BAB 2
LANDASAN TEORI

2.1. Al-Quran

Al-Quran berasal dari bahasa Arab yang berarti "bacaan" atau "sesuatu yang dibaca berulang-ulang". Umat islam percaya bahwa Al-Qur'an merupakan puncak dan penutup wahyu Allah yang diperuntukkan bagi manusia, dan bagian dari rukun iman, yang disampaikan kepada Nabi Muhammad *Shallallahu 'alaihi wa sallam*, melalui perantaraan Malaikat Jibril.

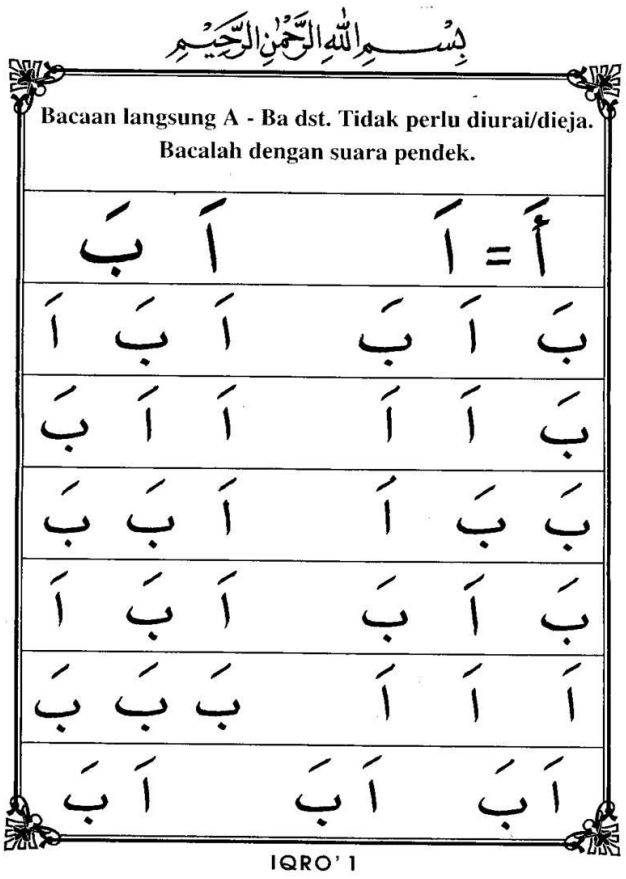
Maka setiap muslim (beragama islam) diwajibkan bisa untuk membaca Al-Quran, yang di dalamnya terkandung bagaimana menjalani hidup di dunia dan mendapatkan surga (selamat dunia akhirat). Di indonesia sendiri terdapat beberapa metode pembelajaran Al-Quran, salah satunya adalah "Metode *iqra* cara cepat belajar membaca Al-Quran".

2.1.1. Metode *Iqra*

Pembelajaran iqra atau yang sering kita sebut dengan "metode *iqra* cara cepat belajar membaca Al-Quran", metode ini di kembangkan oleh Tim Tadarrus Angkatan Muda Masjid (AMM) Yogyakarta bersama KH. As'ad Humam sebagai pencetus ide. Mereka mencari bentuk baru bagi sistem pengelolaan pengajian anak-anak dan metode pembelajaran membaca Al-Quran.

Setelah melalui studi banding dan uji coba, maka pada tanggal 16 Maret 1988 didirikanlah Taman Kanak-Kanak Al-Quran (TKA) "AMM" Yogyakarta. Setahun kemudian, tepatnya tanggal 23 April 1989 didirikan pula Taman Pendidikan Al-Quran (TPA). Antara TKA dan TPA tidaklah memiliki perbedaan dalam sistem, keduanya hanya berbeda dalam hal usia anak didiknya.

Bersamaan dengan didirikannya TKA-TPA, KH. As'ad Humam tekun menulis dan menyusun buku *iqra, Cara Cepat Belajar Membaca Al-Quran*, yang kemudian lebih di kenal dengan " Metode *Iqra* ". Metode ini ternyata sanggup membawa anak-anak lebih mudah dan cepat dalam belajar membaca Al-Quran.



Gambar 2.1 Metode Iqra Jilid 1

Sebagai materi pokoknya adalah pengembanagn membaca Al Qur'an dengan menggunakan buku *Iqra* julid 1 – 6 susunan ustadz As'ad Humam . Bila seorang telah mampu membaca Al Qur'an dengan benar, maka sebagai kelanjutannya ia mulai tadarus Al Qur'an (mulai juz 1 dan bulan berikutnya dimulai dari juz 'Amma).

Adapun pokok-pokok bahasan membaca Al Qur'an dalam materi pokok itu, menurut masing-masing jilid ialah seperti di bawah ini :

1. Iqro' jilid 1
   a. Membunyikan huruf tunggal berharakat *fathah*, mulai dari huruf *alif* sampai *ya'*.
   b. Pelajaran khusus *makhraj-makhraj* yang sukar.
   c. Pengenalan huruf-huruf *hijaiyah* sekedarnya saja

2. Iqro' jilid 2
    a. Mengenal dan membaca huruf sambung.
    b. Mengenal dan membaca huruf dengan *mad* (panjang).
3. Iqro' jilid 3
    a. Mengenal dan membaca huruh bertanda *kasroh*.
    b. Mengenal *mad* untuk *kasroh*.
    c. Mengenal berbagai bentuk huruf *ha*.
    d. Mengenal dan membaca huruf *berharokat dhommah*.
    e. Mengenal dan membaca mad untuk *dhommah*.
4. Iqro' jilid 4
    a. Mengenal bacaan tanwin *fathah*.
    b. Mengenal bacaan tanwin *kasroh*.
    c. Mengenal bacaan tanwin *dhommah*.
    d. Mengenal dan membaca *mad bayyinah*, seperti bacaan *baina*.
    e. Mengenal bacaan *izhar*.
    f. Mengenal bacaan *qolqolah*.
5. Iqro' jilid 5
    a. Pengenalan bacaan tanpa *alif*, misalnya bacaan *al hamdu*.
    b. Cara membaca bila ada *waqof*.
    c. Pengenalan mad khusus, misalkan bacaan "*ulaaika*", dibaca panjang 5 *harakat*. Huruf *wawu* di belakang *alif* dianggap tidak ada.
    d. Cara membaca huruf *nun sukun* bertemu dengan huruf *mim/ ba*. Cara membacanya adalah masuk pada mim dengan dengung.
    e. Pengenalan bacaan *al / alif lam syamsiyah*, misalnya *binnudzur*.
    f. Pengenalan bacaan *tasydid* misalnya *yukabbiru*, cara membacanya adalah ditekan dan ditahan dua harokat.
    g. Pengenalan bacaan *ikhfa' syafawi*, mim mati bertemu dengan ba'.
    h. Pengenalan *lafdhul jalalah*.
    i. Pengenalan nun mati atau bacaan *tanwin* bertemu dengan huruf *ra'* seperti, disini cara membacanya huruf *nun* dihilangkan.

j. Pengenalan mad khusus yang *bertsydid*, seperti bacaan *waladdolliin*, cara membacanya yaitu panjang 6 *harokat* baru dengan *tasydid*.

6. Iqro' jilid 6
   a. Pengenalan bacaan nun mati atau tanwin bertemu dengan huruf wau.
   b. Pengenalan nun mati atau tanwin bertemu dengan huruf ya'.
   c. Pengenalan nun mati atau tanwin bertemu dengan huruf ba.
   d. Pengenalan nun mati atau tanwin bertemu dengan huruf fa. Cara membacanya adalah dengan samar-samar dan dengung.
   e. Pengenalan cara membaca bila *waqof*, yaitu *waqof bertasydid*, *waqof qolqolah*, *waqof* pada huruf *hamzah* dan *waqof mu'anaqoh*.
   f. Pengenalan bacaan *muqotho'at* atau membuka pada surat, seperti pada surat Al Baqoroh, An Nisa dan sebagainya.

2.2. Multimedia

Multimedia adalah gabungan video, audio, grafik dan teks dalam suatu produksi bertingkat berbasis komputer yang dapat dialami secara interaktif atau menurut McCormick multimedia secara umum merupakan kombinasi tiga elemen yaitu suara, gambar dan teks atau menurut Robin dan Linda multimedia merupakan alat yang dapat menciptakan presentasi yang dinamis dan interaktif yang mengkombinasikan teks, grafik, animasi, audio dan gambar video.

Sedangkan menurut wikipedia.org : "*Multimedia is the use of several different media to convey information (text, audio, graphics, animation, video, and interactivity). Multimedia also refers to computer data storage devices, especially those used to store multimedia content.".* Orang yang merangkai atau mengkombinasikan elemen multimedia (teks, suara, animasi, grafik serta video) menjadi suatu produk multimedia yang dapat memenuhi tujuan tertentu dikenal dengan istilah pengembang multimedia (*multimedia developers).*

Ketika pemakai (*user*) suatu produk multimedia dapat mengontrol, apa yang harus ditampilkan serta kapan, sesuai dengan kebutuhan *user*, maka multimedia tersebut dikenal dengan istilah multimedia interaktif (*interactive multimedia).*

Meskipun definisi dari multimedia tersebut sederhana, tetapi untuk membuat multimedia sampai berfungsi bukanlah suatu pekerjaan yang mudah.

Tidak cukup hanya mengerti bagaimana membuat tiap-tiap elemen multimedia, tetapi kita harus mengetahui cara penggunaan setiap jenis *tools* multimedia serta teknologinya yang kita pilih, sehingga dapat menghasilkan produk multimedia yang baik mutunya. Dengan pertolongan sebuah tools yang disebut dengan authoring tools, elemen-elemen multimedia dapat diintegrasikan menjadi produk multimedia. *Tools* software ini didesain untuk memanajemen (*to manage)* tiap-tiap elemen multimedia serta dilengkapi dengan fasilitas untuk menghasilkan dan mengedit teks dan citra, dan dapat dihubungkan ke driver dari *videodisc player*, *videotape player* serta hardware pheriperi (*hardware pheriperals)* lainnya.

2.2.1. Sejarah Multimedia

Istilah multimedia berawal dari teater, bukan komputer. Pertunjukan yang memanfaatkan lebih dari satu medium sering kali disebut pertunjukan multimedia. Pertunjukan multimedia mencakup monitor video, *synthesized band*, dan karya seni manusia sebagai bagian dari pertunjukan. Sistem Multimedia dimulai pada akhir 1980-an dengan diperkenal¬kannya Hypercard oleh Apple pada tahun 1987, dan pengumuman oleh IBM pada tahun 1989 mengenai perangkat lunak Audio Visual Connec¬tion (AVC) dan video adapter card bagi PS/2. Sejak permulaan tersebut, hampir setiap pemasok perangkat keras dan lunak melom¬pat ke multimedia. Pada 1994, diperkirakan ada lebih dari 700 produk dan sistem multimedia di pasaran.

Citra visual dapat dimasukkan ke dalam sistem dari paket perangkat lunak yang menyatukan digital, dan dari kamera video, pita dan piringan video, dan scanner optik. Input audio dapat dimasukkan melalui mikrofon, pita kaset, dan compact disk. Output visual dapat ditampilkan di layar komputer dan di monitor televisi yang tersambung. Output audio dapat disediakan oleh alat output suara, speaker stereo, dan headset. Pada 1990, harga sistem multimedia yang lengkap berkisar $10.000, tapi harganya sejak itu menurun, membuat teknologi itu dalam

jangkauan perusahaan kecil yang benar-benar membutuhkan. Namun, harga perangkat keras tidak mencerminkan total biaya untuk Multimedia dalam makalah buku ini digunakan untuk menjelaskan suatu sistem yang terdiri dari perangkat keras, perangkat lunak dan alat-alat lain seperti televisi, monitor video dan sistem piringan optik atau sistem stereo-semua dimaksudkan untuk menghasilkan penyajian audiovisual penuh. Multimedia memungkinkan pemakai komputer untuk mendapatkan output dalam bentuk yang jauh lebih kaya daripada media tabel dan grafik konvensional. Pemakai dapat melihat gambar tiga dimensi, foto, video bergerak, atau animasi, dan mendengar suara stereo, perekaman suara,atau musik.

2.2.2. Definisi Multimedia

Dalam industri elektronika Multimedia adalah kombinasi dari komputer dan video atau multimedia secara umum merupakan kombinasi tiga elemen, yaitu suara, gambar dan teks atau multimedia adalah kombinasi dari paling sedikit dua media input atau output dari data, media ini dapat audio (suara, musik), animasi, video, teks, grafik dan gambar atau multimedia merupakan alat yang dapat menciptakan presentasi yang dinamis dan interaktif yang mengkombinasikan teks, grafik, animasi, audio dan gambar. Definisi yang lain dari multimedia, yaitu dengan menempatkan dalam konteks, multimedia adalah pemanfaatan komputer untuk membuat dan menggabungkan teks, grafik, audio, gambar bergerak (video dan animasi) dengan menggabungkan link dan tool yang memungkinkan pemakai melakukan navigasi, berinteraksi, berkreasi dan berkomunikasi.

2.2.3. *Multimedia Personal Computer* (MPC)

Dengan kemajuan teknologi informasi dalam beberapa tahun terakhir ini, semua komputer yang ada dipasaran mempunyai kemampuan untuk memproses (*playback, recording, editing*) citra (*image*), suara dan video selain teks, menjadi suatu elemen multimedia. Komputer yang mempunyai kemampuan tersebut dikenal dengan istilah Komputer multimedia *(Multimedia computer, MPC)*. Kemampuan ini tercipta disebabkan oleh hardware dan software multimedia yang terpasang pada PC multimedia.

Hardware multimedia antara lain meliputi sound card, video card, cd-player, mix serta box suara. Sound card, video card dan cd player terpasang dan terintegrasi pada motherboards yang merupakan bagian terpenting dari computer. Mix beserta loud speaker terpasang pada sound card sehingga dapat merekam serta dapat mengeluarkan suara dari sutau produk multimedia. PC Multimedia terbagi menjadi dua jenis MPC level 1(*MPC-1)* dan MPC level 2 (*MPC-2).*

Software multimedia terdiri dari software driver dan software aplikasi. Dengan pertolongan software driver, komputer (CPU) dapat mengontrol komponen hardware multimedia, jadi driver itu merupakan interface antara komponen hardware multimedia dengan komputer itu sendiri. Tiap pembelian komponen hardware multimedia, selalu dilengkapi software install untuk memasang driver tersebut pada komputer. Software aplikasi multimedia yang digunakan untuk menghasilkan elmen-elemen multimedia, tersedia dipasaran baik yang dibuat oleh perusahaan software terkenal seperti Microsoft atau perusahaan software lainnya.

2.2.4. Penggunaan Multimedia

Setelah diperkenalkan PC multimedia serta harganya yang dapat terjangkau oleh masyarakat, multimedia dapat diterapkan pada bidang-bidang kehidupan, seperti bisnis, sekolah, rumah, pariwisata, hotel dan lain-lain.

Aplikasi multimedia dibidang bisnis meliputi presentasi, training, pemasaran, dan demonstrasi suatu produk.

Aplikasi dibidang pemasaran mengakibatkan, salesman dapat memberi informasi secara baik dan lengkap tentang produk yang dipasarkan kepada konsumen, karena semua informasi dan pesan yang ingin disampaikan, tersajikan secara singkat dan jelas melalui tayangan gambar, animasi gerak, suara dan sebagainya. Melalui multimedia, konsumen dengan santai dapat mempelajari lebih detil mengenai suatu produk sebelum membeli.

Berkat multimedia untuk training, seorang operator dapat belajar dengan mudah bagaimana mengoperasikan mesin yang baru dibeli, seorang mekanik pada

memperbaiki mesin yang rusak sesuai dengan instruksi yang terdapat pada software multimedia.

Pemakaian multimedia disekolah dapat membantu seorang guru dalam menerapkan suatu mata pelajaran yang sulit untuk dicerna oleh anak-anak didiknya, seperti dalam mata pelajaran IPA. Melalui animasi gerak dan visualisasi video, misalnya hukum-hukum newton dapat diterangkan dengan baik, serta dipahami dengan mudah oleh murid-muridnya. Dengan demikian multimedia dapat meningkatkan qualitas proses belajar-mengajar disekolah.

Dengan pertolongan multimedia seorang tamu yang hendak menginap dihotel, dapat melihat terlebih dulu keadaan kamar, wc beserta perlengkapannya, sebelum memutuskan memilih suatu kamar tertentu. Dengan demikian qualitas service hotel terhadap tamunya dapat diperbaiki. Multimedia dapat membantu dinas kepariwisataan untuk mengiklankan potensi pariwisata daerahnya kepada turis local ataupun mancanegara, seperti kesenian tradisional daerah tersebut, tempat-tempat pariwisata, dan lain-lain.

2.2.5. Elemen-Elemen Multimedia

Dalam multimedia terdapat beberapa elemen pendukung, diantaranya sebagai berikut:

1. Teks

   Hampir semua orang yang biasa menggunakan komputer sudah terbiasa dengan teks. Teks merupakan dasar dari pengolahan kata dan informasi berbasis multimedia. Dalam kenyataannya multimedia menyajikan informasi kepada audiens dengan cepat, karena tidak diperlukan membaca secara rinci dan teliti. Menurut Hofstetter (dalam adalah kebanyakan sistem multimedia dirancang dengan menggunakan teks karena teks merupakan sarana yang efektif untuk mengemukakan ide-ide dan menyediakan instruksi-instruksi kepada user (pengguna).

2. Image

   Secara umum image atau grafik berarti still image seperti foto dan gambar. Manusia sangat berorientasi pada visual dan gambar merupakan sarana yang sangat baik untuk menyajikan informasi.

3. Animasi

   Animasi adalah pembentukan gerakan dari berbagai media atau objek yang divariasikan dengan gerakan transisi, efek-efek, juga suara yang selaras dengan gerakan animasi tersebut atau animasi merupakan penayangan frame-frame gambar secara cepat untuk menghasilkan kesan gerakan.

4. Audio

   Penyajian audio atau suara merupakan cara lain untuk lebih memperjelas pengertian suatu informasi. Contohnya, narasi merupakan kelengkapan dari penjelasan yang dilihat melalui video. Suara dapat lebih menjelaskan karakteristik suatu gambar, misalnya musik dan suara efek (sound effect). Salah satu bentuk bunyi yang bisa digunakan dalam produksi multimedia adalah Waveform Audio yang merupakan format file audio yang berbentuk digital. Kualitas produknya bergantung pada sampling rate (banyaknya sampel per detik). Waveform (wav) merupakan standar untuk Windows PC.

2.3. Kecerdasan Buatan

Kecerdasan Buatan *(Artificial Intelegent)* merupakan cabang dari ilmu komputer yang dalam merepresentasi pengetahuan lebih banyak menggunakan bentuk simbol-simbol daripada bilangan, dan memproses informasi berdasarkan metode heuristic atau dengan berdasarkan sejumlah aturan.

2.3.1. Kecerdasan Buatan Dalam Berbagai Perspektif

Dari perspektif Kecerdasan (Intelligence) AI adalah bagaimana membuat mesin yang “cerdas” dan dapat melakukan hal-hal yang sebelumnya dapat dilakukan oleh manusia. Dari perspektif bisnis, AI adalah sekelompok alat bantu (*tools*) yang berdaya guna, dan metodologi yang menggunakan *tool-tool* tersebut guna menyelesaikan masalah-masalah bisnis. Dari perspektif pemrograman

(*Programming*), AI termasuk didalamnya adalah studi tentang pemrograman simbolik, pemecahan masalah, proses pencarian (*search*).

2.3.2. Domain Penelitian Dalam Kecerdasan Buatan

Adapun domain-domain dari Penelitian dalam kecerdasan buatan diantaranya adalah :

1. Permainan *(Game)*

   Kebanyakan permainan dilakukan dengan menggunakan sekumpulan aturan. Dalam permainan digunakan apa yang disebut dengan pencarian ruang. Teknik untuk menentukan alternatif dalam menyimak problema ruang merupakan sesuatu yang rumit. Teknik tersebut disebut dengan *heuristic*. Permainan merupakan bidang yang menarik dalam studi *heuristic*.

2. *Natural Language*

   Suatu teknologi yang memberikan kemampuan kepada komputer untuk memahami bahasa manusia sehingga pengguna komputer dapat berkomunikasi dengan komputer dengan menggunakan bahasa sehari -hari.

3. Robotik

   Robotik dan sistem sensor Sistem sensor, seperti sistem vision, sistem *tactile*, dan sistem pemrosesan sinyal jika dikombinasikan dengan AI, dapat dikategorikan kedalam suatu sistem yang luas yang disebut sistem robotik.

4. Sistem Pakar (*Expert System*)

   Sistem pakar (*Expert System*) adalah program penasehat berbasis computer yang mencoba meniru proses berpikir dan pengetahuan dari seorang pakar dalam menyelesaikan masalah-masalah spesifik.

2.3.3. Konsep Dan Definisi Dalam Kecerdasan Buatan

Turing Test merupakan sebuah metode pengujian kecerdasan yang dibuat oleh Alan Turing. Proses uji ini melibatkan seorang penanya (manusia) dan dua obyek yang ditanyai. Yang satu adalah seorang manusia dan satunya adalah sebuah mesin yang akan diuji. Penanya tidak bisa melihat langsung kepada obyek yg ditanyai Penanya diminta untuk membedakan mana jawaban komputer dan mana jawaban manusia berdasarkan jawaban kedua obyek tersebut. Jika penanya

tidak dapat membedakan mana jawaban mesin dan mana jawaban manusia maka Turing berpendapat bahwa mesin yang diuji tersebut dapat diasumsikan cerdas.

Dalam pemrosesan simbolik Komputer semula didisain untuk memproses bilangan/angka-angka (pemrosesan numerik). Sementara manusia dalam berpikir dan menyelesaikan masalah lebih bersifat simbolik, tidak didasarkan kepada sejumlah rumus atau melakukan komputasi matematis. Sifat penting dari AI adalah bahwa AI merupakan bagian dari ilmu komputer yang melukan proses secara *simbolik* dan *non-algoritmik* dalam penyelesaian masalah.

Istilah *Heuristic* diambil dari bahasa Yunani yang berarti menemukan Heuristic merupakan suatu strategi untuk melakukan proses pencarian (*search*) ruang problema secara selektif, yang memandu proses pencarian yang kita lakukan disepanjang jalur yang memiliki kemungkinan sukses paling besar.

Penarikan Kesimpulan (Inferencing) dimana AI mecoba membuat mesin memiliki kemampuan berpikir atau mempertimbangkan (*reasoning*) Kemampuan berpikir (*reasoning*) termasuk didalamnya proses penarikan kesimpulan (*inferencing*) berdasarkan fakta-fakta dan aturan dengan menggunakan metode heuristik atau metode pencarian lainnya.

Pencocokan Pola (*Pattern Matching*) AI bekerja dengan metode pencocokan pola (*pattern matching*) yang berusaha untuk menjelaskan obyek, kejadian (*events*) atau proses, dalam hubungan logik atau komputasional.

2.3.4. Hubungan AI Dengan Pengenalan Suara

Pemahaman Ucapan/Suara (*Speech/Voice Understanding*), adalah teknik agar komputer dapat mengenali dan memahami bahasa ucapan. Proses ini mengijinkan seseorang berkomunikasi dengan komputer dengan cara berbicara kepadanya. Istilah “pengenalan suara” mengandung arti bahwa tujuan utamanya adalah mengenai kata yang diucapkan tanpa harus tahu artinya, di mana bagian itu merupakan tugas “pemahaman suara”. Secara umum prosesnya adalah usaha untuk menerjemahkan apa yang diucapkan seorang manusia menjadi kata-kata atau kalimat yang dapat dimengerti oleh komputer.

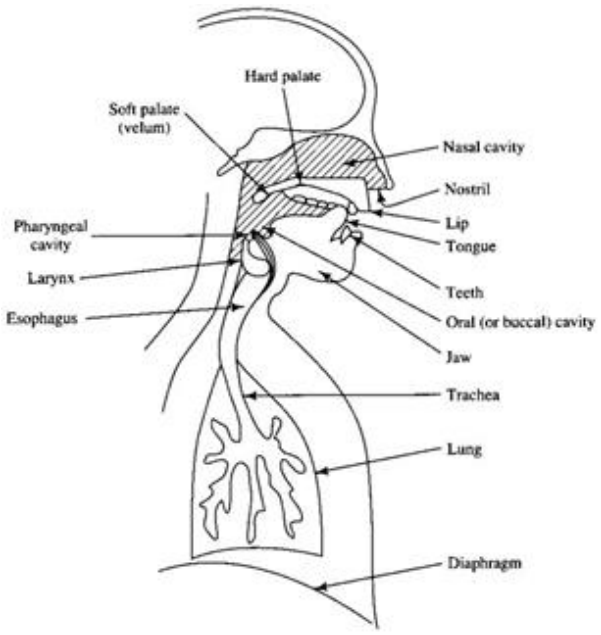
2.4. Pengenalan Suara Pada Manusia

Pada sistem pengenalan suara oleh manusia terdapat tiga organ penting yang saling berhubungan yaitu : telinga yang berperan sebagai transduser dengan menerima sinyal masukan suara dan mengubahnya menjadi sinyal syaraf, jaringan syaraf yang berfungsi mentransmisikan sinyal ke otak, dan otak yang akan mengklasifikasi dan mengidentifikasi informasi yang terkandung dalam sinyal masukan.

2.4.1. Proses Produksi Suara

Proses produksi suara pada manusia dapat dibagi menjadi tiga buah proses fisiologis, yaitu : pembentukan aliran udara dari paru-paru, perubahan aliran udara dari paru-paru menjadi suara, baik *voiced*, maupun *unvoiced* yang dikenal dengan istilah *phonation*, dan artikulasi yaitu proses modulasi/ pengaturan suara menjadi bunyi yang spesifik.

Organ tubuh yang terlibat pada proses produksi suara adalah : paru-paru, tenggorokan (*trachea*), laring (*larynx*), faring (*pharynx*), pita suara (*vocal cord*), rongga mulut (*oral cavity*), rongga hidung (*nasal cavity*), lidah (*tongue*), dan bibir (*lips*), seperti dapat dilihat pada gambar dibawah.



Gambar 2.2 Organ tubuh manusia

Organ tubuh ini dapat dikelompokkan menjadi tiga bagian utama, yaitu : *vocal tract* (berawal di awal bukaan pita suara atau glottis, dan berakhir di bibir), *nasal tract* (dari velum sampai nostril), dan *source generator* (terdiri dari paru-paru, tenggorokan, dan *larynx*). Ukuran *vocal tract* bervariasi untuk setiap individu, namun untuk laki-laki dewasa rata-rata panjangnya sekitar 17 cm. Luas dari vocal tract juga bervariasi antara 0 (ketika seluruhnya tertutup) hingga sekitar 20 cm^2. Ketika *velum*, organ yang memiliki fungsi sebagai pintu penghubung antara *vocal tract* dengan *nasal tract*, terbuka, maka secara akustik *nasal tract* akan bergandengan dengan vocal tract untuk menghasilkan suara *nasal*.

Aliran udara yang dihasilkan dorongan otot paru-paru bersifat konstan. Ketika pita suara dalam keadaan berkontraksi, aliran udara yang lewat membuatnya bergetar. Aliran udara tersebut dipotong-potong oleh gerakan pita suara menjadi sinyal pulsa yang bersifat *quasi-periodik*. Sinyal pulsa tersebut kemudian mengalami modulasi frekuensi ketika melewati *pharynx*, rongga mulut ataupun pada rongga hidung. Sinyal suara yang dihasilkan pada proses ini dinamakan sinyal *voiced*. Namun, apabila pita suara dalam keadaan relaksasi, maka aliran udara akan berusaha melewati celah sempit pada permulaan vocal tract sehingga alirannya menjadi turbulen, proses ini akan menghasilkan sinyal *unvoiced*. Ketika sumber suara melalui *vocal tract*, kandungan frekuensinya mengalami modulasi sehingga terjadi resonansi pada *vocal tract* yang disebut *formants*. Apabila sinyal suara yang dihasilkan adalah sinyal *voiced*, terutama vokal, maka pada selang waktu yang singkat bentuk *vocal tract* relative konstan (berubah secara lambat) sehingga bentuk *vocal tract* dapat diperkirakan dari bentuk spektral sinyal *voiced*.

Aliran udara yang melewati pita suara dapat dibedakan menjadi *phonation*, bisikan, *frication*, kompresi, vibrasi ataupun kombinasi diantaranya. *Phonated excitation* terjadi bila aliran udara dimodulasi oleh pita suara. *Whispered excitation* dihasilkan oleh aliran udara yang bergerak cepat masuk ke dalam lorong bukaan segitiga kecil antara *arytenoids cartilage* di belakang pita suara yang hampir tertutup. *Frication excitation* dihasilkan oleh desakan di vocal tract. *Compression excitation* dihasilkan akibat pelepasan udara melalui vocal

tract yang tertutup dengan tekanan tinggi. *Vibration excitation* disebabkan oleh udara yang dipaksa memasuki rusang selain pita suara, khususnya lidah. Suara yang dihasilkan oleh *Phonated excitation* disebut *voiced.* Suara yang dihasilkan oleh *Phonated excitation* ditambah *frication* disebut *mixed voiced,* sedangkan yang dihasilkan oleh selain itu disebut *unvoiced*. Karakteristik suara tiap individu bersifat unik karena terdapat perbedaan dalam hal panjang maupun bentuk vocal tract.

2.4.2. Karakteristik Telinga

Telinga terbagi menjadi tiga bagian, yaitu bagian luar, tengah, dan dalam. Pinna, sebagai bagian luar telinga, berfungsi sebagai corong, untuk mengumpulkan sinyal suara menuju *auditory canal* sehingga dapat memberikan kesan arah sinyal suara yang diterima.

Auditory canal adalah struktur berbentuk pipa lurus sepanjang 2,7 cm, dengan diameter sekitar 0,7 cm, yang pada bagian ujungnya terdapat selaput membrane, yaitu gendang telinga. Membran ini merupakan pintu masuk telinga bagian tengah, yaitu ruangan berisi udara dengan volume sebesar 2 cm^3, yang terdiri dari tiga buah tulang, yaitu *malleus* (martil), *incus* (landasan), dan *stapes* (sanggurdi). Bagian ini terhubung dengan tenggorokan melalui *Eustachian tube*. Getaran pada gendang telinga ditransmisikan ke *malleus* melalui *incus*, dan *stapes*, yaitu membentuk *oval window*.

Telinga bagian dalam (*labyrinth*) memiliki tiga bagian, yaitu *vestibule* (ruang pintu masuk), *semicular canal*, dan *cochlea*. *Vestibule* terhubung dengan telinga bagian tengah melalui dua jalur, yaitu *oval window*, dan *round window*. Keduanya tertutup untuk mencegah keluarnya cairan yang mengisi telinga telinga bagian dalam. Pada *cochlea*, yang berstruktur seperti rumah siput, terdapat syaraf pendengaran. Syaraf ini memanjang sampai ke *basilar membrane*. Pada bagian atas *basilar membrane* terdapat *organ of corty* yang memiliki empat baris sel rambut (sekitar 3×10^4 sel seluruhnya).

2.4.3. Proses Pendengaran

Proses pendengaran pada telinga manusia dijelaskan sebagai berikut :

1. Sinyal suara memasuki saluran telinga dan variasi tekanan yang dihasilkannya menekan gendang telinga. Karena sisi bagian dalam dari gendang telinga mempunyai tekanan yang nilainya dijaga konstan maka gendang telinga akan bergetar.
2. Getaran dari gendang telinga disalurkan pada tiga rangkaian tulang yaitu; *martil*, *incus* dan *stapes*. Mekanisme ini dirancang untuk mengkopel variasi suara dari udara luar ke telinga bagian dalam. Karena luas permukaan penampang yang ditekan *stapes* lebih kecil dari luas penampang gendang telinga maka tekanan suara yang sampai ke telinga bagaian dalam bertambah besar.
3. Cairan pada *cochlea* bergetar dengan frekuensi yang sama dengan gelombang yang datang. *Basilar membrane* kemudian memisahkan sinyal berdasarkan frekuensinya. *Basilar membrane* berstruktur kuat dan panjang di daerah sekitar *oval window* namun bersifat lentur pada bagian ujungnya. Frekuensi resonansi yang dihasilkan membrane tersebut berbeda sepanjang dimensi *basilar membrane*. Dimana resonansi frekuensi tinggi terjadi pada bagian bagian *basilar membrane* yang berada dekat dengan *oval window*, sedangkan resonansi frekuensi rendah terjadi pada daerah ujung lainnya. Syaraf yang berada pada mambran kemudian mendeteksi posisi terjadinya resonansi yang juga akan menentukan frekuensi suara yang datang. Ukuran dari *basilar membrane* rata-rata sekitar 35 mm. Dari ukuran panjang tersebut dapat dihasilkan 10 resolusi frekuensi, sehingga pada setiap 3.5 mm panjang membran terdapat 1 oktaf frekuensi resonansi.

2.4.4. Translator Bahasa Alami ke Bahasa Buatan

Translator bahasa alami ke bahasa buatan, yaitu translator yang mengubah perintah-perintah dalam bahasa alami menjadi bahasa buatan yang dapat dieksekusi oleh mesin atau komputer. Sebagai contoh, translator yang memungkinkan kita memberikan perintah bahasa alami kepada komputer. Dengan sistem seperti ini, pengguna sistem dapat memberikan perintah dengan bahasa sehari-hari, misalnya, untuk menghapus semua file, pengguna cukup memberikan

perintah" komputer, tolong hapus semua file !" Translator akan mentranslasikan perintah bahasa alami tersebut menjadi perintah bahasa formal yang dipahami oleh komputer, yaitu " dir *.* < ENTER>".

2.5. Google API

Google API bisa di katakan bagian dari *Framework* Google. Google menyediakan berbagai API (*Application Programming Interface*) yang sangat berguna bagi pengembang web maupun aplikasi desktop untuk memanfaatkan berbagai fitur yang disediakan oleh Google seperti misalnya: AdSense, *Search Engine*, *Translation* maupun YouTube.

API secara sederhana bisa diartikan sebagai kode program yang merupakan antarmuka atau penghubung antara aplikasi atau web yang kita buat dengan fungsi-fungsi yang dikerjakan. Misalnya dalam hal ini Google API berarti kode program yang dapat kita tambahkan pada aplikasi atau web kita untuk mengakses/menjalankan/memanfaatkan fungsi atau fitur yang disediakan Google. Misalnya saja kita bisa menambahkan fitur Google Map pada website kita.

Google API dapat dipelajari langsung melalui Google *Code*. Melalui Google *Code* kita dapat belajar tentang Google API dan dapat mengimplementasikan pada aplikasi atau website yang kita kembangkan.

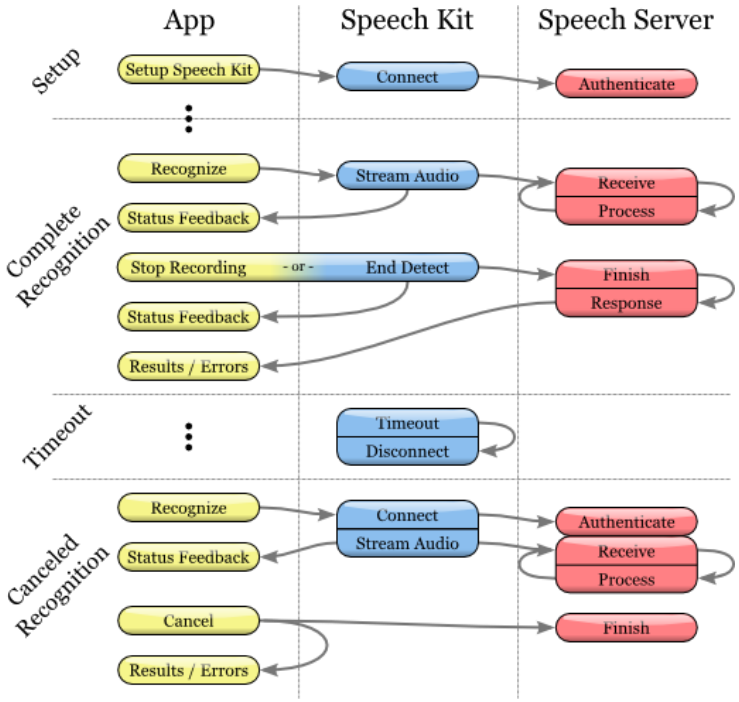
Ada banyak API yang disediakan oleh Google, beberapa diantaranya adalah:

1. Language API: untuk memanfaatkan fitur translation yang dimiliki Google.
2. Earth API: memanfatkan fitur yang ada pada Google Earth
3. Javascript API
4. Maps API: memanfaatkan fitur yang ada pada Google Maps
5. Search API: memanfaatkan fitur pencarian pada Google Search
6. Visualization API: membuat grafik maupun chart dengan Google API
7. YouTube API: memanfaatkan fitur yang ada pada YouTube misalnya untuk pencarian video

Sekarang kita dapat mendikte pesan hanya tekan tombol mikrofon pada layar, dan kita dapat berbicara hampir di mana saja tanpa harus bersusah payah

mengetik. Aplikasi ini di sebut dengan google *voice* API yang disediakan oleh google untuk pengembang yang ingin menciptakan aplikasi yang menarik menggunakan *speech input* API atau google *voice search*.

Google *Voice Search* atau Pencarian dengan suara adalah produk Google yang memungkinkan seseorang untuk menggunakan Google *Search* dengan berbicara pada ponsel atau komputer, pencarian perangkat untuk data saat memasuki informasi mengenai apa yang harus di cari ke dalam perangkat dengan berbicara. Di kutip dari developer android bahwa penggunaan *speech input* API pada Android sangat bagus digunakan untuk edukasi dan lainnya.



Gambar 2.3 Speech Recognition Process

Saat ini *speech input* API atau google *voice search* sudah bisa menggunakan berbagai bahasa yang telah google sediakan di antaranya :

1. Afrikaans since 2010
2. Basque since 2012
3. Bulgarian since 2012
4. Catalan since 2012

5. Arabic (7 dialects) since 2011
6. Czech since 2010
7. Dutch since 2010
8. English (Australia, Canada, INDIA, New Zealand, South Africa, UK, US) since the 2008
9. Finnish since 2012
10. French since 2010
11. Galician since 2012
12. German since 2010
13. Hebrew since 2011
14. Hungarian since 2012
15. Icelandic since 2012
16. Italian since 2010
17. Indonesian since 2011
18. Japanese since 2009
19. Korean since 2010
20. Latin
21. Mandarin Chinese (Traditional Taiwan, Simplified China, Simplified Hong Kong) since 2009
22. Malaysian since 2011
23. Norwegian since 2012
24. Polish since 2010
25. Pig Latin since 1 April 2011
26. Portuguese (Brazilian; European since 2012)
27. Romanian since 2012
28. Russian since 2010
29. Serbian since 2012
30. Slovak since 2012
31. Spanish since 2010 and Latin American Spanish since 2011
32. Swedish since 2012
33. Turkish since 2010

34. Yue Chinese (Traditional Hong Kong) since 2010

2.5.1. Google API *speech input*

Untuk bisa menggunakan fitur google api *speech input* yang telah disediakan google dalam *developer* android kita dapat menggunakan interface dan class yang disediakan google.

2.5.1.1. Interfaces

RecognitionListener Digunakan untuk menerima pemberitahuan dari *SpeechRecognizer* ketika peristiwa yang terkait terjadi. Di dalam *RecognitionListener* mempunyai publich methods sebagai berikut :

Tabel 2.1 *Public Method Recognitionlistener*

Abstract void	onBeginningOfSpeech() pengguna mulai berbicara.
Abstract void	onBufferReceived(byte[] buffer) suara telah di terima. Tujuan dari fungsi ini adalah untuk memungkinkan memberi umpan balik kepada pengguna mengenai audio yang ditangkap *buffer* : buffer yang mengandung urutan big-endian integer 16-bit yang mewakili audio streaming saluran tunggal. Sample rate adalah implementasi tergantung.
Abstract void	onEndOfSpeech() pengguna telah berhenti bicara.
Abstract void	onError(int error) terjadi kesalahan jaringan. *error* : kode kesalahan didefinisikan dalam SpeechRecognizer.
Abstract void	onEvent(int eventType, Bundle params)

	menambahkan kegiatan untuk *event* yang terjadi. *eventType* : jenis peristiwa yang terjadi. *params* : sebuah bundle parameter yang dikirimkan
Abstract void	onPartialResults(Bundle partialResults) Ketika hasil pengakuan parsial tersedia. Callback bisa dipanggil antara onBeginningOfSpeech () dan onResults (Bundle) ketika hasil parsial siap. Metode ini dapat disebut nol, satu atau beberapa kali untuk setiap panggilan ke startListening (Intent), tergantung pada pelaksanaan layanan pengenalan suara. Untuk meminta hasil parsial, gunakan EXTRA_PARTIAL_RESULTS *partialResults* : kembali pada hasil. Untuk mengambil hasil di ArrayList Format penggunaan getStringArrayList (String) dengan RESULTS_RECOGNITION sebagai parameter
Abstract void	onReadyForSpeech(Bundle params) Disebut ketika endpointer sudah siap bagi pengguna untuk mulai berbicara. *params* : parameter yang ditetapkan oleh layanan pengenalan
Abstract void	onResults(Bundle results) dipanggil ketika pencocokan siap. *results* : Untuk mengambil hasil di ArrayList<string> Format penggunaan getStringArrayList (String) dengan RESULTS_RECOGNITION sebagai parameter.

2.5.2. Classes

Ada beberapa class yang terdapat dalam google api *speech input*, class ini dapat dipanggil langsung karena telah terdapat dalam google api, diantaranya adalah:

1. RecognitionService : Kelas ini menyediakan kelas dasar untuk implementasi layanan pengenalan.
2. RecognitionService.Callback : Kelas ini menerima callback dari layanan pengenalan suara dan mengirimkannya ke pengguna.
3. RecognizerIntent : Konstanta untuk mendukung pengenalan suara melalui mulai Intent
4. RecognizerResultsIntent : Konstanta untuk maksud yang berkaitan dengan menunjukkan hasil pengenalan suara.
5. SpeechRecognizer : Kelas ini menyediakan akses ke layanan pengenalan suara.

Salah Satu fitur android yang telah tersedia yaitu google api *Speech Input* yang didukung oleh *server* google dimana membutuhkan koneksi *internet* agar dapat merespon RecognizerIntent. Untuk mengimplementasikan Speech Input, aplikasi memverifikasi bahwa perangkat target mampu mengenali *speech input*:

```
//cek fitur voice recognition
PackageManager pm = getPackageManager();
List activities = pm.queryIntentActivities(new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH), 0);
if (activities.size() != 0){
speakButton.setOnClickListener(this); } else {
speakButton.setEnabled(false);
```

```
speakButton.setText("Pengenalan suara tidak aktif"); }
```

Aplikasi kemudian menggunakan `startActivityForResult()` untuk memberitahukan bahwa sedang meminta *voice recognition*, serta sebuah parameter tambahan yang menentukan salah satu dari dua model bahasa. Aplikasi *voice recognition* yang menangani hasil dari proses *voice input*, kemudian melewati pengenalan *string* kembali ke aplikasi dengan memanggil `onActivityResult()`.

```
//memulai voice recognition

private void startVoiceRecognitionActivity() {

Intent intent = new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,Recog
nizerIntent.LANGUAGE_MODEL_FREE_FORM);

intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Speech
recognition demo");

startActivityForResult(intent,VOICE_RECOGNITION_REQUEST_COD
E);}
//menangani hasil balik

protected void onActivityResult(int requestCode, int
resultCode, Intent data) {

if (requestCode == VOICE_RECOGNITION_REQUEST_CODE &&
resultCode == RESULT_OK) {

//memasukkan hasil kedalam daftar

ArrayList<String> matches = data.getStringArrayListExtra(
  RecognizerIntent.EXTRA_RESULTS);

    mList.setAdapter(new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,matches));
} super.onActivityResult(requestCode, resultCode, data);
}
```

2.6. Pemodelan Sistem

Dalam suatu proses pengembangan software, analisa dan rancangan telah merupakan terminologi yang sangat tua. Pada saat masalah ditelusuri dan spesifikasi dinegosiasikan, dapat dikatakan bahwa kita berada pada tahap rancangan. merancang adalah menemukan suatu cara untuk menyelesaikan masalah, salah satu tool/model untuk merancang pengembangan software yang berbasis object-oriented adalah UML. Alasan mengapa UML digunakan adalah, pertama, scalability dimana objek lebih mudah dipakai untuk menggambarkan sistem yang besar dan komplek. Kedua, dynamic modeling, dapat dipakai untuk pemodelan sistem dinamis dan real time. Sebagaimana dalam tulisan pertama, penulis menjelaskan konsep mengenai obyek, OOA&D (Obyek Oriented Analyst/ Design) dan pengenalan UML, maka dalam tulisan kedua ini lebih ditekankan pada cara bagaimana UML digunakan dalam merancang sebuah pengembangan software yang disertai gambar atau contoh dari sebuah aplikasi.

2.6.1. Basis Data

Basis data (*database*), atau sering pula dieja basisdata, adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*) basis data disebut sistem manajemen basis data (*database management system*, DBMS). Sistem basis data dipelajari dalam ilmu informasi.

Istilah "basis data" berawal dari ilmu komputer. Meskipun kemudian artinya semakin luas, memasukkan hal-hal di luar bidang elektronika, artikel ini mengenai basis data komputer. Catatan yang mirip dengan basis data sebenarnya sudah ada sebelum revolusi industri yaitu dalam bentuk buku besar, kuitansi dan kumpulan data yang berhubungan dengan bisnis.

Konsep dasar dari basis data adalah kumpulan dari catatan-catatan, atau potongan dari pengetahuan. Sebuah basis data memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya: penjelasan ini disebut skema. Skema

menggambarkan obyek yang diwakili suatu basis data, dan hubungan di antara obyek tersebut. Ada banyak cara untuk mengorganisasi skema, atau memodelkan struktur basis data: ini dikenal sebagai model basis data atau model data. Model yang umum digunakan sekarang adalah model relasional, yang menurut istilah layman mewakili semua informasi dalam bentuk tabel-tabel yang saling berhubungan dimana setiap tabel terdiri dari baris dan kolom (definisi yang sebenarnya menggunakan terminologi matematika). Dalam model ini, hubungan antar tabel diwakili denga menggunakan nilai yang sama antar tabel. Model yang lain seperti model hierarkis dan model jaringan menggunakan cara yang lebih eksplisit untuk mewakili hubungan antar tabel.

Istilah *basis data* mengacu pada koleksi dari data-data yang saling berhubungan, dan perangkat lunaknya seharusnya mengacu sebagai *sistem manajemen basis data* (*database management system/DBMS*). Jika konteksnya sudah jelas, banyak administrator dan programer menggunakan istilah basis data untuk kedua arti tersebut.

Perangkat lunak basis data yang banyak digunakan dalam pemrograman dan merupakan perangkat basis data aras tinggi (*high level*):

Tabel 2.2 Perangkat Lunak Basis Data

1.	DB2	13.	FoxPro
2.	Microsoft SQL Server	14.	Visual FoxPro
3.	Oracle	15.	Arago
4.	Sybase	16.	Force
5.	Interbase	17.	Recital
6.	XBase	18.	dbFast
7.	Firebird	19.	dbXL
8.	MySQL	20.	Quicksilver
9.	PostgreSQL	21.	Clipper
10.	Microsoft Access	22.	FlagShip
11.	dBase III	23.	Harbour
12.	Paradox	24.	Visual dBase

Selain perangkat lunak pada tabel 2.1, terdapat juga perangkat lunak pemrograman basis data aras rendah (*low level*), diantaranya:

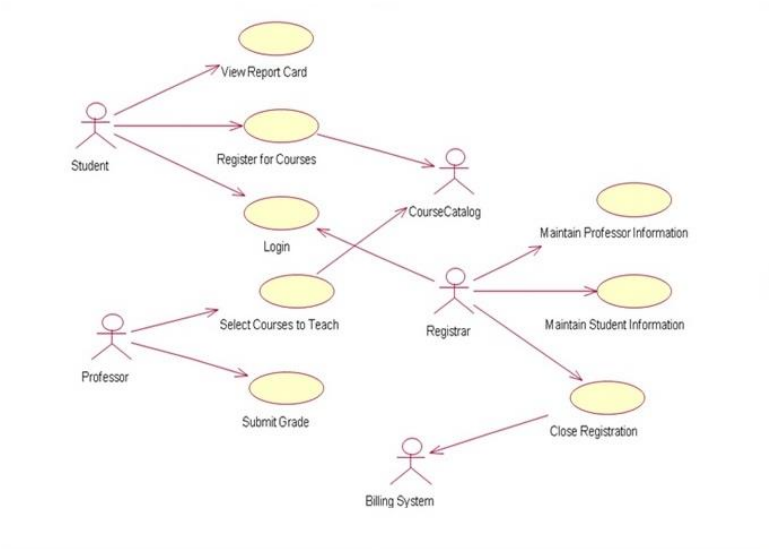
1. Btrieve
2. Tsunami Record Manager

DBMS (Database Management System) adalah software yang menangani semua akses ke basis data. Secara konsep apa yang terjadi adalah sebagai berikut :

1. user melakukan pengaksesan basis data untuk informasi yang diperlukannya menggunakan suatu bahasa manipulasi data, biasanya disebut SQL.
2. DBMS menerima *request* dari user & menganalisa request tersebut
3. DBMS memeriksa skema eksternal user, pemetaan eksternal/konseptual, skema konseptual, pemetaan konseptual/internal, & struktur penyimpanan.
4. DBMS mengeksekusi operasi-operasi yang diperlukan untuk memenuhi permintaan user.

2.6.2. *Use Case Diagram*

Sebuah *use case* menggambarkan suatu urutan interaksi antara satu atau lebih aktor dan sistem. Dalam fase *requirements*, model *use case* mengambarkan sistem sebagai sebuah kotak hitam dan interaksi antara aktor dan sistem dalam suatu bentuk naratif, yang terdiri dari input user dan respon-respon sistem. Setiap use case menggambarkan perilaku sejumlah aspek sistem, tanpa mengurangi struktur internalnya. Selama pembuatan model use case secara pararel juga harus ditetapkan obyek-obyek yang terlibat dalam setiap use case.



Gambar 2.4 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem.

2.6.2.1. Aktor

Sebuah aktor mencirikan suatu bagian *outside user* atau susunan yang berkaitan dengan user yang berinteraksi dengan sistem [Rumbaugh, Booch, da Jacobson 1999]. Dalam model use case, aktor merupakan satu-satunya kesatuan eksternal yang berinteraksi dengan sistem. Terdapat beberapa variasi bagaimana aktor dibentuk [Fowler dan Scott 1999]. Sebuah aktor sering kali merupakan manusia (*human user*). Pada sejumlah sistem informasi, manusia adalah satu-satunya aktor. Dan mungkin saja dalam sistem informasi, seorang aktor bisa saja menjadi suatu sistem eksternal. Pada aplikasi real-time dan distribusi, sebuah aktor bisa saja menjadi satu perangkat eksternal I/O atau sebuah alat pengatur waktu. Perangkat eksternal I/O dan pengatur waktu aktor secara khusus lazimnya berada dalam real-time yang tersimpan dalam sistem (*real-time embedded systems*), sistem berinteraksi dengan lingkungan eksternal melalui sensor dan aktuator.

Primary actor (aktor utama) memprakarsai sebuah use case. Jadi, suatu primary aktor memegang peran sebagai proaktif dan yang memulai aksi dalam sistem. Aktor lainnya yang berperan sebagai *secondary* aktor bisa saja terlibat dalam use case dengan menerima *output* dan memberikan *input*. Setidaknya satu actor harus mendapatkan nilai dari use case. Biasanya adalah *primary* aktor (actor utama). Bagaimanapun, dalam *real-time embedded systems*, *primary* aktor dapat berperan sebagai perangkat eksternal I/O atau pengatur waktu, penerima utama dari *use case* bisa menjadi *secondary human* aktor yang menerima sejumlah informasi dari sistem.



Gambar 2.5 Aktifitas Actor Dan Use Case

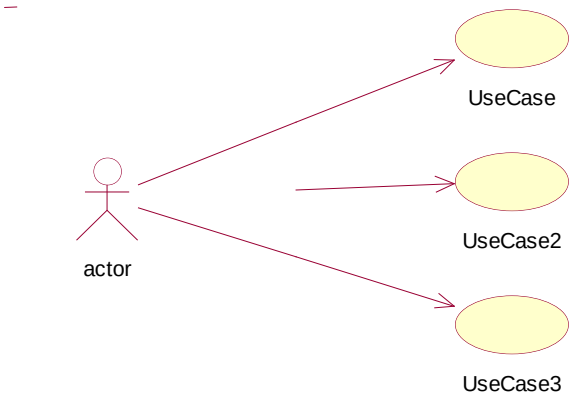
2.6.2.2. Identifikasi *Use Case*

Sebuah use case dimulai dengan masukan/input dari seorang aktor. Use case merupakan suatu urutan lengkap kejadian-kejadian yang diajukan oleh seorang aktor, dan spesifikasi interaksi antara aktor dengan sistem. Use case yang sederhana hanya melibatkan satu interaksi/hubungan dengan sebuah aktor, dan use case yang lebih kompleks melibatkan beberapa interaksi dengan aktor. Use cases yang lebih kompleks juga melibatkan lebih dari satu aktor.

Untuk menjabarkan use case dalam sistem, sangat baik bila dimulai dengan memperhatikan aktor dan actions/aksi yang mereka lakukan dalam sistem. Setiap use case menggambarkan suatu urutan interaksi antara aktor dengan sistem. Sebuah use case harus memberikan sejumlah nilai pada satu aktor.

Kemudian, kebutuhan fungsional sistem dijelaskan dalam use case yang merupakan suatu spesifikasi eksternal dari sebuah sistem. Bagaimanapun juga, ketika membuat use case, sangatlah penting menghindari suatu dekomposisi fungsional yang dalam beberapa use case kecil lebih menjelaskan fungsi-fungsi

9/10 individual sistem daripada menjelaskan urutan kejadian yang memberikan hasil yang berguna bagi aktor.



Gambar 2.6 interaksi actor dan use case

Urutan utama use case menjelaskan urutan interaksi yang paling umum antara aktor dan sistem. Dan mungkin saja terdapat cabang-cabang urutan use case utama, yang mengarah pada berkurangnya frekuensi interaksi antara actor dengan sistem. Deviasi-deviasi dari urutan utama hanya dilaksanakan pada beberapa situasi, contohnya jika aktor melakukan kesalahan input pada sistem. Ketergantungan pada aplikasi kebutuhan, alternatif ini memecahkan use case dan kadang-kadang bersatu kembali dengan urutan utama. Cabang-cabang alternative digambarkan juga dalam use case.

2.6.2.3. Pendokumentasian Model *Use Case*

Use case didokumentasi dalam *use case model* sebagai berikut:

1. *Use Case Name.*

   Setiap use case diberi nama.

2. *Summary.*

   Deskripsi singkat use case, biasanya satu atau dua kalimat.

3. *Dependency.*

   Bagian ini menggambarkan apakah use case yang satu tergantung pada use case yang lain, dalam arti apakah use case tersebut termasuk pada use case yang lain atau malah memperluas use case lain.

4. *Actors*

   Bagian ini memberikan nama pada actor dalam use case. Selalu terdapat use case utama (primary use case) yang memulai use case. Disamping itu terdapat juga secondary use case yang terlibat dalam use case. Contohnya, dalam use case Withdraw Funds, ATM Customer adalah actor-nya.

5. *Preconditions*

   Satu atau lebih kondisi harus berjalan dengan baik pada permulaan use case; contohnya mesin ATM yang tidak jalan, menampilkan pesan Selamat Datang.

6. *Deskripsi*

   Bagian terbesar dari use case merupakan deskripsi naratif dari urutan utama use case yang merupakan urutan yang paling umum dari interaksi antara aktor dan sistem. Deskripsi tersebut dalam bentuk input dari aktor, diikuti oleh respon pada sistem. Sistem ditandai dengan sebuah kotak hitam (black box) yang berkaitan dengan apa yang sistem lakukan dalam merespon input aktor, bukan bagaimana internal melakukannya.

7. *Alternatif-alternatif*

   Deskripsi naratif dari alternatif merupakan cabang dari urutan utama. Terdapat beberapa cabang alternatif dari urutan utama. Contohnya, jika rekening customer terdapat dana yang tidak sesuai, akan tampil permohonan maaf dan menolak kartu.

8. *Postcondition*

   Kondisi yang selalu terjadi di akhir use case, jika urutan utama telah dilakukan; contohnya dana customer telah ditarik.

9. *Outstanding questions*

   Pertanyaan-pertanyaan tentang use case didokumentasikan untuk didiskusikan dengan para user.

2.6.3. *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).

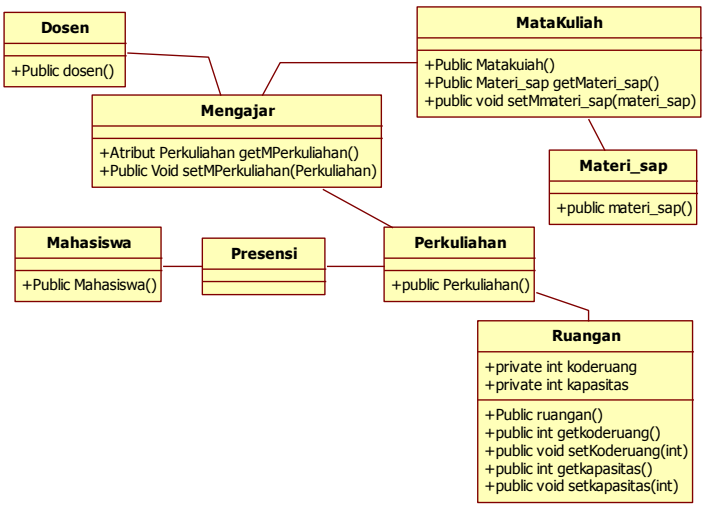
Class diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

1. Nama(danstereotype)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

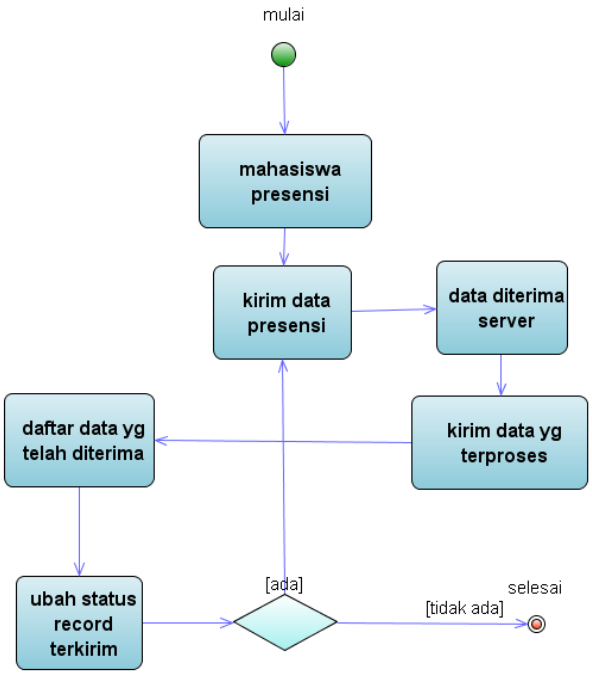
1. Private, tidak dapat dipanggil dari luar class yang bersangkutan
2. Protected, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya
3. Public, dapat dipanggil oleh siapa saja



Gambar 2.7 Class Diagram

2.6.4. *StateChart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke *state* lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya statechart diagram menggambarkan class tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).



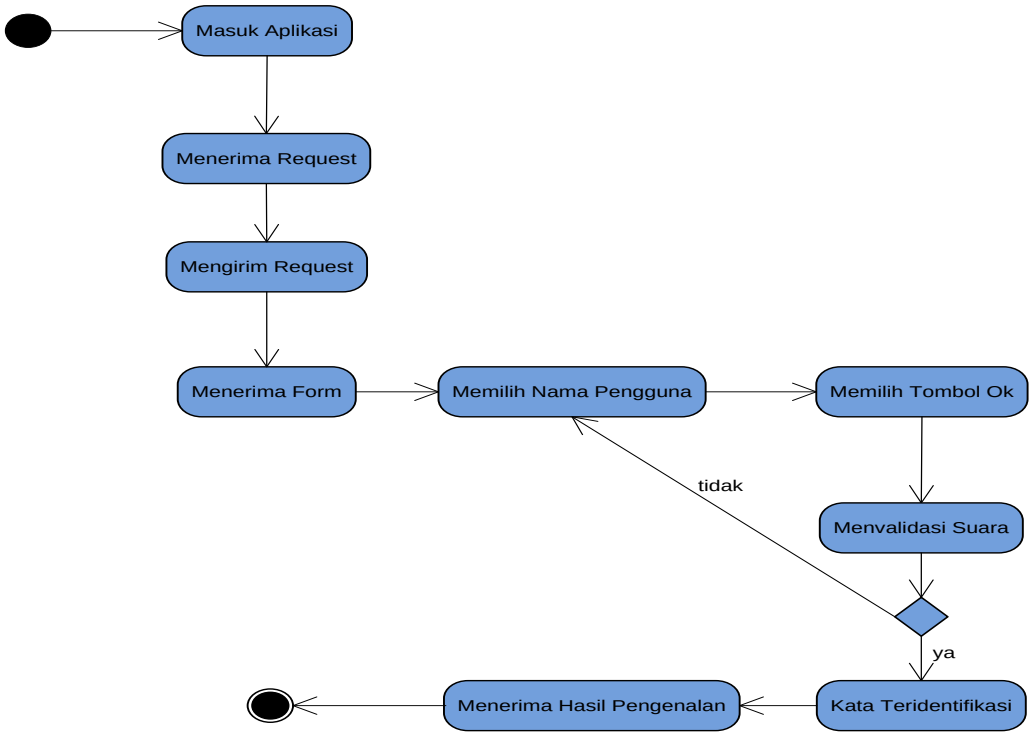
Gambar 2.8 StateChart Diagram

2.6.5. *Activivty Diagram*

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem)

secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

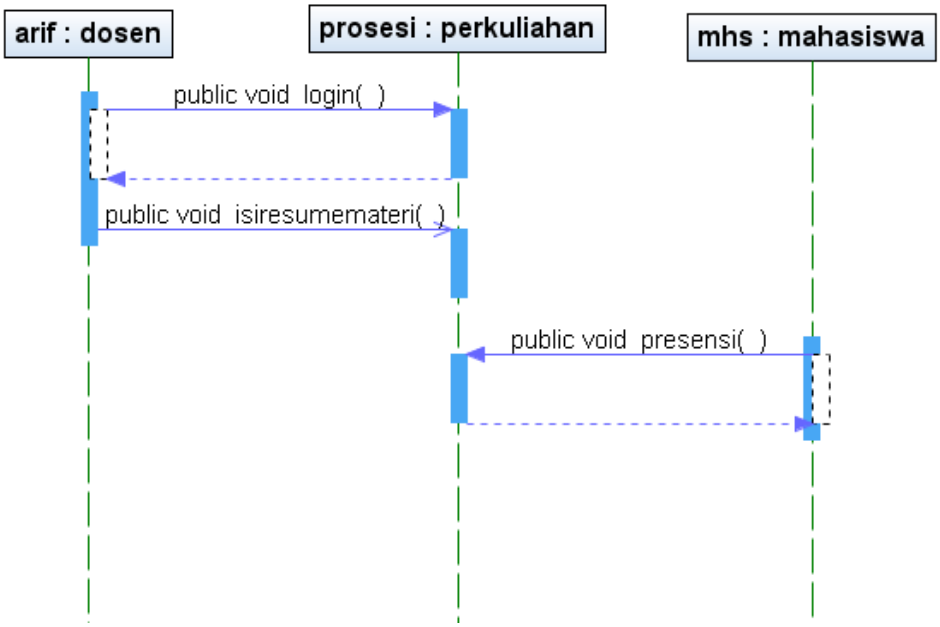


Gambar 2.9 Activity Diagram

2.6.6. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. *Sequence* diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.



Gambar 2.10 Sequence Diagram

2.7. Metode Pengembangan Perangkat Lunak

Merupakan sebuah model *chaos* yang menggambarkan "perkembangan P/L sebagai sebuah kesatuan dari pemakai ke pengembang dan ke teknologi" disebut dengan "Prescriptive" karena Menentukan sekumpulan elemen proses (Aktivitas, Aksi, tugas, produk kerja, jaminan kualitas, dll untuk setiap proyek), setiap model proses juga menentukan alur kerjanya.

Pada saat kerja bergerak maju menuju sebuah sistem yang lengkap, keadaan yang digambarkan secara rekursif diaplikasikan kepada kebutuhan pemakai dan spesifikasi teknis P/L pengembang Saat ini, prescriptive memberikan jawaban secara definitive untuk masalah pengembangan P/L dalam setiap perubahan lingkungan komputasi

Pada Tugas akhir ini menggunakan metode penggunaan perangkat lunak siklus kehidupan klasik atau model air terjun. Model ini mengusulkan sebuah pendekatan kepada perkembangan software yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode,

pengujian, dan pemeliharaan. Dimodelkan setelah siklus rekayasa konvensional, model sekuensial linier melingkupi aktivitas – aktivitas sebagai berikut :

1. Rekayasa dan pemodelan sistem/informasi

   Karena sistem merupakan bagian dari sebuah sistem yang lebih besar, kerja dimulai dengan membangun syarat dari semua elemen sistem dan mengalokasikan beberapa subset dari kebutuhan ke software tersebut. Pandangan sistem ini penting ketika software harus berhubungan dengan elemen-elemen yang lain seperti software, manusia, dan database. Rekayasa dan anasisis system menyangkut pengumpulan kebutuhan pada tingkat sistem dengan sejumlah kecil analisis serta disain tingkat puncak. Rekayasa informasi mancakup juga pengumpulan kebutuhan pada tingkat bisnis strategis dan tingkat area bisnis.

2. Analisis kebutuhan Software

   Proses pengumpulan kebutuhan diintensifkan dan difokuskan, khusunya pada software. Untuk memahami sifat program yang dibangun, analis harus memahami domain informasi, tingkah laku, unjuk kerja, dan interface yang diperlukan. Kebutuhan baik untuk sistem maupun software didokumentasikan dan dilihat lagi dengan pelanggan.

3. Desain

   Desain software sebenarnya adalah proses multi langkah yang berfokus pada empat atribut sebuah program yang berbeda; struktur data, arsitektur software, representasi interface, dan detail (algoritma) prosedural. Proses desain menterjemahkan syarat/kebutuhan ke dalam sebuah representasi software yang dapat diperkirakan demi kualitas sebelum dimulai pemunculan kode. Sebagaimana persyaratan, desain didokumentasikan dan menjadi bagian dari konfigurasi software.

4. Generasi Kode

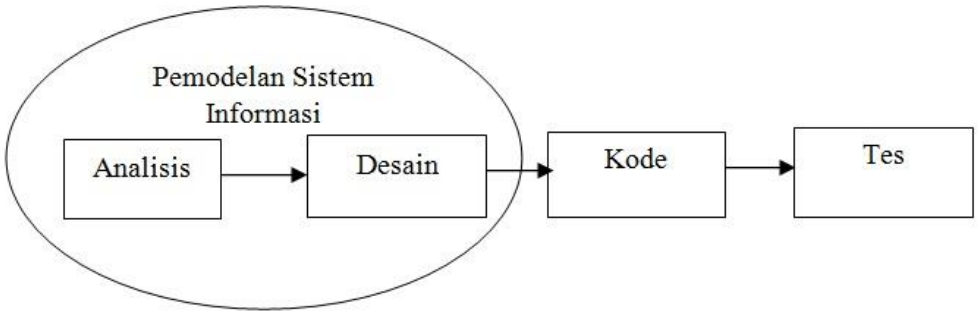
   Desain harus diterjemahkan kedalam bentuk mesin yang bias dibaca. Langkah pembuatan kode melakukan tugas ini. Jika desain dilakukan dengan cara yang lengkap, pembuatan kode dapat diselesaikan secara mekanis.

5. Pengujian

   Sekali program dibuat, pengujian program dimulai. Proses pengujian berfokus pada logika internal software, memastikan bahwa semua pernyataan sudah diuji, dan pada eksternal fungsional, yaitu mengarahkan pengujian untuk menemukan kesalahan – kesalahan dan memastikan bahwa input yang dibatasi akan memberikan hasil aktual yang sesuai dengan hasil yang dibutuhkan.

6. Pemeliharaan

   Software akan mengalami perubahan setelah disampaikan kepada pelanggan (perkecualian yang mungkin adalah software yang dilekatkan). Perubahan akan terjadi karena kesalahan – kesalahan ditentukan, karena software harus disesuaikan untuk mengakomodasi perubahan – perubahan di dalam lingkungan eksternalnya (contohnya perubahan yang dibutuhkan sebagai akibat dari perangkat peripheral atau sistem operasi yang baru), atau karena pelanggan membutuhkan perkembangan fungsional atau unjuk kerja. Pemeliharaan software mengaplikasikan lagi setiap fase program sebelumnya dan tidak membuat yang baru lagi.



Gambar 2.11 Model Skuensial Linear (model metodologi Waterfall)

Masalah yang kadang terjadi ketika model sekuensial linier diaplikasikan adalah :

1. Jarang sekali proyek nyata mengikuti aliran sekuensial yang dianjurkan oleh model. Meskipun model linier bisa mengakomodasi iterasi, model ini melakukannya dengan cara tidak langsung. Sebagai hasilnya, perubahan – perubahan dapat menyebabkan keraguan pada saat tim proyek berjalan.

2. Kadang – kadang sulit bagi pelanggan untuk menyatakan semua kebutuhannya secara eksplisit. Model linier sekuensial memerlukan hal ini dan mengalami kesulitan untuk mengakomodasi ketidakpastian natural yang ada pada bagian awal beberapa proyek. Pemodelan Sistem Informasi
3. Pelanggan harus bersifat sabar. Sebuah versi kerja dari program – program kerja itu tidak akan diperoleh sampai akhir waktu proyek dilalui. Sebuah kesalahan besar, jika tidak terdeteksi sampai program yang bekerja tersebut dikaji ulang, bisa menjadi petaka.
4. Pengembang sering melakukan penundan yang tidak perlu. Sifat alami dari siklus kehidupan klasik membawa kepada *blocking state* di mana banyak anggota tim proyek harus menunggu tim yang lain untuk melengkapi tugas yang saling memiliki ketergantungan. Blocking state cenderung menjadi lebih lazim pada awal dan akhir sebuah proses sekuensial linier.

2.8. Android

Android adalah sebuah sistem operasi untuk pernagkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Awalnya, Gogle Inc. Membeli Android Inc. Yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone.*

Sistem operasi yang mendasari Android dilisensikan di bawah GNU, General Public Lisensi Versi 2 (GPLV2), yang sering di kenal dengan istilah "*copyleft*" lisensi di mana setiap perbaikan pihak ketiga harus jatuh terus dibawah *terms*. Android di distribusikan dibawah lisensi Apache *Software* (ASL/Apache2), yang memungkinkan untuk distribusi kedua dan seterusnya.

2.8.1. Android SDK (*Software Development Kit)*

Android SDK adalah *tolls* API (*Aplikation Programing Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemograman Java.

Untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Handphone*/*Smartphone*. Beberapa fitur Android yang paling penting adalah :

1. *Framework* aplikasi yang mendukung penggantian komponen dan *reusable.*
2. Mesin *Virrtual Dalvik* dioptimalkan untuk perangkat *mobile*.
3. *Integrated browser* berdasarkan *engine open source Webkit.*
4. Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0.
5. SQLite untuk penyimpanan Data.
6. *Media Support* yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF) GSM *Telephone.*
7. *Bluetooth,* EDGE, 3G, HSDPA dan WiFi (tergantung hardware)
8. Kamera, GPS, kompas dan *accelerometer*.
9. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging,* profil dan kinerja memori, dan *plugin* untuk IDE Eclipse.

2.8.2. Arsitektur Android

Secara garis besar asitektur android dapat dijelaskan sebagai berikut :

1. *Aplication* dan *Widgets.*

Aplication dan *Widgets* ini adalah layer di mana kita berhubungan dengan aplikasi saja, di mana kita biasanya mendownload aplikasi kemudian kita lakukan instalasi dan menjalankan aplikasi tersebut.

2. *Aplications Frameworks.*

Android merupakan "*Open Development Platform*" yaitu menawarkan kepada pengembang atau memberi kamampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Jadi, *Aplications Frameworks* ini adalah *layer* di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi android.

Komponen-komponen yang termasuk di dalam *Aplications Frameworks* adalah sebagai berikut :

a. *Views*
b. *Content Provider*

c. *Resource Manager*
d. *Notification Manager*
e. *Activity Manager*

3. *Libraries.*

Libraries ini adalaha layer di mana fitur-fitur android berada, biasanya pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas kernel, *Layer* ini meliputi *library* C/C++ inti seperti Libc dan SSL, serta :

a. *Libraries* media untuk pemutaran media audio dan video.
b. *Libraries* untuk manajemen tampilan.
c. *Libraries Grapichs* mencakup SGL dan OpenGL untuk grafis 2D dan 3D.
d. *Libraries* SQLite untuk dukungan *database.*
e. *Libraries LiveWebcore* mencakup *modern web browser* dengan *engine embeded web view.*
f. *Libraries* 3D yang mencakup implementasi OpenGL ES 1.0 API's

4. *Android Run Time.*

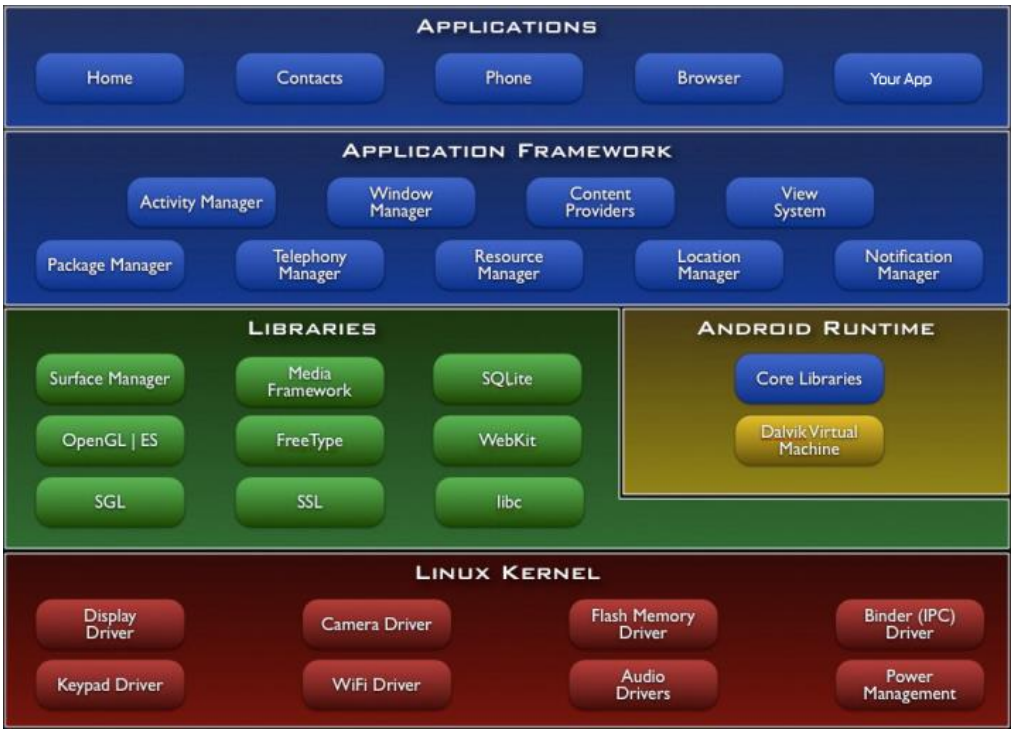
Layer yang membuat aplikasi android dapat dijalankan di mana dalam prosesnya menggunakan implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi android. Dalam *Android Run Times* terbagi menjadi dua bagian yaitu :

a. *Core Libraries* : aplikasi android dibangun dalam bahasa java, sementara *Dalvik* yang merupakan *virtual machine* bukan *virtual machine Java,* sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa java/c yang ditangani oleh *Core Libraries.*
b. *Dalvik Virtual Machine* : Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, di mana merupakan pengembangan yang mampu membuat linux kernel untuk melakukan *threading* dan manajemen tingkat rendah.

5. Linux *Kernel.*

Linux *Kernel* merupakan *layer* di mana inti dari sistem opersi android berada. Berisi file-file sistem yang mengatur sistem *processing*, *memory, resource,*

drivers, dan sistem-sistem operasi android lainnya. Linux *Kernel* yang digunakan android adalah linux *kernel release* 2.6.



Gambar 2.12 Arsitektur Android.

2.9. Eclipse

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform independent*).

Banyak orang mengenal Eclipse sebagai IDE (integrated development environment) untuk bahasa Java, tapi Eclipse lebih dari sekedar IDE untuk Java.

Komunitas Eclipse memiliki lebih dari 60 proyek open source. Proyek-proyek ini secara konsep terbagi menjadi 7 categori :

1. Enterprise Development
2. Embedded and Device Development
3. Rich Client Platform
4. Rich Internet Applications

5. Application Frameworks
6. Application Lifecycle Management (ALM)
7. Service Oriented Architecture (SOA)

Secara umum Eclipse digunakan untuk membangun software inovatif berstandar industri, dan alat bantu beserta frameworknya membantu pekerjaan menjadi lebih mudah.

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

1. *Multi-platform*: Dapat berjalan pada berbagai platform seperti, Microsoft Windows, Linux, Solaris, AIX, HP-UX, dan Mac OS X.
2. *Multi-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Phyton, Perl, PHP, dan lain sebagainya.
3. *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 yang difavoritkan karena gratis dan *open source*, yang berarti setiap orang boleh melihat kode program perangkat lunak. Selain itu, kelebihan dari Eclipse yang membuatnya popular adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponan yang dinamakan *plug-in*.

2.9.1. Sejarah Eclipse

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak IBM *Visual Age for Java* 4.0. Produk ini diluncurkan oleh IBM pada tanggal, 5 november 2001, yang menginvestasikan sebanyak US$40 juta untuk pengembangannya. Semenjak itu konsursium Eclipse *Foundation* mengambil alih untuk pengembangan lebih lanjut dan pengaturan organisasinya.

2.9.2. Arsitektur Eclipse

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah kernel, yang mengangkut 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 :

1. *Core Platform*
2. OSGi
3. SWT (*Standard Widget Toolkit*)
4. JFace
5. *Eclipse Workbench*

Standard pengoperasian 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. Jadi, Eclipse tidak saja 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.

2.9.3. Perkembangan Eclipse

Sejak tahun 2006, Eclipse Foundation mengkoordinasikan peluncuran Eclipse secara rutin dan simultan yang dikenal dengan nama *Simultaneous Release*. Setiap versi peluncuran terdiri dari Eclipse Platform dan juga sejumlah proyek yang terlibat dalam proyek Eclipse. Tujuan dari sistem ini adalah untuk

menyediakan distribusi Eclipse dengan fitur-fitur dan versi yang terstandarisasi. Hal ini juga dimaksudkan untuk mempermudah pengembangan (*deployment*) dan perawatan (*maintenance*) untuk sistem enterprise, serta untuk kenyamanan. Peluncuran simultan dijadwalkan pada bulan Juni setiap tahunnya.

Tabel 2.3 Perkembangan Versi Eclipse

Kode Peluncuran	**Tanggal Peluncuran**	**Platform**	**Nama Proyek**
Eclipse 3.0	21 Juni 2004	3.0	-
Eclipse 3.1	28 Juni 2005	3.1	-
Callisto	30 Juni 2006	3.2	Callisto projects
Europa	29 Juni 2007	3.3	Europa projects
Ganymede	25 Juni 2008	3.4	Ganymede projects
Galileo	24 Juni 2009	3.5	Galileo projects
Helios	23 Juni 2010	3.6	Indigo projects
Juno	27 Juni 2012	4.2	Juno projects