BAB II

LANDASAN TEORI

Penyusunan tugas akhir ini berisi tentang pembuatan aplikasi yang dapat membantu masyarakat untuk mencapai ke tempat tujuan dengan menggunakan alat transportasi umum yang berjalan pada aplikasi *smartphone* berbasis android dan memanfaatkan GPS dari google maps (*Api's Key*). Sehingga pembahasan pembahasan teori yang mendukung isi dari tugas akhir ini mengenai teori, teknologi, android beserta *tools* yang mendukung dalam mengerjakan aplikasi ini.

2.1. Aplikasi

Perangkat lunak aplikasi yaitu perangkat lunak yang digunakan untuk membantu pemakai komputer untuk melaksanakan pekerjaannya. Jika ingin mengembangkan program aplikasi sendiri, maka untuk menulis program aplikasi tersebut, dibutuhkan suatu bahasa pemrograman, yaitu *language software*, yang dapat berbentuk *assembler*, *compiler* ataupun *interpreter*. Jadi *language software* merupakan bahasanya dan program yang ditulis merupakan program aplikasinya. *Language software* berfungsi agar dapat menulis program dengan bahasa yang lebih mudah, dan akan menterjemahkannya ke dalam bahasa mesin supaya bisa dimengerti oleh komputer. Bila hendak mengembangkan suatu program aplikasi untuk memecahkan permasalahan yang besar dan rumit, maka supaya program aplikasi

tersebut dapat berhasil dengan baik, maka dibutuhkan prosedur dan perencanaan yang baik dalam mengembangkannya.

Sekarang, banyak sekali program-program aplikasi yang tersedia dalam bentuk paket-paket program. Ini adalah program-program aplikasi yang sudah ditulis oleh orang lain atau perusahaan-perusahaan perangkat lunak. Beberapa perusahaan perangkat lunak telah memproduksi paket-paket perangkat lunak yang mempunyai reputasi internasional. Program-program paket tersebut dapat diandalkan, dapat memenhi kebutuhan pemakai, dirancang dengan baik, relatip bebas dari kesalahan-kesalahan, *user friendly* (mudah digunakan), mempunyai dokumentasi manual yang memadai, mampu dikembangkan untuk kebutuhan mendatang, dan didukung perkembangannya. Akan tetapi, bila permasalahannya bersifat khusus dan unik, sehingga tidak ada paket-paket program yang sesuai untuk digunakan, maka dengan terpaksa harus mengembangkan program aplikasi itu sendiri (Jogiyanto Hartono, 2004, pp397-398).

2.2. Transportasi

Transportasi adalah pemindahan manusia atau barang dari satu tempat ke tempat lainnya dengan menggunakan sebuah kendaraan yang digerakkan oleh manusia atau mesin. Transportasi digunakan untuk memudahkan manusia dalam melakukan aktivitas sehari-hari. Di negara maju, mereka biasanya menggunakan kereta bawah tanah (*subway*) dan taksi. Penduduk disana jarang yang mempunyai kendaraan pribadi karena mereka sebagian besar menggunakan angkutan umum

sebagai transportasi mereka. Transportasi sendiri dibagi 3 yaitu, transportasi darat, laut, dan udara. Transportasi udara merupakan transportasi yang membutuhkan banyak uang untuk memakainya. Selain karena memiliki teknologi yang lebih canggih, transportasi udara merupakan alat transportasi tercepat dibandingkan dengan alat transportasi lainnya, dan Transportasi darat adalah segala bentuk transportasi menggunakan jalan untuk mengangkut penumpang atau barang. Bentuk awal dari transportasi darat adalah menggunakan kuda, keledai atau bahkan manusia untuk membawa barang melewati jalan setapak. Seiring dengan berkembangkan perdagangan, jalan diratakan atau dilebarkan untuk mengakomodir aktivitas. Roda kemudian ditemukan.

2.3. Basis Data

Basis data terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, symbol, teks, gambar, bunyi, atau kombinasinya.

Basis data sendiri dapat didefinisiikan dalam sejumlah sudut pandang seperti :

a. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
b. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
c. Kumpulan file/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimjpanan elektronis.

Basis data dan lemari arsip sesungguhnya memiliki prinsip kerja dan tujuan yang sama. Prinsip utamanya adalah pengaturan data/arsip. Dan tujuan utamanya adalah kemudahan dan kecepatan dalam pengambilan kembali data/arsip. Perbedaannya hanya terletak pada media penyimpanan yang digunakan. Jika lemari arsip menggunakan lemari dari besi atau kayu sebagai media penyimpanan, maka basis data menggunakan media penyimpanan elektronis seperti disk (disket atau hard disk). Hal ini merupakan konsekuensi yang logis, karena lemari arsip langsung dikelola/ditangani oleh manusia, sementara basis data dikelola/ditangani melalui perantara alat/mesin pintar elektronis (yang dikenal sebagai komputer). Perbedaan media ini yang selanjutnya melahirkan perbedaan media ini yang selanjutnya melahirkan perbedaan-perbedaan lain yang menyangkut jumlah dan jenis metoda/cara yang dapat digunakan dalam upaya penyimpanan.

2.3.1. Operasi Dasar Basis Data

Didalam sebuah disk, basis data dapat diciptakan dan dapat pula ditiadakan. Didalam sebuah disk, kita dapat pula menempatkan beberapa (lebih dari satu) basis data. Sementara dalam sebuah basis data, kita dapat menempatkan satu atau lebih file/tabel. Pada file/tabel inilah sesungguhnya data disimpan/ditempatkan. Setiap basis data umumnya dibuat untuk mewakili sebuah semesta data yang spesifik. Misalnya, ada basis data kepegawaian, basis data akademik, basis data inventori (Pergudangan), dan sebagainya. Sementara dalam basis data akademik, misalnya, kita dapat menempatkan file mahasiswa, file mata_kuliah, file dosen, file jadwal, file kehadiran, file nilai, dan seterusnya.

Karena itu, operasi-operasi dasar yang dapat kita lakukan berkenaan dengan basis data dapat meliputi :

a. Pembuatan basis data baru (*create database*), yang identik dengan pembuatan lemari arsip yang baru
b. Penghapusan basis data (*drop database*), yang identik dengan perusakan lemari arsip (sekaligus beserta isinya, jika ada).
c. Pembuatan file/tabel dari suatu basis data (*create table*), yang identik dengan penambahan map arsip baru ke sebuah lemari sarsip yang telah ada.
d. Penghapusan file/tabel dari suatu basis data (*drop table*), yang identik dengan perusakan map arsip lama yang ada di sebuah lemari arsip.

e. Penambahan/pengisian data baru ke sebuah file/tabel disebuah basis data (*insert*), yang identik dengna penambahan ke lemari arsip ke sebuah map arsip

f. Pengambailan data dari sebuah file/tabel (*retrieve/search*) yang identik dengan pencarian lembaran arsip dari sebuah map arsip

g. Pengubahan data dari sebuah file/tabel (*update*), yang identik dengan perbaikan isi lembaran arsip yang ada di sebuah map arsip.

h. Penghapusan data dari sebuah file/tabel (*delete*), yang identik dengan penghapusan sebuah lembaran arsip yang ada di sebuah map arsip

Operasi yang berkenaan dengan pembuatan objek (basis data dan tabel) merupakan operasi awal yang hanya dilakukan sekali dan berlaku seterusnya. Sedang operasi-operasi yang berkaitan dengan isi tabel (data) merupakan operasi rutin yang akan berlangsung berulang-ulang dank arena itu operasi-operasi inilah yang lebih tepat mewakili aktivitas pengelolaan (*management*) dan pengolahan (*processing*) data dalam basis data.

2.3.2 Objektif Basis Data

Telah disebutkan di awal bahwa tujuan awal dan utama dalam pengelolaan data dalam sebuah basis data adalah agar kita dapat memperoleh menemukan kembali data (yang dicari) dengan mudah dan cepat. Disamping itu, pemanfaatan basis data untuk pengelolaan data, juga memiliki tujuan-tujuan lain.

Secara lebih lengkap, pemanfaatan basis data dilakukan untuk memenuhi sejumlah tujuan (objektif) seperti berikut ini :

a. Kecepatan dan kemudahan

Pemanfaatan basis data memungkinkan untuk dapat menyimpan data atau melakukan perubahan/manipulasi terhadap data atau menampilkan kembali data tersebut dengan lebih cepat dan mudah, daripada jika menyimpan data secara manual (non-elektronis) atau secara elektronis (tetapi tidak dalam bentuk penerapan basis data, misalnya dalam bentuk *spread sheet* atau dokumen teks biasa).

b. Efisiensi ruang penyimpanan (*Space*)

Karena keterkaitan yang erat antara kelompok data dalam sebuah basis data, maka redudansi (pengulangan) data pasti akan selalu ada. Banyaknya redudansi ini tentu akan memperbesar ruangan penyimpanan (baik di memori utama maupun memori sekunder) yang harus disediakan. Dengan basis data, efisiensi/optimalisasi penggunaan ruang penyimpanan dapat dilakukan, karena kita dapat melakukan penekanan jumlah redudansi data, baik dengan menerapkan sejumlah pengkodean atau dengan membuat relasi-relasi (dalam bentuk file) antara kelompok data yang saling berhubungan.

c. Keakuratan (*Accurancy*)

Pemanfaatan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan/batasan (*constrain*) tipe data, domain data, keunikan data, dan sebagainya, yang secara ketat diterapkan dalam sebuah basis data, sangat berguna untuk menekan ketidakakuratan pemasukan/penyimpanan data.

d. Ketersediaan (*Availability*)

Pertumbuhan data (baik dari sisi jumlah maupun jenisnya) sejalan dengan waktu akan semakin membutuhkan ruang penyimpanan yang besar. Padahal tidak semua data itu selalu kita gunakan/butuhkan. Karena itu kita dapat memilah adanya data utama/*master*/refereensi, data transaksi, data histori hingga data kadaluarsa. Data yang sudah jarang atau bahkan tidak pernah lagi kita gunakan, dapat kita atur untuk dilepaskan dari sistem basis data yang sedang aktif (menjadi *off-line*) baik dengan cara penghapusan atau dengan memindahkannya ke media penyimpanan *off-line* (seperti *removable disk*, atau *tape*). Di sisi lain, karena kepentingan pemakaian data, sebuah geografis. Data nasabah sebuah bank, misalnya, dipisah-pisahkan dan disimpan di lokasi yang sesuai dengna keberadaan nasabah. Dengan pemanfaatan teknologi jaringan komputer, data yang berada di suatu lokasi/cabang, dapat juga di akses (menjadi tersedia/*available*) bagi lokasi/cabang lain.

e. Kelengkapan (*Completeness*)

Lengkap/tidaknya data yang kita kelola dalam sebuah basis data bersifat relative (baik terhadap kebutuhan pemakai maupun terhadap waktu). Bila seorang pemakai sudah menganggap bahwa data yang dipelihara sudah lengkap, maka pemakai yang lain belum tentu berpendapat sama. Atau, yang sekarang diangap sudah lengkap belum tentu di masa yang akan data juga demikian. Dalam sebuah basis data, disamping data kita juga harus menyimpan struktur (baik yang mendefinisikan objek-objek dalam basis data maupun definisi detail dari tiap objek seperti struktur file/tabel atau indeks). Untuk mengakomodasi kebutuhan kelengkapan data yang semakin berkembang, maka kita tidak hanya dapat menambahkan *record-record* data, tetapi juga dapat melakukan perubahan struktur dalam basis data baik dalam bentuk penambahan objek baru (tabel) atau dengan penambahan field-field baru pada suatu tabel.

f. Keamanan (*Security*)

Memang ada sejumlah sistem (aplikasi) pengelola basis data yang tidak menerapkan aspek keamanan dalam penggunaan basis data. Tetapi untuk sistem yang besar dan serius, aspek keamanan juga dapat diterapkan dengan ketat. Dengan begitu kita dapat menentukan siapa-siapa (pemakai) yang boleh menggunakan basis data beserta objek-objek di dalamnya dan menentukan jenis-jenis operasi apa saja yang boleh dilakukannya.

g. Kebersamaan Pemakai (*Sharebility*)

Pemakai basis data seringkali tidak terbatas pada satu pemakai saja, atau di satu lokasi saja atau boleh satu sistem/aplikasi saja, data pegawai dalam basis data kepegawaian, misalnya, dapat digunakan oleh banyak pemakai, dari sejumlah departemen dalam perusahaan atau oleh banyak ssitem (sistem penggajian, sistem akuntansi, sistem inventori, dan sebagainya). Basis data yang dikelola oleh sistem (aplikasi) yang mendukung lingkungan *multiuser*, akan dapat memenuhi kebutuhan ini, tetapi tetap dengan menjaga/menghindari terhadap munculnya persoalan baru seperti inkonsistensi data (karena data yang sama diubah oleh banyak pemakai pada saaat yan gbersamaan) atau kondisi *deadlock* (karena banyak pemakai yang saling menunggu untuk menggunakan data).

2.4. Perkembangan Teknologi *Mobile*

2.4.1. Generasi I (1G)

Komunikasi *mobile* yang ada saat ini berawal pada akhir tahun 1970-an, dengan diimplementasikannya sistem percobaan di Chicago pada tahun 1978. Sistem tersebut menggunakan teknologi yang disebut *Advanced Mobile Phone Service* (AMPS), yang beroperasi pada frekuensi 800 Mhz. Di Jepang, sistem AMPS komersial diluncurkan tahun 1979 sedangkan di Amerika, sistem komersialnya diluncurkan di Chicago pada tahun 1983.

Setelah itu, beberapa sistem *wireless* 1G lain dikembangkan, namun AMPS tetap merupakan sistem yang paling populer. Sistem *wireless* 1G

merupakan sistem analog walaupun memanfaatkan sinyal digital dibeberapa aspek jaringannya. Sistem *wireless* 1G menggunakan skema *Frequency Division Multiple Access* (FDMA).

Layanan utama sistem *wireless* 1G adalah komunikasi suara. Walaupun sistem ini sangat sukses, sistem ini memiliki beberapa kekurangan. Kekurangan yang utama adalah bahwa sistem 1G memiliki kapasitas sistem yang terbatas selain kurangnya mekanisme keamanan serta masalah yang dikenal dengan "*lost mobile*", yaitu jaringan yang salah menentukan posisi pelanggan. Beberapa kekurangan ini mendorong dikembangkannya sistem *wireless* 2G

2.4.2. Generasi II (2G)

Tidak seperti sistem 1G yang masih analog, sistem 2G menggunakan teknologi digital. Penggunaan teknologi digital mempunyai beberapa keuntungan, diantaranya memperbesar kapasitas dan mekanisme keamanan yang lebih baik. Seperti sistem 1G, beberapa tipe teknologi 2G juga dikembangkan. Tiga teknologi yang populer adalah *Time Division Multiple Access* (TDMA), *Global System for Mobile Communication* (GSM), dan *Code Division Multiple Access* (CDMA).

TDMA atau *Interim Standard 136* (IS-136) muncul dari dua tahap evolusi AMPS analog. Tahap pertama adalah pengenalan *channel* suara digital. Tahap ini meliputi aplikasi *Time Division Multiplexing* (TDM) pada *channel* suara

dimana setiap *channel* suara dibagi kedalam beberapa *time slot* yang memungkinkan tiga percakapan secara serentak pada *channel* RF yang sama. Tahap ini dikenal dengan *Digital* AMPS, dipublikasikan tahun 1990 dan sistem ini memberi tambahan kapasitas yang cukup signifikan dibandingkan dengan AMPS. *Digital* AMPS hanya menggunakan teknologi digital pada *channel* suara, sedangkan *channel* kendali (*control channel*) masih menggunakan teknologi analog. Ini menyebabkan keterbatasan jumlah layanan yang dapat disediakan. Untuk mengatasi hal ini, pada tahun 1994 dikembangkanlah TDMA dengan *channel* suara dan *channel* kendali yang sudah digital. Baik *Digital* AMPS maupun TDMA hingga saat ini masih digunakan. Digital AMPS beroperasi pada frekuensi 800 Mhz, sedangkan TDMA pada frekuensi 800 Mhz dan 1900 Mhz.

GSM muncul dari keinginan orang Eropa untuk memenuhi kebutuhannya akan sistem digital pan-Eropa. Pada tahun 1982, *Conference on European Posts and Telecommunication* (CEPT) mulai mengembangkan sistem ini. Organisasi ini mendirikan perkumpulan yang diberi nama *Group Speciale Mobile* (GSM) yang menentukan teknis standar sistem digital yang dikembangkannya. Pada tahun 1989, usaha ini diambil alih oleh *European Telecommunications Standards Institute* (ETSI) dan ETSI merampungkan spesifikasi teknis I sistem ini yang diberi nama seperti perintisnya, *Group Speciale Mobile* (GSM). GSM diluncurkan tahun 1991 dan tidak hanya populer di Eropa, tetapi mulai mendunia. Oleh karena itu GSM berubah menjadi *Global System for Mobile*

Communication. GSM awalnya beroperasi pada frekuensi 900 Mhz, setelah itu dapat juga beroperasi pada frekuensi 1800 Mhz dan 1900 Mhz. GSM juga menggunakan TDMA yang berarti *channel* radio tersendiri dibagi kedalam *timeslot* yang memungkinkan beberapa *user* membagi satu *channel* RF berbasis *time sharing*. Teknik ini dapat menyediakan penambahan kapasitas dibandingkan dengan sistem analog dimana tiap *channel* radio diperuntukkan untuk satu percakapan.

CDMA atau *Interim Standard 95* (IS-95) merupakan sistem 2G lain yang memungkinkan beberapa *user* untuk berbagi frekuensi radio, semua *user* berbagi frekuensi pada waktu yang sama. Berbagi frekuensi secara bersama menyebabkan mereka dapat saling menggangu. Tekniknya adalah memilih sinyal seorang *user* yang dapat dilakukan jika sinyal dari tiap *user* dimodulasi dengan urutan kode yang unik, dimana bit rate kode jauh lebih tinggi dari bit rate informasi yang dikirimkan. Terakhir, pengetahuan akan urutan kode yang digunakan untuk sinyal yang diberikan akan memungkinkan sinyal untuk diekstrak.

Dibandingkan dengan sistem 1G, sistem 2G lebih aman, menyediakan kapasitas yang lebih besar, dan lebih banyak fitur-fitur baru. Namun sistem ini masih terus dioptimalkan dan belum memadai untuk komunikasi data.

2.4.3. Generasi Transisi (2,5G)

Sistem 2G dikembangkan karena kekurangan yang fundamental dari sistem 1G yaitu terbatasnya kapasitas dan kurangnya mekanisme keamanan. Dalam

dunia internet, *e-commerce*, dan komunikasi multimedia seperti saat ini, dukungan yang terbatas untuk melakukan komunikasi data adalah sebuah kekurangan yang serius. Untuk itulah muncul wacana pengembangan sistem yang lebih sesuai untuk komunikasi data, 3G.

Untuk menjembatani sistem 2G dan 3G, dikembangkanlah teknologi transisi yaitu sistem 2.5G, diantaranya *General Packet radio Service*

(GPRS), CDMA2000 phase 1 dan *Enhanced Data Rates for Glogal Evolution* (EDGE). Konsep penting dalam sistem 2.5G adalah pengenalan layanan paket data.

GSM menyediakan layanan suara dan data yang berbasis *circuit switched*. Untuk layanan layanan data, jaringan GSM mengemulasi modem antara perangkat pada sisi *user* dengan jaringan data tujuan secara efektif. Mekanisme ini ternyata tidak efisien untuk mendukung lalu-lintas data. Terlebih lagi GSM hanya mendukung kecepatan transfer data ± 9.6 – 14.4 Kbps yang tergolong sangat lambat.

GPRS didesain untuk menyediakan layanan paket data dengan kecepatan transfer yang lebih tinggi dari GSM. GPRS berbasis teknologi *packet switched* yang berarti bahwa *user* hanya menggunakan *resources* RF hanya ketika mengirim dan menerima data. Ketika seorang *user* tidak mengirim atau menerima data, *timeslot* pada *interface* udara akan dapat digunakan oleh *user* lain. Misalnya, ketika seorang *user* membuka halaman web, data hanya ditransfer ketika halaman web *di-request* atau dikirim, tidak pada saat *user* melihat-lihat isinya.

EDGE mempunyai tujuan utama mempertinggi kemampuan *throughput* data jaringan GSM atau menyediakan keluaran *bits per second* yang lebih besar hingga delapan *timeslot* TDMA. Hal ini dilakukan dengan merubah skema modulasi *interface* udara dari *Gaussian Minimum Shift Keying* (GMSK) yang digunakan pada GSM menjadi *8 Phase Shift Keying* (8-PSK) yang secara teori dapat memberikan hasil kecepatan transfer 384 Kbps.

CDMA 2000 phase 1 juga dikategorikan sistem 2.5G karena tidak menyediakan semua spesifikasikebutuhan IMTS-2000, sebuah badan yang mengembangkan sistem 3G, yaitu kebutuhan mobilitas penuh

2.4.4. Generasi III (3G)

3G adalah generasi ketiga teknologi *wireless* yang dikembangkan untuk tujuan komunikasi data dengan kecepatan transfer yang tinggi. 3G didefinisikan dalam spesifikasi *International Mobile Telecommunications-2000*(IMT-2000).Spesifikasi tersebut antara lain standar global, kualitas tinggi, terminal yang kecil, layanan aplikasi multimedia, meningkatkan efisiensi spektrum, kecepatan transfer yang tinggi (hingga 2 Mbps), dan lain sebagainya. Teknologi 3G diantaranya *Universal Mobile Telecommunication Service* (UMTS), *Wideband* CDMA (WCDMA), dan CDMA2000.

UMTS menggunakan arsitektur jaringan yang sama dengan GSM/GPRS dengan beberapa pengembangan. UMTS menyediakan layanan transfer data kecepatan tinggi hingga 2 Mbps. *Interface* udara dari UMTS

adalah *Direct Sequence* CDMA (DS-CDMA) yang merupakan perubahan yang radikal dari teknik TDMA GSM/GPRS. Dengan DS-CDMA, data *user* disebarkan melalui *bandwidth* yang lebih lebar dengan jalan perkalian dengan sebuah urutan bit *pseudo-random* yang disebut *chips*.

Sedangkan WCDMA dan CDMA2000 sama-sama menggunakan teknologi CDMA. Dalam proses mencapai level *throughput* yang dibutuhkan, WCDMA menggunakan *channel wideband*, sementara CDMA2000 menggunakan baik *channel wideband* maupun *narrowband*. Baik WCDMA dan CDMA2000 didesain untuk beroperasi pada beberapa frekuensi

2.4.5. Generasi Transisi (3.5G)

Generasi 3.5G merupakan pengembangan dari 3G yang memungkinkan pengiriman data lebih cepat. Perbandingan antara 3G dan 3.5G terlihat jelas pada kecepatan transmisinya. Pada 3G kecepatan transmisi maksimal mencapai 3.6Mbps. generasi 3G dan 3.5G mendukung layanan *video call* yang memungkinkan penelpon dan penerima saling bertatap muka.

2.4.6. Generasi IV (4G)

Belakangan ini industri nirkabel mulai mengembangkan 4G, meskipun sebenernya teknologi 4G ini seperti *Long Term Evolution* (LTE) hanya merupakan evolusi dari teknologi 3GPP dan *Ultra Mobile Broadband* (UMB) berasal dari 3GPP2, sehingga sulit untuk membedakan dengan jelas teknologi 3G dan 4G. salah

satu teknologi 4G yaitu WiMax *mobile standard* telah diterima oleh ITU untuk ditambahkan pada IMT-2000, sehingga teknologi baru ini masih digolongkan ke dalam keluarga 3G. *International Telecommunication Union* (ITU) sedang mempelajari kemampuan *mobile broadband* yang disebut IMT-*advanced* yang disebut teknologi generasi keempat (4G). diharapkan ITU segera melaksanakan penggunaan IMT-2000 (3G) dan IMT-*Advanced* (4G), konsekuensinya ITU harus menambah pita baik dibawah 1 Ghz maupun diatas 2Ghz.

2.5. Global Positioning System (GPS)

2.5.1. Sistem Navigasi GPS

GPS adalah sistem radio navigasi dan penentuan posisi dengan menggunakan satelit navigasi yang dimiliki dan dikelola oleh Departemen Pertahanan Amerika Serikat. Nama formalnya adalah **NAVSTAR GPS** (*Navigation Satellite Timing and Ranging Global Positioning System*). Sistem ini digunakan untuk memberikan informasi mengenai posisi, waktu dan kecepatan kepada siapa saja secara global tanpa ada batasan waktu dan cuaca. Satelit GPS pertama diluncurkan pada tahun 1978 dan secara resmi sistem GPS dinyatakan operasional pada tahun 1994.



Gambar 2.1. Satelit GPS

2.5.2. Segmen Penyusun Sistem GPS

Sistem GPS tediri atas tiga segmen utama, yaitu segmen angkasa (*space segment*), segmen sistem kontrol (*control system segment*), dan segmen pengguna (*user segment*). [4].

2.5.2.1. Segmen Angkasa

Segmen angkasa terdiri dari 24 buah satelit GPS yang secara kontinyu memancarkan sinyal – sinyal yang membawa data kode dan pesan navigasi yang berguna untuk penentuan posisi, kecepatan dan waktu. Satelit-satelit tersebut ditempatkan pada enam bidang orbit dengan periode orbit 12 jam dan ketinggian orbit 20.200 km di atas permukaan bumi. Keenam orbit tersebut memiliki jarak spasi yang sama dan berinklinasi 55o terhadap ekuator dengan masing-masing orbit ditempati oleh empat buah satelit dengan jarak antar satelit yang tidak sama.

2.5.2.2. Segmen Sistem Kontrol

Segmen sistem kontrol terdiri dari *Master Control Station* (MCS), *Ground Station,* dan beberapa *Monitor Station* (MS) yang berfungsi untuk mengontrol dan memonitor pergerakan satelit.

2.5.2.3. Segmen Pengguna

Segmen pengguna terdiri dari para pengguna satelit GPS baik yang ada di darat, laut maupun udara Dalam hal ini *receiver* GPS dibutuhkan untuk menerima dan memproses sinyal-sinyal dari GPS untuk digunakan dalam penentuan posisi, kecepatan, dan waktu.

2.5.3. Sinyal GPS

Sinyal GPS yang dipancarkan oleh satelit-satelit GPS menggunakan *band* frekuensi L pada spektrum gelombang elektromagnetik. Setiap satelit GPS memancarkan dua (2) gelombang pembawa yaitu L1 dan L2 yang berisi data kode dan pesan navigasi. Pada dasarnya sinyal GPS terdiri dari tiga komponen, yaitu: penginformasi jarak (kode), penginformasi posisi satelit (*navigation message*) dan gelombang pembawanya (*carrier wave*). [4].

2.5.3.1. Penginformasi Jarak

Penginformasi jarak yang dikirimkan oleh satelit GPS terdiri dari dua buah kode PRN (*Pseudo Random Noise*) yaitu kode-C/A (*Coarse Acquisition/Clear Access*) yang dimodulasikan pada gelombang pembawa L1 dan kode-P(Y) (*Private*) yang dimodulasikan baik pada gelombang pembawa L1

maupun L2. Kedua kode tersebut disusun oleh rangkaian kombinasi bilangan-bilangan biner (0 dan 1). Setiap satelit GPS mempunyai struktur kode yang unik dan berbeda antara satu satelit dengan satelit lainnya yang memungkinkan *receiver* GPS untuk membedakan sinyal- sinyal yang datang dari satelit-satelit GPS yang berbeda. Sinyal-sinyal tersebut dapat dibedakan oleh *receiver* dengan menggunakan teknik yang dinamakan CDMA (*Code Division Multiple Accsess*). [4].

2.5.3.2. Penginformasi Posisi

Pesan navigasi yang dibawa oleh sinyal GPS terdiri dari informasi *ephemeris* (orbit) satelit yang biasa disebut *broadcast ephemeris* yang terdiri dari parameter waktu, parameter orbit satelit dan parameter perturbasi dari orbit satelit. [4]. Parameter – parameter tersebut digunakan untuk menentukan koordinat dari satelit. Disamping *broadcast ephemeris* , pesan navigasi juga berisi *almanac* satelit yang memberikan informasi tentang orbit nominal satelit yang berguna bagi *receiver* dalam proses akuisasi awal data satelit maupun bagi para pengguna dalam perencanaan waktu pengamatan yang optimal. [4]. Informasi lain yang dibawa oleh pesan navigasi adalah koefisien koreksi jam satelit, parameter koreksi ionosfer, status konstelasi satelit dan informasi kesehatan satelit.

2.5.3.3. Gelombang Pembawa

Kode dan pesan navigasi agar dapat mencapai pengamat harus dimodulasikan terlebih dahulu pada gelombang pembawa. Gelombang pembawa yang digunakan terdiri atas dua gelombang , yaitu

gelombang *L1* dan *L2*. Gelombang L1 (1575.42 Mhz) membawa kode-P(Y) dan kode-C/A sedangkan gelombang L2 *(*1227.60 Mhz*)* hanya membawa kode-P(Y) saja. Teknik modulasi yang digunakan dalam sinyal GPS adalah BPSK (*Binary Phase Shift Keying*) yang menggunakan modulasi fase. [4].

2.5.4. Penentu Posisi Absolut dengan GPS

Penentuan posisi dengan GPS adalah penentuan posisi tiga dimensi yang dinyatakan dalam sistem koordinat kartesian (X,Y,Z) dalam datum WGS (*World Geodetic System)* 1984. Untuk keperluan tertentu, koordinat kartesian tersebut dapat dikonversi ke dalam koordinat geodetik (φ,λ,h). Titik yang akan ditentukan posisinya dapat diam (*static positioning*) maupun bergerak (*kinematic positioning*). Penentuan posisi absolut merupakan metode penentuan posisi yang paling mendasar dan paling banyak digunakan untuk aplikasi-aplikasi yang tidak memerlukan tingkat ketelitian posisi yang tinggi dan tersedia secara instant *real-time*) seperti pada aplikasi navigasi wahana bergerak (darat, laut dan udara).

2.5.4.1. Prinsip Penentu Posisi Absolut dengan GPS

Prinsip dasar penentuan posisi absolut dengan GPS adalah dengan reseksi jarak ke beberapa satelit GPS sekaligus yang koordinatnya telah diketahui. [4]. Pada penentuan posisi absolut dengan data *pseudorange*, jarak pengamat (*receiver*) ke satelit GPS ditentukan dengan mengukur besarnya waktu tempuh sinyal GPS dari satelit ke *receiver* pengamat.Waktu tempuh ditentukan

dengan menggunakan teknik korelasi kode (*code correlation technique*) dimana sinyal GPS yang datang dikorelasikan dengan sinyal replika yang

diformulasikan dalam *receiver*. Jarak dari *receiver* ke pengamat kemudian dapat ditentukan dengan mengalikan waktu tempuh dengan kecepatan cahaya. Karena ada perbedaan waktu pada jam satelit dan jam *receiver* maka data jarak yang diperoleh bukan merupakan jarak yang sebenarnya melainkan jarak *pseudorange.*

2.5.4.2. Ketelitian Posisi Absolut

Ketelitian posisi absolut GPS sangat bergantung pada tingkat ketelitian data *pseudorange* serta geometri dari satelit pada saat pengukuran. [4]. Ketelitian posisi GPS = Geometri Satelit x Ketelitian *Pseudorange* Faktor – faktor yang mempengaruhi ketelitian penentuan posisi dengan GPS adalah sebagai berikut:

a. Satelit, seperti kesalahan *ephemeris* (orbit) dan jam satelit.
b. Medium propagasi, seperti ionosfer dan bias troposfer yang mempengaruhi kecepatan (memperlambat) dan arah perambatan sinyal GPS.
c. *Receiver* GPS seperti kesalahan jam *receiver*, kesalahan yang terkait dengan antenna dan *noise* (derau). Kesalahan-kesalahan ini bergantung pada kualitas dari *receiver* GPS dan berbanding lurus dengan harga dari *receiver* GPS, semakin tinggi harga *receiver*, semakin tinggi kualitasnya.
d. Lingkungan sekitar *receiver* GPS, seperti *multipath* yaitu fenomena dimana sinyal GPS yang tiba di antenna *receiver* GPS merupakan resultan dari sinyal yang langsung dari GPS dan sinyal yang dipantulkan oleh benda-benda di sekeliling *receiver* GPS.

Dalam kaitannya dengan ketelitian penentuan posisi dengan GPS, terdapat dua level ketelitian yang diberikan oleh GPS, yaitu SPS (*Standard*

Positioning Service) dan PPS (*Precise Positioning Service*). SPS merupakan layanan standar yang diberikan oleh GPS kepada siapa saja tanpa dipungut biaya. Tingkat ketelitian yang diberikan oleh layanan ini adalah ± 100 m pada saat kebijakan SA (*Selective Availability*) masih berlaku dan ± 20 m setelah kebijakan SA dihapus (1 Mei 2000, 00:00 EDT). Sedangkan PPS merupakan jenis layanan yang hanya dikhususkan untuk pihak militer Amerika dan pihak-pihak lain yang diizinkan dengan tingkat ketelitian yang lebih tinggi dari tingkat ketelitian SPS.

2.5.5. Format Data Keluaran GPS.

Receiver GPS memiliki format keluaran sebanyak lima (5) jenis yaitu NMEA 0180, NMEA 0182, NMEA 0183, AVIATION, dan PLOTTING. [4]. Format data tersebut ditetapkan oleh NMEA (*National Maritime Electronic Association*) dan dapat dikoneksikan ke komputer melalui *port* komunikasi *serial* dengan menggunakan kabel RS-232. Data keluaran dalam format NMEA 0183 berbentuk kalimat (s*tring*) yang merupakan rangkaian karakter ASCII 8 *bit*. Setiap kalimat diawali dengan satu karakter '$' , dua karakter *Talker ID*, tiga karakter *Sentence ID*, dan diikuti oleh *data fileds* yang masing-masing dipisahkan oleh koma serta diakhiri oleh *optional cheksum* dan karakter *cariage return/line feed* (CR/LF). Jumlah maksimum karakter dihitung dari awal kalimat ($) sampai dengan akhir kalimat (CR/LF) adalah 82 karakter. Format dasar data NMEA 0183:

NMEA Output

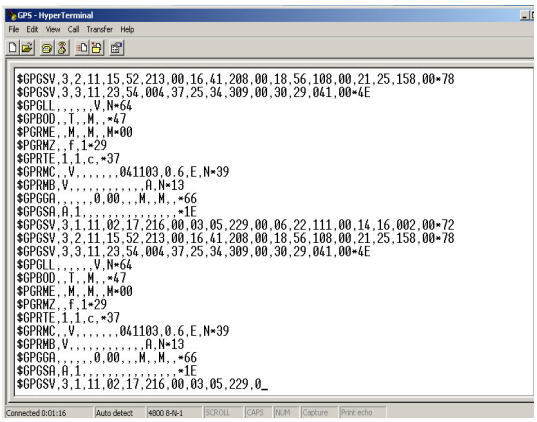
Output dari pesawat penerima GPS memiliki struktur tertentu yang disebut NMEA output. Contoh struktur output beserta cara pembacaannya yaitu sebagai berikut :

$GPRMC,180432,A,4027.027912,N,08704.857070,W,000.04,181.9,131000,1.8,W,D*25, esi4001

Tabel 2.1. Tabel Nmea Output GPRMC

Bidang Data	Nilai	Deskripsi
1	180432	UTC dari penentuan posisi dalam format hhmmss.ss (18jam, 4 mnt dan 32 detik)
2	A	Status (A-data valid, V-peringatan)
3	4027.027912	Latitude geografis dalam format ddmm.mmmmm.m (40 derajat dan 27.027912 menit)
4	N	Arah dari latitude (N–North, S–South)
5	08704.857070	Longitude geografis dalam format dddmm.mmmmm.m(87 derajat dan 4.85707 menit)
6	W	Arah dari longitude (E–East, W–West)
7	000.04	Kecepatan diatas tanah (0.04 knot)
8	181.9	Track made good (heading) (181.9º)
9	131000	Tanggal dalam format ddmmyy (Oktober 13, 2000)
10	1.8	Variasi kompas(1.8º)
11	W	Arah dari variasi kompas (E-East, WWest)

12	D	Mode indikasi (A-autonomous, Ddifferensial, N-data tidak valid)
13	esi4001	Id perangkat yang disertakan pada saat penyetingan oleh installer

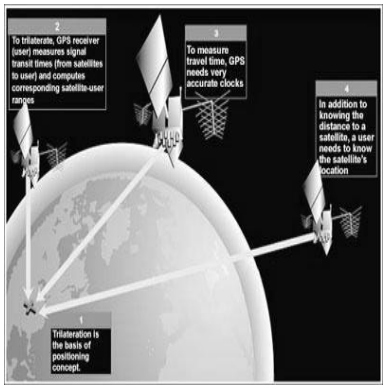


Gambar 2.2 Keluaran NMEA 0183 dari receiver GPS yang teridentifikasi pada Hyperterminal

2.5.6. Cara kerja GPS

Navigasi satelit didasarkan pada jaringan global satelit yang mengirimkan sinyal radio di orbit bumi menengah. Pengguna Navigasi satelit paling akrab dengan 24 Global Positioning System (GPS) satelit. Amerika Serikat, yang mengembangkan dan mengoperasikan GPS, dan Rusia, yang mengembangkan sistem serupa yang dikenal sebagai GLONASS, telah menawarkan sistem bebas menggunakan masing- masing kepada masyarakat internasional. Internasional

Organisasi Penerbangan Sipil (ICAO), serta kelompok-kelompok pengguna internasional, telah menerima GPS dan GLONASS sebagai inti untuk kemampuan navigasi satelit sipil internasional dikenal *sebagai Global Navigation Satellite System* (GNSS).



Gambar 2.3 Cara Kerja GPS

Layanan GPS dasar menyediakan pengguna dengan akurasi sekitar 100 meter, 95% dari waktu, di mana saja pada atau dekat permukaan bumi. Untuk mencapai hal ini, masing-masing 24 satelit memancarkan sinyal ke alat penerima yang menentukan lokasi mereka dengan menghitung perbedaan antara waktu sinyal yang dikirim dan diterima waktu membawa. GPS satelit jam atom yang menyediakan waktu yang sangat akurat.

Waktu Informasi ditempatkan dalam kode yang disiarkan oleh satelit sehingga penerima dapat terus menentukan waktu sinyal disiarkan. Sinyal berisi data yang penerima digunakan untuk menghitung lokasi satelit dan melakukan penyesuaian lainnya yang diperlukan untuk penentuan posisi

yang akurat. Penerima menggunakan perbedaan waktu antara waktu penerimaan sinyal dan waktu siaran untuk menghitung jarak, atau jangkauan, dari receiver ke satelit. Penerima harus rekening atas keterlambatan propagasi, atau penurunan kecepatan sinyal yang disebabkan oleh ionosfer dan troposfer. Dengan informasi tentang berkisar di tiga satelit dan lokasi satelit ketika sinyal dikirim, penerima dapat menghitung sendiri dimensi tiga posisi sinyal. Atom Jam disinkronisasi ke GPS diperlukan untuk menghitung berkisar antara ketiga. Namun, dengan mengambil pengukuran dari satelit keempat, penerima menghindari kebutuhan untuk jam atom. Dengan demikian, penerima menggunakan empat satelit untuk menghitung lintang, bujur, ketinggian, dan waktu.

2.6. Java

2.6.1. Pengertian Java

Java adalah bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan-rekannya seperti Patrick Naugton, Chris Warth, Ed Frank, dan Mike Sheridan di suatu perusahaan perangkat lunak yang bernama Sun Microsystem, pada tahun 1991. Bahasa pemrograman ini mula-mula diinisialisasi dengan nama "oak", namun pada tahun 1995 diganti namanya menjadi "Java"

Alasan utama pembentukan bahasa java adalah untuk membuat aplikasi-aplikasi yang dapat diletakkan diberbagai macam perangkat elektronik, seperti *microwave oven* dan *remote control,* sehingga Java harus bersifat *portable* atau yang sering disebut dengan *platform independent* (tidak tergantung pada *platform*). Itulah yang menyebabkan dalam dunia pemrograman Java, dikenal adanya istilah '*write*

once, run everywhere', yang berarti kode program hanya ditulis sekali, namun dapat dijalankan dibawah *platform* manapun, tanpa harus melakukan perubahan kode program.

2.6.1.1. Arsitektur Java

Secara arsitektur, Java tidak berubah sedikitpun semenjak awal mula bahasa tersebut dirilis. Kompiler Java (yang disebut dengan **Javac** atau *Java Compiler*) akan mentransformasikan kode-kode dalam bahsa Java ke dalam suatu *bytecode*. Apa itu *bytecode*? *Bytecode* adalah sekumpulan perintah hasil kompilasi yang kemudian dapat dieksekusi melalui sebuah mesin komputer abstrak, yang disebut dengan JVM (*Java Virtual Machine*). JVM juga sering dinamakan sebagai *interpreter*, kiarena sifatnya yang selalu menerjemahkan kode-kode yang tersimpan dalam *bytecode* dengan cara baris demi baris.

2.6.1.2. Java Versi Lama (Java 1)

Pada awal perilisannya, versi Java masih disebut dengan JDK (*Java Development kita*). Dalam JDK, semua kebutuhan untuk pengembangan program dan eksekusi program masih tergabung jadi satu. Penamaan ini berlaku sampai Java 1.1. namun sekarang, setelah Java 1.2, Sun Microsystem menamainya dengan JSDK (*Java Software Development Kit*) dalam hal ini kebutuhan untuk pengembangan program dipisahkan dengan kebutuhan kesekusi. Bagian *software* yang digunakan

untuk kebutuhan eksekusi program disebut dengan JRE (*Java-Runtime Environment*). Selanjutnya, Java 1.2 disederhanakan penamaanya menjadi "Java 2"

2.6.1.3. Java 2

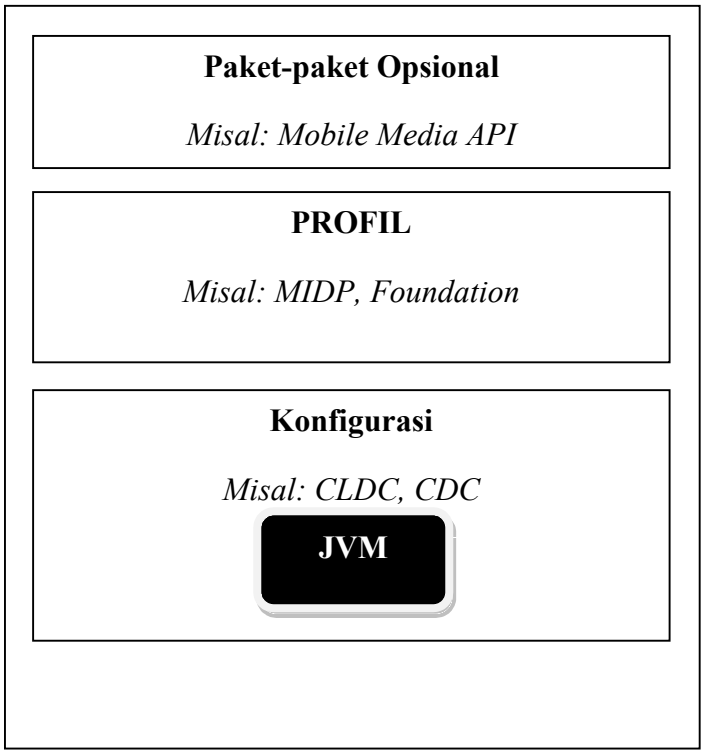
Sun Microsystems telah mendefinisikan tiga buah edisi dari Java 2, yaitu sebagai berikut :

a. Java 2 Standard Edition (J2SE), yang digunakan untuk mengembangkan aplikasi-aplikasi desktop dan applet (aplikasi Java yang dapat dijalankan di dalam *browser web*).
b. Java 2 Enterprise Edition (J2EE), merupakan *superset* dari J2SE yang memperbolehkan untuk mengembangkan palikasi-aplikasi berskala besar (*enterprise*), yaitu dengan melakukan pembuatan aplikasi-aplikasi di sisi *server* dengan menggunakan EJBs (*Enterprise JavaBeans*), aplikasi web dengan menggunakan *Servlet* dan JSP (*JavaServer Pages*) dan teknologi lainnya serperti CORBA (*Common Object Request Broker Architecture*) dan XML (*Extensible Markup Language*).
c. Java 2 Micro Edition (J2ME), merupakan *subset* dari J2SE yang digunakan untuk menangani pemrograman di dalam perangkat-perangkat kecil, yang tidak memungkinkan untuk mendukung implementasi J2SE secara penuh

2.6.2. J2ME

J2ME merupakan sebuah kombinasi yang terbentuk antara kumpulan *interface* Java yang sering disebut dengan Java API (*Application programming interface*) dengan JVM (*Java Virtual Machine*) yang didesain khusus untuk alat, yaitu JVM dengan ruangan terbatas. Kombinasi tersebut kemudian digunakan untuk melakukan pembuatan aplikasi-aplikasi yang dapat berjalan di atas alat (dalam hal ini *mobile device*).

J2ME sendiri pada dasarnya tterdiri dari tiga buah bagian, yaitu konfigurasi, profil, dan paket-paket opsional, seperti yang ditunjukkan oleh gambar 2.4.



Gambar 2,4 Bagian-bagian di dalam platform J2ME

2.6.2.1. Konfigurasi

Konfigurasi merupakan bagian yang berisi JVM dan beberapa *library* kelas lainnya. Perlu diperhatikan bahwa JVM yang dimaksud disini bukanlah JVM tradisional seperti yang terdapat dalam J2SE, melainkan JVM yang sudah didesain secara khusus untuk alat.

Terdapat dua buah konfigurasi yang disediakan oleh Sun Microsystems, yaitu CLDC (*Connected Limited Device Configuration*) dan CDC (*Connected Device Configuration*). Target alat dari konfigurasi CLDC adalah alat-alat kecil, seperti telepon selular, PDA, dan *pager*. Membahas lebih jauh mengenai konfigurasi CLDC pada bagian tersendiri dalam bab ini. Pada sisi yang lain, CDC merupakan *superset* dari CLDC sehinga semua yang kelas yang didefinisikan di dalam CLDC aka nada juga didalam CDC.

2.6.2.2. Profil

Profil merupakan bagian perluasan dari konfigurasi. Artinya, selain sekumpulan kelas yang terdapat pada konfigurasi, terdapat juga kelas-kelas spesifik yang didefinisikan lagi di dalam profil. Dengan kata lain, profil akan membantu secara fungsional yaitu dengan mehyediakan kelas-kelas yang tidak terdapat di level konfigurasi.

Adapun profil yang sangat popular penggunaannya adalah profile yang disediakan oleh Sun Microsystems, yaitu yang dinamakan MIDP (*Mobile Information*

Device Profile). Berikut ini beberapa profil yang tersedia untuk kebutuhan-kebutuhan spesifik lainnya.

a. PDAP (*Personal DigitalAassistant Profile*) yaitu profil untuk PDA yang memperluas fungsi-fungsi pada konfigurasi CLDC dan digunakan khusus untuk menambahkan kemampuan-kemampuan lebih apabila dibandingkan dengan penggunaan profil MIDP.
b. *Foundation Profile*, yaitu profil yang digunakan untuk konfigurasi CDC. Profil ini menambahkan beberapa kelas dari J2SE ke dalam konfigurasi CDC, dan berperan juga sebagai pondasi untuk membentuk profil baru lainnya.
c. *Personal Profile*, yaitu profil yang mendefinisikan ulang *Personal* Java sebagai profil yang dapat digunakan sebagai profil dalam J2ME, Profil ini merupakan hasil perluasan dari *Foundation Profile*.
d. *RMI Profile*, yaitu profil yang menambahkan dukungan RMI (*Remote Method Invocation*) ke dalam konfigurasi CDC.

2.6.2.3. Paket-paket Opsional

Paket-paket opsional merupakan paket-paket tambahan yang dibutuhkan oleh aplikasi sehingga pada saat proses *deployment* paket-paket tersebut perlu didistribusikan juga sebagai bagian dari aplikasi bersangkutan. Sebagai catatan bahwa paket-paket opsional ini bukan merupakan paket yang dibuat oleh perusahaan alat yang digunakan.

2.7. Android

Pembahasan mengenai android dan *tools* yang digunakan dalam pembuatan aplikasi, akan dijelaskan pada sub bab berikut.

2.7.1. Sejarah Android

Android adalah sebuah sistem operasi untuk perangkat lunak *mobile* berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, Htc, Intel, Motorola, Qualcomm, T-*Mobile*, dan Nvidia.

Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di lain pihak, Google merilis kode-kode android dibawah lisensi *Apache*, sebuah lisensi perangkat lunak dan open *platform* perangkat seluler.

Didunia ini terdapat dua jenis distributor sistem operasi android. Pertama yang mendapat dukungan penuh dari Google atau Google Mail *Service*s (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai Open Handset Distribution (OHD).

Sekitar September 2007 Google mengenalkan Nexus One, salah satu jenis *smartphone* yang menggunakan android sebagai sistem operasinya. Telpon seluler ini diproduksi oleh HTC Corporation dan tersedia di pasaran pada 5 Januari 2010. Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android AEM Holdings, Atheros Communication, diproduksi oleh Asustek Komputer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan Open Handset Alliance, OHA mengumumkan produk perdana mereka, android perangkat *mobile* yang merupakan modifikasi kernel Linux 2.6. sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Pada masa saat ini kebanyakan vendor-vendor *smartphone* sudah memproduksi *smartphone* berbasis android, vendor-vendor itu antara lain HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, LG, Acer, Philips, T-*Mobile*, Nexian, IMO, Asus dan masih banyak lagi vendeor *smartphone* didunia yang memproduksi android. Hal ini karena android adalah sistem oerasi yang *open source* sehingga bebas didistribusikan dan dipakai oleh vendor manapun.

Tidak hanya menjadi sistem operasi di *smartphone*, saat ini android menjadi pesaing utama dari Apple pada sistem operasi Table PC. Pesatnya pertumbuhan Android selain faktor yang disebutkan diatas adalah karena android itu sendiri adalah *platform* sangat lengkap baik itu sistem oeprasinya, aplikasi dan Tool Developmen,

Market aplikasi android serta dukungan yang sangat tinggi dari komunitas *Open source* didunia, sheingga android terus berkembang pesat dari segi teknologi maupun dari segi jumlah device yang ada didunia.

2.7.2. *The Dalvik Virtual Machine* (DVM)

Salah satu elemen kunci dari android adalah *Dalvik Virtual Machine* (DVM). Android berjalan di *Dalvik Virtual Machine* (DVM) bukan di Java *Virtual Machine* (JVM), sebernarnya banyak persamaan dengan Java *Virtual Machine* (JVM) seperti Java ME (*Java Mobile Edition*), tetapi android menggunakan *Virtual Machine* sendiri yang diskostumisasi dan dirancang untuk memastikan bahwa beberapa fitur-fitur berjalan lebih efisien pada perangkat *mobile*.

Dalvik Virtual Machine (DVM) adalah "*register bases*" sementara java *Virtual Machine* (JVM) bersifat "*stack based*", DVM didesain dan ditulis oleh Dan Bornsten dan beberapa enginers Google lainnya. Jadi bisa dikatakan "*Dalvik* equal(Java) == False" *Dalvik Virtual Machine* menggunakan kernel linux untuk menangani fungsionalitas tingkat rendah termasuk keamanan, *threading*, dan proses serta manajemen memori. Ini memungkinkan kita untuk menulis aplikasi C/C++ sama halnya seperti pada OS Linux kebanyakan. Meskipun dalam kenyataannya kita harus banyak memahami Arsitektur dan proses sistem dari kernel linux yang digunakan dalam Android tersebut.

Semua *hardware* yang berbasis android dijalankan dengan menggunakan *Virtual Machine* untuk eksekusi aplikasi, pengembang tidak perlu khawatir tentang implementasi pernagkat keras tertentu. *Dalvik Virtual Machine* mengeksekusi *Executable* file, sebuah format yang dioptimalkan untuk memastikan memori yang digunakan sangat kecil. *The Executable* file diciptakan dengan mengubah kelas bahasa java dan dikompilasi menggunakan *tools* yang disediakan dalam SDK Android.

2.7.3. Arsitektur Android

Secara garis besar arsitektur android dapat dijelas dan digambarkan sebagai berikut :

a. Application and Widgets

Application and widgets adalah layer dimana berhubungan dengan aplikasi dan biasanya download aplikasi kemudian lakukan instalasi dan jalankan aplikasi tersebut, delayer inilah terdapat seperti aplikasi inti termasuk klien email, program SMS, kalender, peta, *browser*, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

b. Application Frameworks

Android adalah "*Open Development Platform*" yaitu android menawarkan kepada pengembang atau member kemampuan kepada pengembangan untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk

mengakses perangkat keras, akses informasi *resources*, menjalankan serive *background*, mengatur alarm, dan menambahkan tambahan seperti status *notification*s dan masih banyak lagi. Pengembang memiliki akses penuh menuju API *Framework* seperti yang dilakukan oleh aplikasi yang kategori inti. Arsitektur aplikasi dirancang supaya dengan mudah dapat menggunakan komponen yang sudah digunakan (*reuse*).

Sehingga bisa disimpulkan Application Framework adalah layer dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem android, karena pada layer inilah aplikasi dapat dirancang, seperti *content-providers* yang berupa sms dan lain sebagainya.

Komponen-komponen yang termasuk didalam *application Framework* adalah sebagai berikut :

1) *Views*
2) *Content Provider*
3) *Resource Manager*
4) *Notification Manager*
5) *Activity Manager*

c. *Libraries*

Libraries adalah layer dimana fitur-fitur android berada biasanya para pembuat aplikasi kebanyakan mengakses *libraries* untuk menjalankan aplikasinya.

Berjalan diatas kernel, layer ini meliputi berbagai library C/C++ inti seperti Libc dan SSL, serta :

1) *Libraries* media untuk pemutar media audio dan video
2) *Libraries* untuk manajemen tampilan
3) *Libraries Graphics* mencakup SGL dan OpenGL untuk grafis 2D dan 3D
4) *Libraries SQLite* untuk dukungan database
5) *Libraries* SSL dan *WebKit* terintegrasi dengan web *browser* dan security
6) *Libraries Live Webcore* mencakup modern web *browser* dengan engine embedded web view

d. Android Run Time

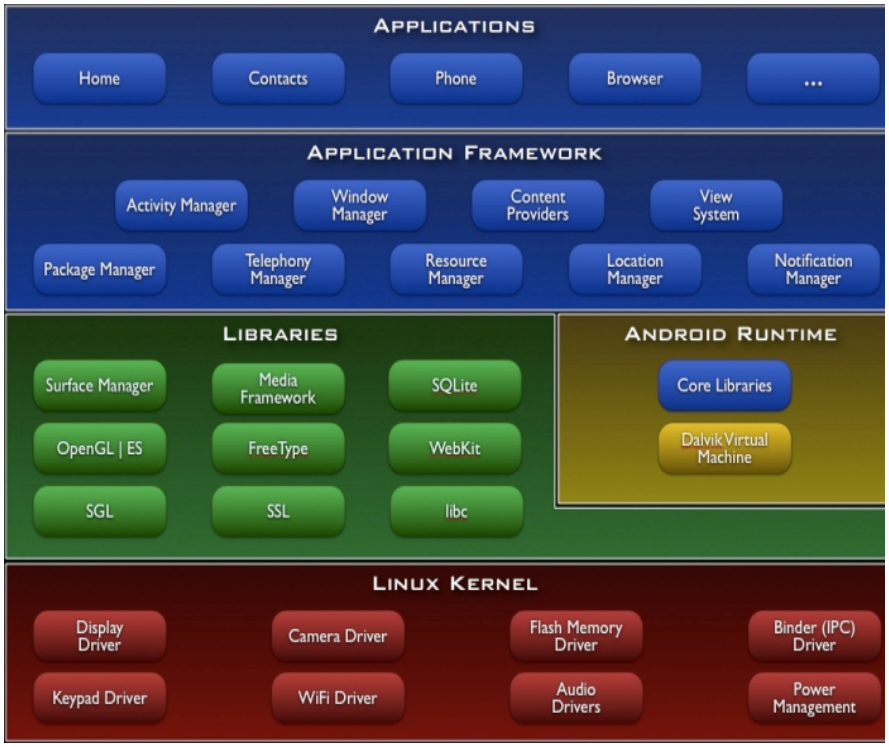
Layer yang membuat aplikasi android dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Didalam *Android Run Time* dibagi menjadi dua bagian yaitu:

1) *Core Libraries* : aplikasi android dibangun dalam bahasa java, sementara *Dalvik* sebagai virtual mesin bukan *Java Virtual Machine*, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa Java/C yang dihandle oleh Core *Libraries*
2) *Dalvik Virtual Machine* : Virtual mesin yang berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien dimana merupakan

pengembangan yang mampu membuat linux kernel untuk *threading* dan manajemen tingkat rendah.

3) Linux kernel

Linux kernel adalah layer dimana inti dari operating sistem dari android itu sendiri, berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resources*, *drivers*, dan sistem-sistem operanting andoroid lainnya. Linux kernel yang digunakan android adalah linux kernel *release* 2.6.



Gambar 2.5 Arsitektur Andoroid

2.7.4. Fundamental Aplikasi

Aplikasi android ditulis dalam bahasa pemrograman java, kode java dikompilasi bersama data file *resource* yang dibutuhkan oleh aplikasi dimana prosesnya dipaket oleh *tools* yang dinamakan "apt *tools*" ke dalam paket android sehingga menghasilkan file dengan ekstensi apk. File apk itulah yang sebenernya disebut dengan aplikasi yang dapat diinstal di perangkat *mobile* nantinya.

Ada enam jenis komponen pada aplikasi android yaitu:

a. Activities

Suatu *activity* akan menyajikan *user interface* (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi android bisa jadi hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity* tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu *activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai *user interface* (UI) saat aplikasi diperlihatkan kepada *user*. Untuk pindah dari satu *activity* ke *activity* lain dapat dilakukan dengan satu even misalnya click tombol, memilih opesi atau menggunakan triggers tertentu. Secara hirarki sebuah window *activity* dinyatakan dengan method *Activity*.setContentView(). ContentView adalah objek yang berada pada root hirarki.

b. Service

Service tidak memiliki visual *user interface* (UI), tetapi *service* berjalan secara *background*, sebagai contoh dalam memainkan music, *service* mungkin memainkan music atau mengambil data dari jaringan, tetapi setiap *service* haruslah berada dalam kelas induknya. Misalnya media player sedang memutar lagu dari *list* yang ada. Aplikasi ini akan memiliki dua atau lebih *activity* yang memungkinkan *user* untuk memilih lagu atau menulis sms sambil *player* sedang jalan untuk menjaga *music* tetapi dijalankan, *activity player* dapat menjalankan *service* untuk membuat aplikasi tetap berjalan. *Service* dijalankan pada *thread* utama dari proses aplikasi.

c. Broadcast Receiver

Broadcast Receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh *broadcast* seperti notifikasi zona waktu berubah, baterai *low*, gambar telah selesai diambil oleh kamera, atau pengubah referensi bahasa yang digunakan. Aplikasi juga dapat menginisiasi *broadcast* misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah didownload ke perangkat dan siap untuk digunakan

Broadcast Receiver tidak memiliki *user interface* (UI) tetapi memiliki sebuah *activity* untuk merespon informasil yang mereka terima atau mungkin

menggunakan *notification* manager untuk memberitahu kepada pengguna seperti lampu latar atau *viberating* (getaran) perangkat dan lain sebagainya.

d. Content Provider

Content Provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database SQLite.*Content Provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity*, misalnya ketika menggunakan aplikasi yang membutuhkan peta (MAP) atau aplikasi yang membutuhkan untuk mengkases data kontak dan navigasi, maka disinilah fungsi *content provider*.

2.7.5. Versi Android

Telepon pertama yang memakai sistem operasi android adalah HTC Dream yang dirilis pada 22 Oktober 2008. Pada penghujung 2010 diperkirakan hamper semua vendor seluler didunia menggunakan android sebagai operating sistem. Adapun versi-versi android yang pernah dirilis adalah sebagai berikut :

a. Android versi 1.1

Pada 9 Maret 2009, Google merilis Android versi 1.1. android versi ini dilengkapi dengan pembaruan estis pada aplikasi, jam, *alarm*, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

b. Android versi 1.5 (*cupcake*)

Pada pertengahan 2009, google kembali merilis telepon seluluer dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengupload video Youtube dan gambar ke Picasa langsung dari telepon, dukungan *Bluetooth* A2DP, kemampuan terhubung secara otomatis ke headset *Bluetooth*, dan *keyboard* pada layer yang dapat disesuaikan oleh sistem.

c. Android versi 1.6 (*Donut*)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik disbanding sebelumnya, pengguna *baterai indicator* dan *control applet* VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, kamera, camcorder dan galeri yang diintegrasikan, CD/EVDO, 802.1x, VPN, Gestures, dan *Text-to-speech engine*, kemampuan *dial* kontak, teknologi *text to change speech* (tidak tersedia pada semua ponsel), pengadaan resolusi VWGA.

d. Android versi 2.0/2.1 (*Eclai*r)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (*Éclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3,1,2, perubahan UI dengan *browser* baru dan

dukungan HTML 5, daftar kontak yang baru, dukungan *flash* untuk kamera 3.2 MP, *digital Zoom*, dan *Bluetooth* 2.1.

e. Android versi 2.2 (Froyo: *Frozen Yoghurt*)

Pada bulan Mei 2010 android 2.2 Rev 1 diluncurkan. Android inilah yang sangat banyak sekarang dipasaran, salah satunya adalah dipakai di Samsung FX tab yang sudah ada dipasaran. Fitur yang tersedia di android versi ini sudah kompleks sekali diantaranya adalah :

1) Kerangka aplikasi memungkinkan penggunaan dan penghapusan komponen yang tersedia
2) *Dalvik Virtual Machine* dioptimalkan untuk perangkat *mobile*.
3) Grafik: grafik di 2D dan 3D berdasarkan *libraries* OpenGL
4) SQLite: untuk penyimpanan data.
5) Mendukung media: audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6) GSM, *Bluetooth*, EDGE, 3G, dan WiFi (tergantung *Hardware*)
7) Kamera, *Global Positioning System* (GPS), kompas, dan accelerometer (tergantung *Hardware*)

f. Android versi 2.3 (*Gingerbread*)

Android versi 2.3 diluncurkan pada desember 2010, hal-hal yang direvisi dari versi sebelumnya adalah kemampuan sebagai berikut :

1) SIP-*based* VoIP
2) *Near Field Communications* (NFC)
3) *Gyroscope* dan sensor
4) *Multiple cameras support*
5) *Mixable audio effects*
6) *Download manager*

Itulah beberapa versi android yang sudah dirilis sampai sekarang, kemungkinan besar versi tersebut akan terus berkembang seiring dengan kebutuhan yang sangat kompleks dibidang penggunaan *smartphone*. Untuk mendapatkan informasi versi terbaru android dapat dilihat di www.developer.android.com

2.8. *Tools-Tools* yang digunakan

2.8.1. Konsep Perancangan Berorientasi Obyek

Teknologi objek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi oleh objek. Didalam membangun sistem berorientasi objek akan menjadi lebih baik apabila langkah awalnya didahului dengan proses analisis dan perancangan yang berorientasi obke. Tujuannya adalah mempermudah programmer didalam mendesain program dalam bentuk objek-objek dan hubungan antar objek tersebut untuk kemudian dimodelkan dalam sistem nyata.

Suatu perusahaan software yaitu Rational Software, telah membentuk konsarium dengan berbagai organisasi untuk meresmikan pemakaian *Unified*

Modelling Language (UML) sebagai bahasa standar dalam *Object Oriented Analysist Design* (OOAD).

2.8.1.1. *Unified Modelling Language* (UML)

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek. Hal ini disebabkan UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan *design* ke dalam empat tahapan iterative, yaitu: identifikasi kelas-kelas dan obyek-obyek, identifikasi semantic dari hubungan obyek dan kelas tersebut, perincian *interface* dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur pemodelan entity-relationship. Tahapan utama dalam metodologi ini adalah analisis, *design* sistem, *design* obyek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih member penekanan pada use case. OOSE memiliki tiga tahapan yaitu membuat model requirement dan analisis, *design* dan

implementasi, dan model pengujian (*test model*). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Design UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen yang tidak praktis ditambah dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam dari pada metode lainnya.

2.8.1.2. Use Case Diagram

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use Case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut *scenario*. Setiap scenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat bisa dikatakan *use case* adalah serangkaian scenario yang digabungkan bersama-sama oleh tujuan umum pengguna.

Dalam pembicaraan tentang *use case*, pengguna biasanya disebut dengan aktor. Aktor adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.

Model *use case* adalah bagian dari model *requirement* (Jacob et all, 1992). Termasuk disini adalah problem domain *object* model dan penjelasan tentang *user*

interface. *Use case* memberikan spesifikasi fungsi-fungsi yang ditawarkan oleh sistem dari perspektif *user*.

2.8.1.3. Sequence Diagram

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh obyek dan *message* yang diletakkan diantara obyek-obyek ini di dalam *use case*. *Sequence* diagram menambahkan dimensi waktu pada interaksi diantara obyek. Pada diagram ini *participant* diletakkan di atas dan waktu ditunjukkan dari atas ke bawah. *Life line participant* diurutkan dari setiap *participant*. Kotak kecil pada *lifelin*e menyatakan *activation*, yaitu menjalankan salah satu operation dari *participant*. State bisa ditambahkan dengan menambahkannya sepanjang life line.

Message (sederhana, *synchronou*s atau *asynchronous*) adalah tanda panah yang menghubungkan suatu *life line* ke *life line* yang lain. Lokasi *life line* dalam dimensi vertikal mewakili urutan waktu dalam *sequence* diagram. *Message* yang pertama terjadi adalah yang paling dekat dengan bagian atas diagram dan yang terjadi belakangan adalah yang dekat dengan bagian bawah.

Pada beberapa sistem, operasi bisa dilakukan kepada dirinya sendiri. Hal ini disebut dengan rekursif. Untuk melukiskannya digunakan anak panah dari *activation* kembali ke dirinya sendiri, dan sebuah kotak kecil diletakkan pada bagian atas dari *activation*.

2.8.1.4. Collaboration Diagram

Collaboration Diagram adalah bentuk lain sequence diagram. Bila *sequence* diagram diorganisir menurut waktu maka *collabrotaion* diagram diorganisir menurut ruang/*space*.

Collaboration diagram merupakan asosiasi diantara obyek-obyek. Panah di dekat garis asosiasi menunjukkan *message*, sedangkan *content message* ditunjukkan dengan label. Angka pada *message* menunjukkan urutan *message*

Dengan *collaboration* diagram memungkinkan untuk memodelkan pengiriman sebuah *message* ke banyak obyek pada class yang sama. Demikian juga halnya untuk menunjukkan adanya obyek aktif yang mengendalikan aliran dari *message*.

2.8.1.5. *Activity* Diagram

Activity diagram seperti sebuah *flow chart*. *Activity* diagram menunjukkan tahapan, pengambilan keputusan dan percabangan. Diagram ini sangat berguna untuk menunjukkan *operation* sebuah obyek dan proses bisnis. Kelebihan *activity* diagram dibandingkan *flowchart* adalah kemampuannya dalam menampilkan aktivitas parallel.

2.8.1.6. Component Diagram

Component diagram merepresentasikan dunia *rill* item yaitu *component software*. *Component software* menetap di komputer bukan di benak para analis.

Component bisa diakses melalui *interface* nya yaitu koleksi operasi-operasi. Relasi antara *component* dan *interface*nya disebut *realization*. Suatu *component* bisa mengakses *service-service* yang ada di *component* lain dengan cara import *interface*. Sedangkan component yang menyediakan *service* menggunakan export *interface*.

Hal penting dari *component* adalah *component* mewakili potongan-potongan yang independen yang bisa dipesan dan diperbaharui sewaktu-waktu. Dengan demikian pembagian sistem ke dalam *component* lebih didorong dengan kepentingan *marketing* daripada kepentingan teknis.

2.8.1.7. Deployment Diagram

Deployement diagram menyediakan gambaran bagaimana sistem secara fisik akan terlihat. Sistem terdiri dari node-node dimana setiap node diwakili untuk sebuah kubus. Garis yang menghubungkan antara 2 kubus menunjukkan hubungan diantara kedua node tersebut. Tipe node bisa berupa *device* yang berwujud *hardware* dan bisa juga *processor* (yang mengeksekusi component) atau *execution environment* (software yang menjadi host atau mengandung software yang lain).

2.8.1.8. Class Diagram

Kotak adalah notasi UML untuk *class*. Nama, *attribute*, *operation* dan *responsibility* dari *class* ada pada kotak tersebut. *Stereotype* bisa dipergunakan untuk mengorganisasikan daftar *attribute* dan *operation*. Dalam beberapa kasus, kadangkala hanya perlu ditampilkan sebagian saja dari *attribute* dan *operation*.

Tipe *attribute* dan nilai default bisa dimunculkan sebagaimana pada *operation*. Untuk mengurangi ambiguitas pada pendeskripsian *class*, *constraint* bisa ditambahkan. Bahkan kalau perlu bisa ditambahkan *attached* notes ke dalam kotak tersebut.

2.8.2. Eclipse

2.8.2.1. Pengertian Eclipse

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*). Berikut ini adalah sifat dari Eclipse:

a. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
b. *Mulit-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.

c. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in*.

2.8.2.2. Arsitektur Eclipse

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah *kernel*, yang mengangkat *plug-in*. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari *plug-in* yang sudah diinstal. Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform* (RCP). Berikut ini adalah komponen yang membentuk RCP:

a. *Core platform*
b. OSGi
c. SWT (*Standard Widget Toolkit*)
d. JFace
e. *Eclipse Workbench*

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), *plug-in* yang membuat Eclipse kompatibel untuk mengembangkan program Java, dan PDE (*Plug-in Development Environment*) untuk mengembangkan *plug-in* baru. Eclipse beserta *plug-in*-nya diimplementasikan dalam bahasa pemrograman Java.

Konsep Eclipse adalah IDE yang terbuka (*open*), mudah diperluas (*extensible*) untuk apa saja, dan tidak untuk sesuatu yang spesifik[2]. Jadi, Eclipse tidak saja untuk mengembangkan program Java, akan tetapi dapat digunakan untuk berbagai macam keperluan, cukup dengan menginstal *plug-in* yang dibutuhkan. Apabila ingin mengembangkan program C/C++ terdapat *plug-in* CDT (*C/C++ Development Tools*). Selain itu, pengembangan secara visual bukan hal yang tidak mungkin oleh Eclipse, *plug-in* UML2 tersedia untuk membuat diagram UML. Dengan menggunakan PDE setiap orang bisa membuat *plug-in* sesuai dengan keinginannya.

2.8.3. Android SDK

Android SDK adalah *tools* API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh Google. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform* android menggunakan bahasa pemrograman Java. Sebagai *platform* aplikasi-netral,

android member anda kesempatan unutk membuat aplikasi yang kita butuhkan yang ubkan merupakan aplikasi bawaan Hadphone/*Smartphone*. Beberapa fitur-fitur android yang paling penting adalah :

a. *Framework* : aplikasi yang mendukung pengganti komponen dan reusable.
b. *Dalvik Virtual Machine* dioptimalkan untuk perangkat *mobile*
c. Integrated *Browser* verdasarkan engine *open source* WebKit.
d. Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional Ekselerasi *hardware*)
e. SQLite untuk penyimpanan data.
f. Media *Support* yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PING, GIF), GSM Telephony (tergantung *Hardware*)
g. *Bluetooth*, EDGE, 3G, dan WiFi (tergantung *hardware*)
h. Kamera, GPS, Kompas, dan *Accelerometer* (tergantung *hardware*)
i. Lingkungan *Development* yang lengkap dan termasuk pernagkat *emulator*, *tools* untuk *debugging*, profil dan kinerja memori, dan *plugin* untuk IDE Eclipse.

Untuk source SDK Android ini dapat dilihat dan didownload langsung di situs resmi pengembang SDK Android di http://www.developer.android.com

2.8.4. Google Maps

Google Maps merupakan sebuah layanan peta dunia virtual berbasis web yang disediakan oleh Google. Layanan ini gratis dan dapat ditemukan di http://maps.google.com.

Google Maps menawarkan peta yang dapat digeser (*panned*), diperbesar (*zoom in*), diperkecil (*zoom out*), dapat diganti dalam beberapa mode (*ma*p, *satelit*, *hybrid*, dan lain-lain), fitur pencarian rute (*routing*), penunjuk arah dari satu objek peta ke objek yang lain (*direction*) dan juga pencarian tempat (*place*). Sampai saat ini, Google Maps masih berada dalam tahap beta, dan masih terus dikembangkan dengan data yang selalu diperbarui secara berkala.

Google maps merupakan hak cipta Google secara propriety, sehingga dalam menggunakannya memerlukan adanya perjanjian, Google membuat mekanisme untuk dapat mengakses Google Maps dengan coding aplikasi dengan sebuah kunci yang dikenal dengan API *Key*.

2.8.5. MySQL

MySQL adalah salah satu jenis database *server* yang sangat terkenal, kepopulerannya disebabkan MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya. Selain itu , ia bersifat *Open source* (Anda tidak perlu membayar untuk menggunakannya) pada pelbagai *platform* (kecuali untuk jenis enterprise, yang bersifat komersial). Perangkat lunak MySQL sendiri bisa di *download* dari http://www.mysql.com

MySQL termasuk jenis RDBMS (*Relational Database Management*). Itulah sebabnya, istilah seperti tabel, baris, dan kolom digunakan pada MySQL. Pada MySQL, sebuah database mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom.

2.8.6. PHP

Menurut dokumen resmi PHP, PHP merupakan singkatan dari *PHP Hypertext Preprocessor*. Ia merupakan bahasa berbentuk skrip yang ditempatkan dalam *server* dan diproses di *server*. Hasilnya yang dikirim ke klien tempat pemakai menggunakan *browser*.

Secara khusus, PHP dirancang untuk membentuk aplikasi web dinamis. Artinya, ia dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya, Anda bisa menampilkan isi *database* ke halaman web. Pada prinsipnya PHP mempunyai fungsi yang sama dengan skrip-skrip seperti ASP (*Active Server Page*), Cold Fusion, ataupun Perl. Namun, perlu diketahui bahwa PHP sebenernya bisa dipakai secara *command line*. Artinya, skrip PHP dapat dijalankan tanpa melibatkan *web server* maupun *browser*.

Kelahiran PHP bermula saat Rasmus Lerdorf membuat sejumlah skrip Perl yang dapat mengamati siapa saja yang melihat-lihat daftar riwayat hidupnya, yakni pada tahun 1994. Skrip-skrip ini selanjutnya dikemas menjadi *tool* yang disebut "*Personal home page*". Paket inilah yang menjadi cikal-bakal PHP. Pada tahun 1995,

Rasmus menciptakan PHP/FI Versi 2. Pada versi inilah pemrograman dapat menempelkan kode terstruktur di dalam tag HTML. Yang menarik, kode PHP juga bisa berkomunikasi dengan database dan melakukan perhitungan-perhitungan kompleks sambil jalan.

Pada saat ini PHP cukup popular sebagai peranti pemrograman Web, terutama di lingkungan Linux. Walaupun demikian, PHP sebenarnya juga dapat berfungsi pada *server-server* yang berbasis UNIX, Windows, dan Machintosh.

Pada awalnya, PHP dirancang untuk diintegrasikan dengan web *server Apache*. Namun, belakangan PHP juga dapat bekerja dengan web *server* seperti PWS (*Personal Web Server*), IIS (*Internet Information Server*), dan Xitami.

Untuk mencoba PHP, anda tidak perlu menggunakan komputer bekelas *server*. Hanya dengan sebuah Komputer biasa, anda bisa mempelajari dan mempraktikkan PHP.

Pemrograman berorientasi objek merupakan model pemrograman yang menjadi tren saat ini. PHP juga mendukung hal ini.

Suatu objek dapat dibentuk melalui kelas. Dalam hal ini, kelas dapat dianggap sebagai cetakan objek. Jika menggunakan istilah teknis, kelas adalah definisi umum untuk sekelompok objek. Objek sendiri sering disebut instan kelas. Dengan menggunakan cetakan, anda bisa menciptakan objek.