BAB II

LANDASAN TEORI

2.1 Definisi Aplikasi

Kata aplikasi berasal dari kata "*application* (aplikasi, penerapan, amaran)" penggunaan, program aplikasi adalah program siap pakai. Program yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain.

Istilah *application* atau Aplikasi ini mulai perlahan masuk ke dalam istilah Teknologi Informasi semenjak tahun 1993, yang biasanya juga disingkat dengan aplikasi. Secara historis,aplikasi adalah *softwar*e yang dikembangkan oleh sebuah perusahaan. Aplikasi adalah *sofware* yang dibeli perusahaan dari tempat pembuatnya. Industri *personal computer* tampaknya menciptakan istilah ini untuk merefleksikan medan pertempuran persaingan yang baru

2.2 Definisi Pembelajaran

Pembelajaran adalah proses interaksi peserta didik dengan pendidik dan sumber belajar pada suatu lingkungan belajar. Pembelajaran merupakan bantuan yang diberikan pendidik agar dapat terjadi proses pemerolehan ilmu dan pengetahuan, penguasaan kemahiran dan tabiat, serta pembentukan sikap dan kepercayaan pada peserta didik. Dengan kata lain, pembelajaran adalah proses untuk membantu peserta didik agar dapat belajar dengan baik. Proses pembelajaran dialami sepanjang hayat seorang manusia serta dapat berlaku di manapun dan kapanpun.

Pembelajaran mempunyai pengertian yang mirip dengan pengajaran, walaupun mempunyai konotasi yang berbeda. Dalam konteks pendidikan, guru mengajar supaya peserta didik dapat belajar dan menguasai isi pelajaran hingga mencapai sesuatu objektif yang ditentukan (aspek *kognitif*), juga dapat mempengaruhi perubahan sikap (aspek *afektif*), serta keterampilan (aspek *psikomotor*) seseorang peserta didik. Pengajaran memberi kesan hanya sebagai pekerjaan satu pihak, yaitu pekerjaan guru saja. Sedangkan pembelajaran juga menyiratkan adanya interaksi.

2.3 Bahasa Jepang

Bahasa Jepang (日本語; *romaji: Nihongo*) merupakan bahasa resmi di Jepang dan jumlah penutur 127 juta jiwa. Bahasa Jepang juga digunakan oleh sejumlah penduduk negara yang pernah ditaklukkannya seperti Korea dan Republik Tiongkok . Bahasa Jepang juga dapat didengarkan di Amerika Serikat (California dan Hawaii) dan Brasil akibat emigrasi orang Jepang ke sana. Namun keturunan mereka yang disebut *nisei* (二世, generasi kedua), tidak lagi fasih dalam bahasa tersebut.

Bahasa Jepang sendiri mempunyai tiga jenis huruf yaitu Hiragana (ひらがな), Katakana (カタカナ) dan kanji. Akan tetapi yang paling umum ditemui dan paling mudah dipelajari adalah huruf Hiragana, sedangkan Katakan biasanya digunakan untuk kata serapan asing dan Kanji adalah huruf yang lumayan sulit untuk dipelajari.

2.4 Huruf Jepang

Terdapat tiga jenis huruf bahasa Jepang yaitu hiragana, katakana dan kanji. Ketiganya berbeda-beda, baik dalam segi bentuk, fungsi maupun penggunaannya.

2.4.1 Huruf Hiragana

Hiragana atau huruf hiragana (ひらがな) adalah salah satu cara penulisan dalam bahasa Jepang dan mewakili suku kata. Hiragana mulai digunakan secara luas pada abad ke-10 Masehi. Tulisan Hiragana ini adalah tulisan yang memiliki bentuk yang sangat halus yang juga pada zaman dahulu dikenal sebagai tulisan wanita atau *onna de* (女手), karena pada zaman dahulu para wanita di Jepang sering menggunakan huruf-huruf Hiragana untuk penulisan bahasa Jepang. Huruf Hiragana terbentuk dari garis-garis dan coretan-coretan yang melengkung (*Kyokusenteki*). Huruf Hiragana yang digunakan sekarang adalah bentuk huruf yang dipilih dari Soogana yang ditetapkan berdasarkan Petunjuk Departemen Pendidikan Jepang tahun 1900.

Selain itu Hiragana memiliki kegunaan tersendiri, seperti untuk menulis akhiran kata, menulis kata keterangan, digunakan untuk situasi yang formal, menulis untuk bahan bacaan anak-anak seperti buku teks, animasi dan komik serta perkataan dimana huruf Kanjinya lama tidak digunakan atau bahkan sudah tidak diketahui.

Hiragana sendiri mempunyai 46 karakter yang mewakili 46 bunyi yang berbeda. Dalam pemakaiannya, Hiragana dipakai untuk menyatakan elemen tatabahasa seperti partikel dan akhiran kata sifat dan kata kerja yang menunjukkan tatabahasa. Sedangkan Kanji lebih menyatakan elemen yang memiliki arti seperti kata benda, kata-kata sifat dan kata kerja.

Berikut ini adalah tabel Huruf Katakana :

Tabel 2.1 Huruf Hiragana

あ a	い i	う U	え e	お o
か ka	き ki	く Ku	け ke	こ ko
さ sa	し shi	す Su	せ se	そ so
た ta	ち chi	つ Tsu	て te	と to
な na	に ni	ぬ Ne	ね ne	の no
は ha	ひ hi	ふ Fu	へ he	ほ ho
ま ma	み mi	む Mu	め me	も mo
や ya		ゆ Yu		よ yo
ら ra	り ri	る Ru	れ re	ろ ro
わ wa		ん N		を wo

2.4.2 Huruf Katakana

Katakana atau huruf Katakana (カタカナ) adalah kabalikan dari huruf Hiragana, karena Katakana ini sering digunakan para laki-laki di Jepang pada zaman dahulu untuk penulisan bahasa Jepang. Katakana diciptakan sekitar tahun 800 M Huruf-huruf Katakana tentu saja berbeda dengan huruf-huruf Hiragana,

huruf Hiragana terlihat atau memiliki bentuk yang sangat halus sedangkan huruf-huruf Katakana memiliki bentuk yang tegak dan lurus.

Selain itu Katakana juga memiliki kegunaan tersendiri, dalam ilmu fonologi, katakana bisa digunakan untuk penulisan lambang bunyi atau pengucapan. Katakana melambangkan suara-suara yang sama dengan Hiragana. Katakana juga sering digunakan untuk kata-kata bahasa asing yang sudah diserap menjadi bahasa Jepang, serta digunakan untuk menuliskan *onomatope* dan kata- kata asli dalam bahasa Jepang dan mempunyai sifat sebagai penegasan saja.

Huruf Katakana biasa dipakai untuk menulis kata serapan dari Jepang, bahasa asing, nama binatang, nama orang asing, nama tumbuhan dan kota-kota luar negeri dari Jepang. Sebagaimana dalam alfabet, huruf Katakana dan Hiragana hanya mewakili satu bunyi tanpa arti. Walaupun kalimat dalam bahasa Jepang terdiri dari Hiragana, Katakana, dan Kanji. Tetapi bisa juga hanya ditulis dalam Hiragana dan Katakana.

Dalam ilmu *Fonologi*, Katakana biasa digunakan utnuk penulisan lambang bunyi atau pengucapan. Katakana digunakan untuk menulis bahasa rahasia (*Ingo*) dan bahasa *slang* (*Zokugo*). Selain itu, huruf Katakana sering digunakan pada surat-surat atau buku-buku yang berhubungan dengan perusahaan atau pekantoran. Dengan demikian, Katakana juga bisa digunakan utnuk menuliskan kata-kata yang sebenarnya bisa dituliskan dengan Hiragana atau Kanji.

Katakana melambangkan suara-suara yang sama dengan hiragana, namun tentu saja semua hurufnya berbeda. Kalau dipikir-pikir, ini tidaklah aneh karena di bahasa Indonesia jiga terdapat dua jenis huruf yaitu huruf besar dan huruf kecil yang sebetulnya melambangkan suara yang sama.

Berikut ini adalah tabel Huruf Katakana :

Tabel 2.2 Huruf Katakana

ア A	イ i	ウ u	エ e	オ o
カ Ka	キ ki	ク ku	ケ ke	コ ko
サ Sa	シ shi	ス su	セ se	ソ so
タ Ta	チ chi	ツ tsu	テ te	ト to

ナ Na	ニ ni	ヌ ne	ネ ne	ノ no
ハ Ha	ヒ hi	フ fu	ヘ he	ホ ho
マ Ma	ミ mi	ム mu	メ me	モ mo
ヤ Ya		ユ yu		ヨ yo
ラ Ra	リ ri	ル ru	レ re	ロ ro
ワ Wa		ン n		ヲ wo

2.4.3 Huruf Kanji

Kanji adalah tulisan bergambar yang diambil dari tulisan China. Meskipun bentuk kanji China dan kanji Jepang sama, namun cara bacanya berbeda. Dalam bahasa Jepang banyak kata-kata yang ditulis menggunakan kanji. Sering kali, penulisan huruf kanjinya dicampur dengan huruf hiragana karena memang ada sejumlah huruf atau kata yang tidak dapat "dikanjikan".

Diperlukan ruang dan waktu tersendiri untuk mengulas huruf kanji secara lengkap. Namun, sebagai perkenalan dengan huruf kanji, di bawah ini terdapat beberapa kanji sederhana yang mudah untuk dipahami.

Tabel 2.3 Tabel Contoh Huruf Kanji Angka

Kanji	Pelafalan	Arti
一	Ichi	Satu
二	Ni	Dua
三	San	Tiga
四	Yon	Empat
五	Go	Lima
六	Roku	Enam
七	Nana	Tujuh
八	Hachi	Delapan
九	Kyuu	Sembilan
十	Juu	Sepuluh

2.4.4 Lambang Bunyi pada Huruf Hiragana dan Katakana

Selain huruf hiragana dasar, ada pula huruf hiragana yang dimodifikasi dengan menambahkan tanda tertentu atau menggabungkan dengan huruf tertentu sehingga menghasilkan bunyi yang berbeda. Lambang bunyi tersebut antara lain *sei-on, daku-on, handaku-on* dan *yoo-on*.

1. Sei-on

   Huruf Hiragana dan Katakana yang berjumlah 46 diatas disebut *sei-on*. Lambang bunyi *sei-on* adalah sebagai berikut:

Tabel 2.4 Huruf Hiragana Sei-on

あ A	い I	う u	え e	お o
か Ka	き Ki	く ku	け ke	こ ko
さ Sa	し Shi	す su	せ se	そ so
た Ta	ち Chi	つ tsu	て te	と to
な na	に Ni	ぬ ne	ね ne	の no
は ha	ひ Hi	ふ fu	へ he	ほ ho
ま ma	み Mi	む mu	め me	も mo
や ya		ゆ yu		よ yo
ら ra	り Ri	る ru	れ re	ろ ro
わ wa		ん n		を wo

2. Daku-on

   Bunyi *daku-on* adalah bunyi yang dihasilkan dari huruf hiragana yang diberi tambahan tanda *tenten* [“]. Lambang bunyi *daku-on* berjumlah 20 huruf. Bunyi ini hanya berasal dari set huruf konsonan vokal k, s, t dan h. Lambang bunyi *daku-on* adalah sebagai berikut:

Tabel 2.5 Huruf Hiragana *Daku-on*

が Ga	ぎ Gi	ぐ Gu	げ ge	ご go
ざ Za	じ Ji	ず Ju	ぜ ze	ぞ zo
だ Da	ぢ Ji	づ Ju	で de	ど do

ば Ba	び Bi	ぶ Bu	べ be	ぼ bo

3. Handaku-on

Bunyi *handaku-on* adalah bunyi yang dihasilkan dari huruf hiragana yang diberi tambahan tanda *maru [o]*. Lambang bunyi *handaku-on* berjumlah 5 huruf. Bunyi ini hanya berasa dari set huruf konsonan vokal h. Lambang bunyi *handaku-on* adalah sebagai berikut:

Tabel 2.6 Huruf Hiragana *Handaku-on*

ぱ Pa	ぴ pi	ぷ Pu	ぺ pe	ぽ po

4. Yoo-on

Bunyi *yoo-on* adalah bunyi yang dihasilkan dari huruf hiragana ki, shi, chi, ni, hi, mi, ri, dan huruf hiragana lambang bunyi *dakuon* gi, ji, pi dan pi dengan menambahkan huruf や(ya), ゆ(yu), よ(yo) yang ditulis lebih kecil daripada ukuran huruf yang seharusnya. Apabila ditulis sama dengan ukuran huruf yang seharusnya maka cara membacanya akan berbeda. Contoh : ひや *(hiya)* dan ひゃ*(hya)*. Lambang bunyi *yoo-on* berjumlah 33 huruf. Lambang bunyi *yoo-on* adalah sebagai berikut:

Tabel 2.7 Huruf Hiragana *Yoo-on*

きゃ kya	きゅ kyu	きょ Kyo
しゃ sha	しゅ shu	しょ Sho
ちゃ cha	ちゅ chu	ちょ Cho
にゃ nya	にゅ nyu	にょ Nyo
ひゃ hya	ひゅ hyu	ひょ Hyo
みゃ mya	みゅ my	みょ Myo
りゃ rya	りゅ ryu	りょ Ryo
ぎゃ gya	ぎゅ gyu	ぎょ Gyo
じゃ ja	じゅ ju	じょ jo

びゃ bya	びゅ byu	びょ Byo
ぴゃ pya	ぴゅ pyu	ぴょ Pyo

2.4.5 Cara Penulisan Huruf Jepang

Berikut cara penulisan huruf Jepang secara bertahap.

Tabel 2.8 Cara Penulisan Huruf Hiragana

2.5 Android

Android adalah sebuah OS pertangkat *mobile* berbasis Linux yang mencakup *Operating System, Middleware,* dan Aplikasi. Android merupakan OS yang dibeli oleh *Google Inc*. dari Android Inc. Android menyediakan lingkungan hidup atau *Run Time Environment* yang disebut *Dalvik Virtual Machine (DVM)* yang telah dioptimasi untuk alat dengan sistem memori kecil. Sistem ini menggunakan *platform Open Source* bagi para pengembang untuk membuat aplikasi. (Casasola, 2012 :17).

Android memiliki sifat *open platform* yang berarti perangkat android dapat dibuat serta diperjualbelikan oleh semua perusahaan *hardware* dan *provider*, serta bersifat *open source* yang berarti Android dapat digunakan dan dimodifikasi sesuai dengan kebutuhan pengembang Android baik oleh perusahaan *hardware, provider* maupun oleh *developer* aplikasi. Android juga bersifat *cross compatibility* yang dapat berjalan diberbagai perangkat dengan berbagai ukuran dan resolusi layar serta memiliki fitur *detection* yang dapat mengatur aplikasi hanya berjalan pada perangkat yang *compatible*. Aplikasi Android dapat dibuat oleh *developer* dengan menggunakan bahasa pemrograman java. Perangkat lunak yang digunakan dalam pembuatan aplikasi Android juga dapat digunakan secara gratis. Perangkat lunak pengembangan android antara lain : *Java JDK, Android SDK, Eclipse IDE dan Android ADT.*

2.5.1 Sejarah dan Perkembangan Android

Pada Juli 2005, *Google* bekerjasama dengan *Android Inc*., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri *Android Inc*. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa Google hendak memasuki pasar telepon seluler. Di perusahaan *Google*, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh kernel *Linux*. Hal ini menunjukkan indikasi bahwa *Google* sedang bersiap menghadapi persaingan dalam pasar telepon seluler.

Sekitar September 2007 sebuah studi melaporkan bahwa *Google* mengajukan hak paten aplikasi telepon seluler (akhirnya *Google* mengenalkan *Nexus On*e, salah satu jenis telepon pintar *GSM* yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh *HTC Corporation* dan tersedia di pasaran pada 5 Januari 2010). Ada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android *ARM Holdings, Atheros Communications,* diproduksi oleh *Asustek Computer Inc*, *Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc.* Seiring pembentukan *Open Handset Alliance, OHA* mengumumkan produk perdana mereka, Android, perangkat bergerak *(mobile)* yang merupakan modifikasi kernel *Linux* 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Telepon pertama yang memakai sistem operasi Android adalah *HTC Dream,* yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android.

Versi-versi android antara lain :

1. Android versi 1.1 diluncurkan *google* Pada 9 Maret 2009. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan *Gmail*, dan pemberitahuan *email*.
2. Android versi 1.5 *(Cupcake)* diluncurkan *Google* Pada pertengahan Mei 2009 dengan menggunakan *Android dan SDK (Software Development Kit*) dengan versi 1.5 *(Cupcake).* Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke *Youtube dan gambar ke Picasa langsung dari telepon, dukungan Bluetooth A2DP*, kemampuan terhubung secara otomatis ke *headset Bluetooth,* animasi layar*, dan keyboard.*
3. Android versi 1.6 *(Donut)* diluncurkan pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya,

penggunaan baterai indikator dan kontrol *applet VPN*. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus; kamera, *camcorder* dan galeri yang dintegrasikan; *CDMA / EVDO, 802.1x, VPN, Gestures, dan Text-to-speech engine*; kemampuan dial kontak; teknologi *text to change speech* .

4. Android versi 2.0/2.1 *(Eclair)* diluncurkan Pada 3 Desember 2009. perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *Google Maps 3.1.2*, perubahan UI dengan *browser* baru dan dukungan *HTML5*, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 *MP, digital Zoom, dan Bluetooth 2.1.* Untuk bergerak cepat dalam persaingan perangkat generasi berikut, *Google* melakukan investasi dengan mengadakan kompetisi aplikasi mobile terbaik (*killer apps* - aplikasi unggulan). Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik. Dengan semakin berkembangnya dan semakin bertambahnya jumlah handset Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah *Shazam, Backgrounds, dan WeatherBug.* Sistem operasi Android dalam situs Internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh *MySpace dan Facebook.*
5. Android versi 2.2 *(Froyo: Frozen Yoghurt)* diluncurkan Pada 20 Mei 2010, Android inilah sekarang sangat banyak beredar dipasaran, salah satunya adalah dipakai *Samsung FX tab* yang sudah ada dipasaran. Fitur yang tersedia di android versi ini sudah kompleks diantaranya adalah kerangka aplikasi memungkinkan penggunaan dan penghapusan kompenen yang tersedia, *Dalvik Virtual Machine* dioptimalkan untuk perangkat mobile, grafik di 2d dan *grafis 3d berdasarkan libraries OpenGL, SDlite* untuk penyimpanan data, mendukung media (audio, video, dan berbagai format gamabar *(MPEG4, H.264, MP3, AAC, AMR,*

JPG ,PNG, GIF)), GSM, Bluetooth, 3G dan *Wifi (hardware Independent),* kamera *, global positioning system (GPS),*kompas dan *accelerometer (*tergantung *Hadwere)* .

6. Android versi 2.3 *(Gingerbread)* diluncurkan Pada 6 Desember 2010,. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan *(gaming*), peningkatan fungsi copy paste, layar antar muka *(User Interface)* didesain ulang, dukungan *format video VP8 dan WebM,* efek audio baru *(reverb, equalization, headphone virtualization, dan bass boost),* dukungan kemampuan *Near Field Communication (NFC),* dan dukungan jumlah kamera yang lebih dari satu.
7. Android versi 3.0/3.1 *(Honeycomb)* Android versi 3.0/3.1 *(Honeycomb)* dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface pada Honeycomb* juga berbeda karena sudah didesain untuk *tablet. Honeycomb* juga mendukung multi prosesor dan juga akselerasi perangkat keras *(hardware)* untuk grafis. *Tablet* pertama yang dibuat dengan menjalankan *Honeycomb adalah Motorola Xoom.* Perangkat tablet dengan platform Android 3.0 akan segera hadir di Indonesia. Perangkat tersebut bernama *Eee Pad Transformer* produksi dari *Asus.* Rencana masuk pasar Indonesia pada Mei 2011.
8. Android versi 4.0 *(ICS :Ice Cream Sandwich)* diluncurkan pada tanggal 19 Oktober 2011, membawa fitur *Honeycomb* untuk *smartphone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari *e-mail* secara *offline*, dan berbagi informasi dengan menggunakan *NFC.*
9. Android versi 4.0.3.r2 memiliki video untuk sistem baru emulator yang memiliki dukungan *OpenGL ES 2.0 GPU, mengahasilkan akselerasi dari host GPU,* bahkan dapat menjalankan aplikasi permainan pada *OpenGL* emulator tersebut dengan kualitas performa ganda yang di tawarkan,

seperti yang di tawarkan *Hexus* juga (10/4/2012) emulator ini dapat dapat memanfaatkan *CPU host native melalui virtualisasi*, sehingga memberikan akses lebih cepat secara signifikan, beserta *Usability* yang telah ditingkatkan. sekarang emulator manampilkan dukungan untuk sensor dan multi-touch perangkat Android juga ditambatkan untuk perkembangan versi *Android Emulator*. *Google* juga berencana menambahkan dukungan untuk *Bluetooth dan NFC* kedepannya lebih baik.

10. Android versi 4.1 *(Jelly Bean)* Akan segera dirilis tahun 2012 ini. Android *Jelly Bean* ini diperuntukkan untuk komputer tablet dan memungkinkan untuk digunakan pada sistem operasi PC atau Komputer. Sehingga rumornya kemunculan *Android Jelly Bean* ini untuk menyaingi rilis terbaru *Windows 8* yang juga akan segera dirilis. Karena kita ketahui bersama perbincangan versi Android sebelumnya yaitu *Android Ice Cream Sandwhich* pun masih hangat di telinga. Lebih dari satu tahun sampai Android pindah dan merilis versi berikutnya (meskipun Jellybean itu terus ditingkatkan hingga musim panas 2013) dan meluncurkan KitKat pembaruan dengan Nexus 5 pada Halloween tahun 2013. Google Now sekarang lebih baik dari sebelumnya dengan beberapa kemampuan prescient abilities yang mencoba untuk menebak apa yang pengguna inginkan sebelum mereka bertanya, dan Hangouts ditingkatkan dengan beberapa kemampuan SMS yang sangat dibutuhkan.

    Yang paling penting, KitKat OS mempunyai kecepatan akses, yang hanya dengan 512MB RAM bisa menjalankan OS dengan lancar. Ini berguna untuk mendorong Android ke pasar ponsel murah dengan Salah satu program Android yang hampir sempurna.

11. Android Lolipop version 5.0

    Hello Lollipop dan halo modern Android. Klaim terbesar Lollipop untuk ketenaran adalah reimagining dramatis dari sistem operasi Android yang mulai kembali dengan Ice Cream Sandwich. Ikon, animasi, dan menu multitasking yang sepenuhnya diperbarui dengan pendekatan Material

Desain Google, dan lockscreen Android menjadi jauh lebih berguna dengan integrasi pemberitahuan yang lebih baik. Google terus mengembangkan Google Now untuk pengembang pihak ketiga dan untungnya menambahkan "mode diam" kembali untuk pemberitahuan.

12. Android marshmallow version 6.0

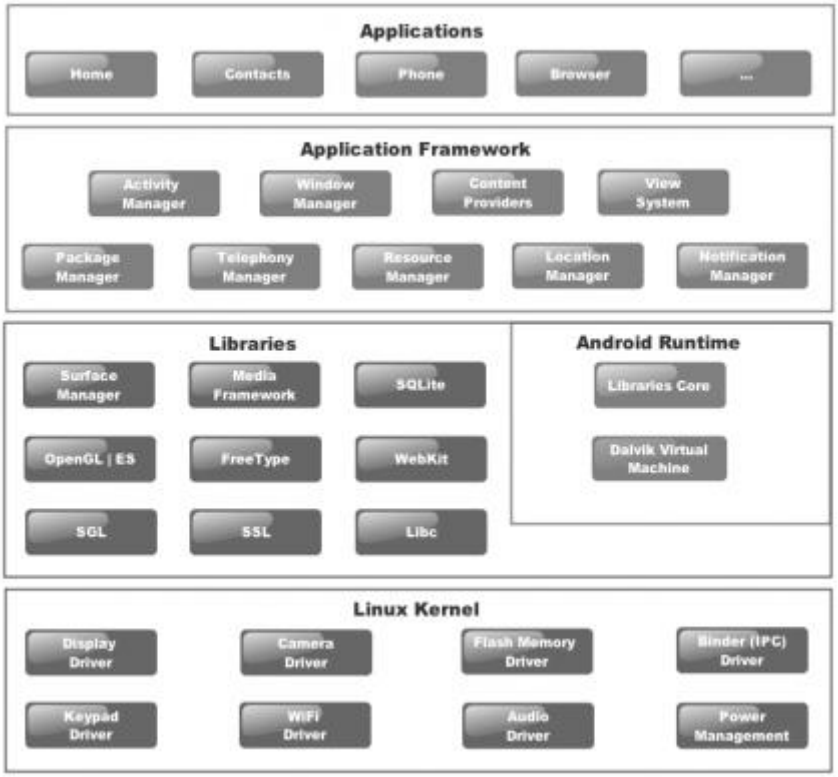
    Dan kita akhirnya tiba di Android Marshmallow, yang semua tentang pengembangan dan tidak begitu banyak makeover. Tapi itu tidak dengan beberapa tambahan mengagumkan, termasuk cara baru menangani izin aplikasi, dukungan sensor sidik jari baru dan ditingkatkan Google Now, dan legal. Juga memperpanjang waktu penggunaan baterai Google dengan Doze dan bundling di Sensor Android Hub di smartphone terbaru Nexus, yang berarti Anda akan mendapat daya tahan baterai seharian tanpa perlu mengeluarkan kabel pengisian.

Itulah sedikit ulasan mengenai perkembangan android yang begitu pesat. Dapat dsimpulkan bahwa android menjadi OS paling populer dikalangan masyarakat menengah keatas, dan dapat menemukan keasikan tersendiri jika kita menggunakan android, dikarenakan sistem ini terus mengupdate dan menyempurnakan fitur agar pengguna tidak meninggalkannya. Demikianlah pembahasan saya mengenai perkembangan OS android hingga versi 6.0 marshmallow. Perlu anda ketahui bahwa banyak hal positif dan negatif di era digital ini.

2.5.2 Arsitektur Android

Agar bisa membuat aplikasi dengan baik, tentunya kita harus mengetahui arsitektur OS Android beserta elemen elemennya.

Gambar dibawah merupakan skema pembagian elemen pada arsitektur Android. Secara garis besar arsitektur android terdiri dari beberapa layer komponen, yaitu:



Gambar 2.1 Arsitektur Android

2.1 Layer Applications dan Widget

Inilah layer pertama pada OS Android, biasa dinamakan *layer Applications* dan *Widget*. Layer ini merupakan layer yang berhubungan dengan aplikasi-aplikasi inti yang berjalan pada Android OS. Seperti klien email, program SMS, kalender, *browser*, peta, kontak, dan lain-lain. Semua aplikasi ini dibuat dengan menggunakan bahasa Java. Apabila kalian membuat aplikasi, maka aplikasi itu ada di layer ini.

2.2 Layer Applications Framework

Applications Framework merupakan layer dimana para pembuat aplikasi menggunakan komponen-komponen yang ada di sini untuk membuat aplikasi mereka. Beberapa contoh komponen yang termasuk di dalam *Applications Framework* adalah sebagai berikut:

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

5) Activity Manager

2.3 Layer Libraries

Libraries merupakan *layer* tempat fitur-fitur android berada. Pada umumnya libraries diakses untuk menjalankan aplikasi. Beberapa *library* yang terdapat pada android diantaranya adalah libraries Media untuk memutar media video atau audio,

libraries untuk menjalankan tampilan, *libraries Graphic, libraries* SQLite untuk dukungan database, dan masih banyak library lainnya.

2.4 Android RunTime

Android RunTime merupakan layer yang membuat aplikasi android bisa dijalankan. Android RunTime dibagi menjadi dua bagian yaitu:

a) *Core Libraries* : berfungsi untuk menerjemahkan bahasa Java/C
b) *Dalvik Virtual Machine* : sebuah mesin virtual berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi pada Android secara efisien.

2.5 Linux Kernel

Linux Kernel merupakan layer tempat keberadaan inti dari *operating system android*. Layer ini berisi file-file system yang mengatur *system processing, memory, resource, drivers*, dan sistem android lainnya. Inilah yang membuat file sistem pada Android mirip dengan file sistem pada sistem operasi berbasis Linux. Kernel yang digunakan adalah kernel Linux versi 2.6, dan versi 3.x pada Android versi 4.0 ke atas. Kernel ini berbasis monolithic .

2.6 Android SDK

Untuk dapat mengembangkan suatu produk aplikasi dengan berbasis sistem operasi Android, maka dibutuhkan Android SDK sebagai *tool* (alat-alat) yang dibutuhkan untuk menunjang *Eclipse* yang digunakan. Android SDK tersedia secara free (bebas digunakan dan gratis) dan menunjang berbagai platform (jenis sistem operasi) Android dari versi pertama kali diperkenalkan hingga sekarang. Dengan adanya Android SDK dalam pembuatan projrk Android dapt dipermudah.

Programmer developer dapat mengkompile hasil dari projeknya langsung dengan virtual emulator Android, yang sudah disediakan dalam tools Android SDK tersebut (Developer.android.com,2012). Dengan cara tersebut maka developer dapat dengan erat mengetahui hasil dari program yang dibuatnya berjalan lancar pada platform. Sebagai platform aplikasi netral, Android dapat membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Handphone/Smartphone.* Beberapa fitur-fitur Android yang paling penting adalah:

1. *Framework* aplikasi yang mendukung penggantian komponen dan *reusable.*
2. Mesin *Virtual Dalvik* dioptimalkan untuk perangkat *mobile*.
3. *Integrated browser* berdasarkan *engine open source WebKit.*

2.7 Finite State Machine (FSM)

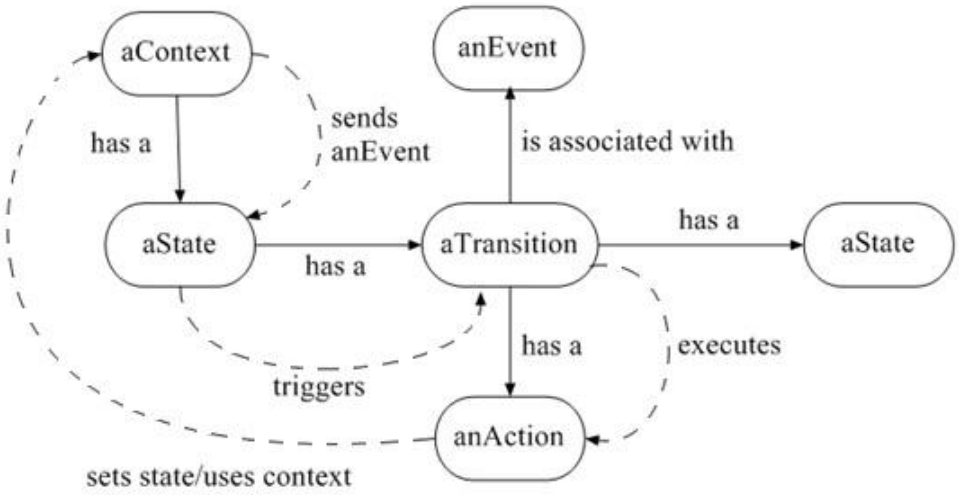
Dalam perancangan *Artificial Intelegent* untuk game, *state machine* merupakan teknik yang paling banyak digunakan untuk permasalahan "*decision making*" dan sekaligus dengan scriptingnya juga digunakan secara luas untuk merancang *system decision making* dalam *game*. *State machine* dikenal secara luas sebagai teknik untuk pemodelan fenomena atau kondisi berbasis *event*, termasuk penguraiannya, serta desain interface. *FSM (Finite State Machine*) atau juga disebut sebagai teknik yang secara luas dipergunakan dalam merancang AI dalam *game.* Teknik ini merupakan metodologi perancangan system untuk memodelkan perilaku (*behavior)* dari *sistem* atau objek yang kompleks dengan kondisi yang telah didefinisikan dalam satu set. Menurut Ian Millington [2006] dalam bukunya yang berjudul *Artificial Itelligence for Games* menyebutkan bahwa *Finite State Machines* (FSM) masuk dalam ranah *Decision Making* (pembuat keputusan) pada *Artificial Intelligence* (AI).

Dalam FSM masing-masing karakter menempati satu *state*. Biasanya, tindakan atau perilaku yang terkait dengan masing-masing *state*. Jadi selama karakter tetap dalam keadaan itu, ia akan terus melakukan tindakan yang sama. State terhubung bersama oleh transition. Setiap transition mengarah dari satu *state* ke state lain yang biasanya state tujuan *state* target ini disebut dengan *action* dan

masing-masing memiliki seperangkat kondisi yang terkait. Jika permainan menentukan bahwa kondisi transition terpenuhi, maka karakter berubah dari *state* ke *state* target (*action*) melalui *transition* itu.

FSM melacak himpunan state yang ada kemudian inputan masuk ke masing-masing *state*, serangkaian keadaan *transition* tetap. Setiap transition dapat diimplementasikan dengan kondisi yang sesuai. Pada setiap iterasi (biasanya setiap *frame*), fungsi update FSM digunakan. Ini memeriksa untuk melihat apakah ada perubahan *transition* dari kondisi saat dipicu oleh inputan. Kemudian menyusun daftar *action* dari negara yang sedang aktif. Jika *transition* telah menemukan *action* yang dituju, maka *transition* berhenti. (Ian Millington : 2006).

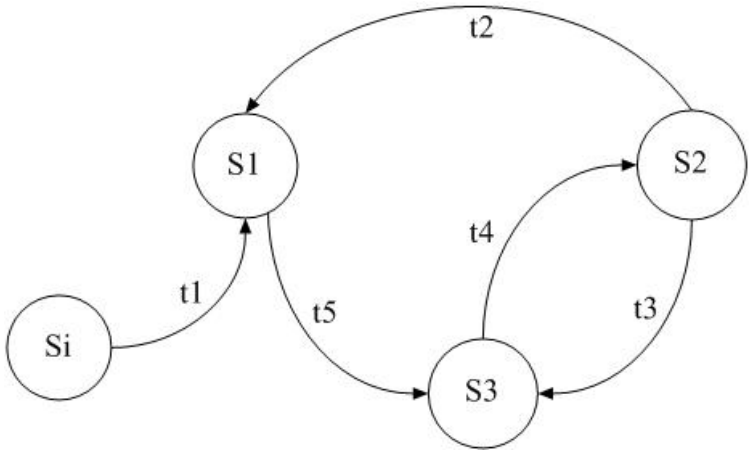
Gambar prinsip dari komponen-komponen yang terintegrasi dalam FSM pada gambar 2.2.



Gambar 2.2 *Finite State Machine* **(Sumber: Brownlee, 2010)**

Finite State Machine (FSM) adalah membagi sebuah respon objek *game* kedalam bagian-bagian *(state)* sehingga objek tersebut memilikki bagian untuk setiap respon objek *game*. Implementasi tersebut menghasilkan suatu urutan scenario tertentu pada *game*. Sehingga dalam *game* akan terdapat alur permainan yang harus di lewati nantinya yang dapat mendifisikan suatu set kondisi yang menentukan kapan suatu bagian harus berubah ke bagian yang lain.

(Rich :2009) *finite state machine* atau FSM adalah merupakan sebuah metodologi perancangan sistem control yang mengambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal berikut : *state* (keadaan), *event* (kejadian) dan *action* (aksi). Pada satu saat dalam periode waktu yang cukup signifikan, sistem akan berada pada salah satu *state* yang aktif. Sistem dapat beralih atau bertransisi menuju state lain jika mendapatkan masukan atau event tertentu, baik yang berasal dari perangkat luar atau komponen dalam sistemnya itu sendiri (misal *interupsi timer*).



Gambar 2.3 Alur dari metode Finite State Machine

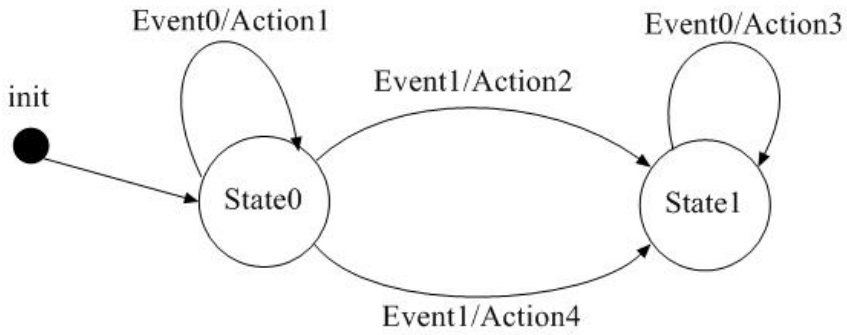
(Sumber: Brownlee, 2010)

Pada Gambar 2.3 terdapat 4 *state* {Si, S1, S2, S3} yang mungkin terjadi, setiap *state*-nya dapat berpindah *state* jika kondisi terpenuhi. Sebagai contoh *state* S1 dapat berpindah jika kondisi t5 terpenuhi.

Dalam diagram ini, *state-state* yang terdapat pada sebuah sistem digambarkan sebagai lingkaran yang diberi label unik, sedangkan transisi *state* yang diakibatkan oleh *event* tertentu direpresentasikan sebagai anak panah yang berasal dari *state* yang ditinggalkan menuju *state* yang aktif. Setiap transisi yang terjadi umumnya juga diikuti oleh aksi yang dilakukan oleh sistem yang dirancang. Secara praktis setiap diagram *state* yang dirancang akan selalu memiliki sebuah transisi awal (inisial) yang menuju salah satu *state* sejak sistem kontrol tersebut mulai dihidupkan.

Diagram keadaan pada dasarnya merupakan salah satu bentuk representasi dari FSM. Diagram ini secara visual menggambarkan tingkah laku yang dimiliki oleh sistem kontrol yang kompleks kedalam bentuk yang lebih sederhana dan relative mudah dipahami. Dalam diagram ini, *state-state* yang terdapat pada sebuah sistem digambarkan sebagai lingkaran yang diberi label unik, sedangkan transisi *state* yang diakibatkan oleh *event* tertentu direpresentasikan sebagai anak panah yang berasal dari *state* yang ditinggalkan menuju *state* yang aktif.

Setiap transisi yang terjadi umumnya juga diikuti oleh aksi yang dilakukan oleh sistem yang dirancang. Secara praktis setiap diagram *state* yang dirancang akan selalu memiliki sebuah transisi awal (inisial) yang menuju salah satu state sejak sistem kontrol tersebut mulai dihidupkan. Gambar 2.4 berikut memperlihatkan contoh penggambaran *diagram state*:



Gambar 2.4 Alur *State Machine* Pada Sebuah Program
(Sumber: Setiawan 2006)

Diagram pada Gambar 2.4 memperlihatkan FSM dengan dua buah state b dan dua buah input serta empat buah aksi output yang berbeda : seperti terlihat pada gambar, ketika sistem mulai dihidupkan, sistem akan bertransisi menuju *state0,* pada keadaan ini sistem akan menghasilkan *Action1* jika terjadi masukan *Event0,* sedangkan jika terjadi *Event1* maka *Action2* akan dieksekusi kemudian sistem selanjutnya bertransisi ke keadaan State1 dan seterusnya.

Secara formal FSM dinyatakan oleh 5 tupel atau M = {Q, Σ, δ, S, F} (Utdirartama, 2001) dimana:

Q = Himpunan state/kedudukan

Σ = Himpunan symbol input/masukan

δ = Fungsi transisi

S = State awal/kedudukan awal (initial state)

F = Himpunan state akhir

Finite State Machine bukanlah metode yang baru. FSM sudah lama ada dan konsep dekomposisi biasanya sudah dipahami dan sering digunakan oleh orang-orang yang memiliki pengalaman dalam membuat program komputer atau desain program komputer. Ada beberapa teknik pemodelan abstrak yang bisa digunakan untuk membantu defenisi atau pemahaman dan desain dari FSM, mayoritas teknik ini berasal dari disiplin ilmu desain atau matematika.

a. Diagram Transisi State Juga dikenal sebagai Diagram Gelembung *(Bubble Diagram)*.
b. Menunjukkan relasi antara state dengan input yang menyebabkan transisi state.
c. Diagram Pengambilan Keputusan State-Aksi. Diagram Alir sederhana dengan tambahan gelembung yang menunjukkan penungguan terhadap input.
d. Diagram Grafik State Salah satu bentuk dari notasi UML yang berfungsi untuk menunjukkan sifat individu dari objek sebagai nomor *state* dan transisi dari state tersebut.
e. Analisa Hirarki Perintah Meskipun tidak seperti state, ini merupakan teknik dekomposisi perintah yang
f. melihat dari sudut pandang bagaimana caranya perintah dibagi jadi sub perintah dan urut sesuai urutan kejadiannya.

Berdasarkan sifatnya, metode FSM ini sangat cocok digunakan sebagai basis perancangan perangkat lunak pengendalian yang bersifat reaktif dan real time. Salah satu keuntungan nyata penggunaan FSM adalah kemampuannya dalam mendekomposisi aplikasi yang relative besar dengan hanya menggunakan

sejumlah kecil item *state.* Selain untuk bidang kontrol, Penggunaan metode ini pada kenyataannya juga umum digunakan sebagai basis untuk perancangan protokol-protokol komunikasi, perancangan.

Implementasi *Finite State Machine* dalam perangkat lunak merupakan permasalahan tersendiri yang sudah banyak diteliti oleh pakar-pakar insinyur perangkat lunak *(software enginer)*. Desain *Finite State Machine* memang tampak mudah dan sederhana karena hanya terdiri dari serangkaian lingkaran dan anak panah yang masing-masing memiliki label. Desain FSM biasanya direpresentasikan dalam tabel transisi *state* atau dengan *state diagram*. Namun jika tiba waktunya mengimplementasikan FSM dalam suatu aplikasi perangkat lunak, maka ada suatu permasalahan yang sering timbul yaitu kode program FSM menjadi rumit dan kompleks ketika sistem yang dibangun adalah sistem yang besar atau kompleks. Bagi pemula yang masih belajar implementasi FSM dengan sistem sederhana mungkin hal ini tidak terlalu berpengaruh maupun terasa. Namun bagi seorang programer profesional, maka implementasi FSM untuk sistem yang besar atau kompleks memerlukan suatu desain struktur yang baik dan optimal.

FSM terdiri dari dua jenis, yaitu FSM ber-output dan FSM tidak ber-output. FSM tidak ber-output digunakan untuk pengenalan bahasa dalam komputer, dengan input yang dimasukkan akan diperoleh apakah input tersebut dikenal oleh bahasa komputer atau tidak. Salah satu penggunaan FSM tidak ber-output adalah program compiler, yaitu program untuk memeriksa apakah perintah yang digunakan pengguna benar atau salah. Sementara untuk FSM ber-output digunakan untuk merancang mesin atau sistem (Zen, 2008).