Yunani, ***poly*** (banyak) dan ***morphos*** (bentuk). Polimorfisme menggambarkan kemampuan kode C++ berperilaku berbeda tergantung situasi pada waktu *run* (program berjalan). Konstruksi ini memungkinkan untuk mengadakan ikatan dinamis (juga disebut ikatan tunda, atau ikatan akhir). Kalau fungsi-fungsi dari suatu kelas dasar didefinisikan ulang atau ditindih pada kelas turunan, maka objek-objek yang dihasilkan hirarki kelas berupa objek polimorfik. Polimorfik artinya mempunyai banyak bentuk atau punya kemampuan untuk mendefinisi banyak bentuk.

Dengan menggunakan OOP maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesasaikan masalah tersebut (terstruktur), tetapi objek-objek apa saja yang dapat melakukan pemecahan masalah tersebut.

2.7 *Unified Modeling language* (UML)

UML adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, memvisualisasikan, dan membangun perangkat lunak. UML tidak berdasarkan pada bahasa pemrograman tertentu. UML dapat digunakan untuk membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Karena UML menggunakan *class* dan *operation* pada dasarnya, maka UML lebih cocok untuk

penulisan perangkat lunak pada bahasa pemrograman berorientasi objek seperti Java, C++ dan bahasa berorientasi objek lainnya.

2.7.1 Tujuan Perancangan UML

Perancangan UML dalam pengembangan sebuah aplikasi mempunyai beberapa tujuan utama sebagai berikut :

1. Menyediakan bahasa pemodelan *visual* yang ekspresif dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti.
2. Menyediakan mekanisme perluasan dan spesifikasi untuk memperluas konsep-konssep inti.
3. Mendukung spesifikasio independen bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk bahasa pemodelan.
5. Mendukung konsep-konsep pemgembangan level lebih tinggi seperti komponen, kolaborasi, *framework*, dan *pattern*.

2.7.2 Diagram dan Teknik Pemodelan UML

Diagram menggambarkan banyak hal, penggunaan notasi yang terdefinisi baik dan ekspresif adalah penting pada proses pemgembangan perangkat lunak, yaitu:

1. Notasi standar memungkinkan pengembang mendeskripsi- kan rumusan arsitektur dan kemudian membuat komunikasi secara tidak ambigu.

2. Notasi yang bagus membebaskan otak untuk berkonsentrasi pada masalah masalah yang lebih lanjut.
3. Ntasi yang baik memungkinkan melakukan eliminasi keperluan pemeriksaan konsistensi dan kebenaran keputusan menggunakan alat yang otomatis.

UML menyediakan beberapa diagram visual yang menunjukna berbagai aspek dalam sistem. Diagram-diagram yang disediakan dalam UML adalah sebagai berikut :

a. Diagram Kelas (*Class Diagram*)

Diagram ini menunjukan sekumpulan kelas, *interface*, dan kolaborasi dan keterhubungannya. Diagram kelas ditujukan untuk pandangan *static* terhadap sistem.

b. Diagram Objek (*Object Diagram*)

Diagram ini menunjukkan sekumpulan objek dan keterhubungannya. Diagram ini menunjukkan potongan statik dari *instant-instant* yang ada di diagram kelas. Diagram ini memperlihatkan satu *prototype* atau kasus tertentu yang mungkin terjadi. Diagram objek menyediakan notasi grafis formal guna memodelkan objek, kelas, dan saling keterhubungannya. Diagram objek berguna untuk *abstract modeling* dan perancangan program-program sesungguhnya. Pada pendekatan ini, bentukan dasar dari sistem perangkat lunak adalah objek atau kelas.

c. Diagram *Use-Case* (*Use-Case Diagram*)

Diagram ini menunjukan sekumpulan kasus fungsional dan aktor (jenis kelas khusus) dan keterhubungannya.

d. Diagram Sekuen (*Sequence Diagram*)

Diagram ini menunjukkan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem. Diagram ini menekankan pada basis urutan waktu dari pesan-pesan yang terjadi.

e. Diagram Kolaborasi (*Collaboration Diagram*)

Diagram ini juga merupakan diagram interaksi. Diagram ini menekankan pada organisasi struktur dari objek-objek yang mengirim dan menerima pesan.

f. Diagram *Statechart* (*Statechart Diagram*)

Diagram ini berisi status, ransisi, kejadian, dan aktifitas. Statechart merupakan pandangan dari sistem. Diagram ini penting dalam memodelkan perilaku antarmuka, kelas, kolaborasi, dan menekankan ada urutan kejadian.

g. Diagram Aktifitas (*Activity Diagram*)

Diagram ini untuk menunjukan aliran aktifitas pada sistem. Diagram ini adalah pandangan dinamis terhadap sistem. Diagram ini penting untuk memodelkan fungsi sistem dan menekankan pada aliran kendali di antara objek-objek.

h. Diagram Komponen (*Component Diagram*)

Diagram ini menunjukkan organisasi dan hubungan ketergantungan di antara sekumpulan komponen. Diagram ini merupakan pandangan static terhadap implementasi sistem.

i. Diagram Pengembangan (*Deployment Diagram*)

Menunjukkan konfigurasi pemrosesan saat jalan dn komponen-komponen yang terdapat di dalamnya. Pilihan model dan diagram yang digunakan dipengaruhi oleh bagaimana persoalan ditangani dan bagaimana solusi dibentuk. Setiap sistem kompleks perlu didekati melalui sekumpulan pendangan model yang hampir *independent*.