BAB II
LANDASAN TEORI

2.1 Konsep Kebahasaan

Dalam bagian ini akan dibahas tentang pengertian bahasa dan pembelajaran bahasa Indonesia.

2.1.1 Hakikat Bahasa

Bahasa adalah suatu sistem lambang berupa bunyi, bersifat arbitrer, digunakan oleh suatu masyarakat tutur untuk bekerja sama, berkomunikasi dan mengidentifikasi diri [4]. Sebagai sebuah sistem, maka bahasa terbentuk oleh suatu aturan, kaidah, atau pola-pola tertentu, baik dalam tata bunyi, tata bentuk kata, maupun kalimat. Bila aturan, kaidah, kaidah atau pola ini dilanggar, maka komuikasi dapat terganggu.

2.1.2 Fungsi Bahasa

Fungsi bahasa yang terutama adalah sebagai alat untuk bekerja sama atau berkomunikasi di dalam kehidupan manusia bermasyarakat. Untuk berkomunikasi sebenarnya dapat juga digunakan cara lainnya. Namun, dengan bahasa komunikasi dapat berlangsung lebih baik dan lebih sempurna.

Bahasa Indonesia sendiri yang mempunyai kedudukan sebagai bahasa nasional dan bahasa resmi negara di tengah-tengah berbagai macam bahasa daerah, mempunyai fungsi sebagai berikut: a) alat untuk menjalankan administrasi negara, b) alat pemersatu berbagai suku bangsa di Indonesia, c) media untuk menampung kebudayaan nasional.

2.2 Faktor Kebahasaan

faktor kebahasaan yang terkait dengan keterampilanberbicara antara lain sebagai berikut.

2.2.1 Ketepatan pengucapan atau pelafalan bunyi

Pembicaraan harus membiasakan diri mengucapkan bunyi-bunyi bahasa secara tepat. Hal ini dapat dilakukan dengan berlatih mengucapkan bunyi-bunyi bahasa. Memang pola ucapan dan artikulasi yang kita gunakan tidak selalu sama, masing-masing kita mempunyai ciri tersendiri. Sampai saat ini lafal bahasa

Indonesia belum dibakukan, namun usaha ke arah itu sudah lama dikemukakan adalah bahwa ucapan atau lafal yang baku dalam bahasa Indonesia adalah ucapan yang bebas dari ciri-ciri lafal dialek setempat atau ciri-ciri lafal daerah.

Di bawah ini disajikan pelafalan huruf, suku kata dan kata yang belum sesuai dengan pelafalan bunyi bahasa Indonesia.

1. **Pelafalan /c/ dengan /se/**

   WC dilafalkan /we-se/ seharusnya /we-ce/

   AC dilafalkan /a-se/ seharusnya /a-ce/

2. **Pelafalan /q/ dengan /kiu/**

   MTQ dilafalkan /em-te-kiu/ seharusnya /em-te-ki/

   PQR dilafalkan /pe-kiu-er/ seharusnya /pe-ki-er/

3. **Pelafalan /e/ sebagai /e'/**

   E dengan dilafalkan dengan / dEngan / seharusnya /d ngan/

   Ruwet dilafalkan /ruwEt/ seharusnya /ruw t/

4. **Pelafalan diftong /au/ dengan /0/**

   Kalau dilafalkan /kalo/ seharusnya /kalaw/

5. **Pelafalan diftong /ai/ dengan /e/**

   Pakai dilafalkan /pake/ seharusnya/ pakay/

2.2.2 Penempatan tekanan, nada, jeda, intonasi dan ritme

Penempatan tekanan, nada, jeda, intonasi dan ritme yang sesuai akan merupakan daya tarik tersendri dalam berbicara, bahkan merupakan factor penentu dalam berbicara. Suatu topic pembicaraan mungkin akan kurang menarik, namun dengan tekanan, nada, jangka dan intonasi yang sesuai akan mengakibatkan pembicara kurang menarik. Sebaliknya, apabila penyampaiannya datar saja, dapat menimbulkan kejemuan bagi pendengar dan ketepatan berbicara akan berkurang.

2.2.3 Pemilihan kata dan ungkapan yang baik, konkret dan bervariasi

Pemilihan kata dan ungkapan harus konkret, maksudnya pemilihan kata atau ungkapan harus jelas, mudah dipahami para pendengar. Kata-kata yang jelas biasanya kata yang sudah dikenal oleh pendengar yaitu kata-kata popular.

Pemilihab kata atau ungkapan yang abstrak akan menimbulkan kekurangjelasan pembicaraan.

2.2.4 Ketepatan Susunan Penuturan

Susunan penuturan berhubungan dengan penataan pembicaraan atau uraian tentang sesuatu. Hal ini menyangkut penggunaan kalimat. Pembicaraan yang menggunakan kalimat efektif akan lebih baik memudahkan pendengar menangkap isi pembicaraan.

2.3 Model Analisis Kesalahan Berbahasa

Proses berkomunikasi perlu menggunakan bentuk kata dan pelafalan yang tepat [32]. Hal ini agar gagasan dan ide-ide inovatif yang anda sampaikan kepada orang lain dapat dipahami secara efektif. Agar hal itu dapat terwujud, perlu kita memiliki pemahaman dan kemampuan menganalisis kesalahan berbahasa. Model-model yang disajikan berikut adalah model-model analisis kesalahan berbahsa Indonesia yang dikembangkan oleh Tarigan (1970) dalam buku Analisis Kesalahan Berbahasa.

2.3.1 Kesalahan Berbahasa

Kesalahan berbahasa Indonesia dalam bidang fonologi (menyelidiki bunyi-bunyi bahasa menurut fungsinya) pertama-tama dipandang dari penggunaan bahasa apakah secara lisan dan apakah secara tulisan. Baik bahasa lisan maupun bahasa tulisan dikaitkan dengan tataran fonologi. Dari kombinasi kedua sudut pandang itu kita temukan aneka jenis kesalahan berbahasa. Ada kesalahan berbahasa karena perubahan pengucapan fonem (bunyi bahasa yang berfungsi membedakan arti kata), penghilangan fonem, penambahan fonem, salah meletakkan penjedaan dalam kelompok kata dan kalimat. Di samping itu kesalahan berbahasa dalam bidang fonologi dapat pula disebabkan oleh perubahan bunyi diftong menjadi bunyi tunggal atau fonem tunggal.

Berikut ini disajikan berbagai kesalahan berbahasa. Perlu ditambahkan bahwa dalam setiap jenis kesalahan tersirat penyebab kesalahan berbahasa tersebut.

1. Pelafalan pengucapan kata /a/ Diubah menjadi /e/

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	Kesalahan pelafalan	Semestinya dilafalkan
Akta	/akte/	/akta/
Frasa	/frase/	/frasa/
Rabu	/rebo/	/rabu/
Sekadar	/sekedar/	/sekadar/

2. Pelafalan pengucapan kata /e/ diubah menjadi /a/

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	Kesalahan pelafalan	Semestinya dilafalkan
Metode	/metoda/	/metode/
Kategori	/katagori/	/kategori/
Seluler	/selular/	/seluler/

3. Pelafalan pengucapan kata /e/ diubah menjadi /i/

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	Kesalahan pelafalan	Semestinya dilafalkan
Antre	/antri/	/antre/
Apotek	/apotik/	/apotek/
Atlet	/atlit/	/atlet/
Atmosfer	/atmosfir/	/atmosfer/
Cedera	/cidera/	/cedera/
Penalti	/pinalti/	/penalti/
Peranti	/piranti/	/peranti/
sistem	/sistim/	/sistem/

4. Pelafalan pengucapan kata /i/ diubah menjadi /e/

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	Kesalahan pelafalan	Semestinya dilafalkan
Hakikat	/hakekat/	/hakikat/
Nasihat	/nasehat/	/nasihat/
Praktik	/praktek/	/praktik/
Risiko	/resiko/	/risiko/

Ritsleting	/retsleting/	/ritsleting/
Senin	/senen/	/senin/

5. Pelafalan pengucapan kata /e/ diubah menjadi /u/.

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	**Kesalahan pelafalan**	**Semestinya dilafalkan**
Penggawa	/punggawa/	/penggawa/

6. Pelafalan pengucapan kata /f/ diubah menjadi /p/.

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	**Kesalahan pelafalan**	**Semestinya dilafalkan**
Foto	/photo/	/foto/

7. Pelafalan pengucapan kata /u/ dilafalkan /i/.

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	**Kesalahan pelafalan**	**Semestinya dilafalkan**
bus	/bis/	/bus/

8. Pelafalan pengucapan kata /o/ dilafalkan menjadi /u/.

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	**Kesalahan pelafalan**	**Semestinya dilafalkan**
Pastor	/pastur/	/pastor/
Sopir	/supir/	/sopir/

9. Pelafalan fonem /u/ dilafalkan menjadi /o/.

   Kata-kata yang mengalami perubahan pelafalan antara lain adalah:

Penulisan	**Kesalahan pelafalan**	**Semestinya dilafalkan**
Guncang	/goncang/	/guncang/
Gua	/goa/	/gua/
Kukuh	/kokoh/	/kukuh/
Kumulatif	/komulatif/	/kumulatif/
Lubang	/lobang/	/lubang/

2.4 Perkembangan Anak

Perkembangan anak dalam aspek motorik [24]:

a. Masa Pralahir

Masa ini cukup penting karena pada saat inilah terbentuknya potensi-potensi manusia, potensi mana berpengaruh pada perkembangan selanjutnya.

b. Masa Bayi

Masa ini banyak disebut-sebut sebagai berlangsung dari saat bayi lahir sampai berumur 2 tahun. Beberapa penelitian membuktikan bahwa bayi-bayi mungil ini sebenarnya sudah memiliki sejumlah kesanggupan untuk belajar melakukan pilihan atau kesanggupan membeda-bedakan.

c. Masa Anak Pra-Sekolah

Masa ini disebut juga masa kanak-kanak awal, terbentang antar umur 2-6 tahun. Perkembangan bahasa dan berpikir : sebagai alat komunikasi dan mengerti dunianya, kemampuan berbahasa lisan pada anak akan berkembang karena selain terjadi oleh organ bicara dan fungsi berpikir, juga lingkungan ikut membantu mengembangkannya.

d. Masa Anak-Sekolah

Masa ini disebut juga masa tenang atau masa latent, terbentang umur 6 – 12 tahun. Tahap usia ini disebut juga sebagai usia kelompok, di mana anak mulai mengalihkan perhatian dan hubungan intim dalam keluarga ke kerjasama antar teman dan sikap-sikap terhadap kerja atau belajar.

Dengan memasuki SD salah satu hal terpenting yang perlu dimiliki anak adalah kematangan sekolah, meliputi kecerdasan dan keterampilan motorik, bahasa.

2.5 Skala Pengukuran

Skala pengukuran merupakan kesepakatan yang digunakan sebagai acuan untuk menentukan panjang pendeknya interval yang ada dalam alat ukur, sehingga alat ukur tersebut bila digunakan dalam pengukuran akan menghasilkan data kuantitatif [25].

a. Skala *Linkert*: skala *Linkert* digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang fenomena sosial.

Skala *Linkert* mempunyai gradiasi dari positif sampai negative yang dapat berupa kata-kata antara lain: Sangat Setuju (SS), Setuju (S), Ragu-Ragu (RR), Tidak Setuju (TS), Sangat Tidak Setuju (STS).

b. Skala *Guttman*: skala pengukuran dengan tipe ini, akan didapat jawaban yang tegas yaitu “ya-tidak”, “benar-salah”, “pernah-tidak pernah”, “positif-negatif” dan lain-lain. Penelitian menggunakan skala guttman bila ingin mendapatkan jawaban yang tegas terhadap suatu permasalahan yang ditanyakan. Skala guttman selain dapat dibuat dalam bentuk pilihan ganda, juga dapat dibuat dalam bentuk *checklist.* Jawaban dapat dibuat skor tertinggi satu dan terendah nol.

2.6 Dasar Penentu Jumlah Sampel

Pedoman penentu jumlah sampel sebagai berikut [33]:

1. Sebaiknya ukuran sampel di antara 30 s/d 500 elemen.
2. Jika sampel dipecah lagi ke dalam subsampel (laki/perempuan, SD/SLTP/SMU, dsb), jumah minimum subsampel harus 30.
3. Pada penelitian multivariate (termasuk analisis regresi mutiaveriate) ukuran sampel harus beberapa kali lebih besar (10kali) dari jumlah variable yang akan dianalisis.
4. Untuk penelitian eksperimen yang sederhana, dengan pengendalian yang ketat, ukuran sampel bisa antara 10 s/d 20 elemen.

2.7 Pengenalan Suara

Suara atau ucapan adalah suatu teknik yang memungkinkan sistem komputer untuk menerima *input* berupa kata yang diucapkan. Kata-kata tersebut diubah bentuk menjadi sinyal *digital* dengan cara mengubah gelombang suara menjadi sekumpulan angka lalu disesuaikan dengan kode-kode tertentu dan dicocokkan dengan suatu pola yang tersimpan dalam suatu perangkat [7]. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi. Ukuran kosakata dari sistem pengenalan suara mempengaruhi kompleksitas, paramater pelatihan dan akurasi sistem. Beberapa aplikasi pengenalan suara hanya memerlukan beberapa kata, sedangkan

yang lainnya memerlukan kamus yang sangat besar. Terdapat 4 jenis ukuran kosakata yaitu:

a. Kosakata ukuran kecil (*small vocabulary*) yang terdiri dari puluhan kata.
b. Kosakata ukuran sedang (*medium vocabulary*) yang terdiri dari ratusan kata.
c. Kosakata ukuran besar (*large vocabulary*) yang terdiri dari ribuan kata,
d. Kosakata ukuran sangat besar (*very large vocabulary*) yang terdiri puluhan ribu kata.

Teknologi pengenalan suara berkembang pesat. Beberapa jenis diantaranya yaitu:

a. Sintesis suara *(speech synthesis)*: dari teks ke suara.
b. Pengenalan pembicara *(speaker recognition)*: dari suara ke identitas pembicara.
c. Pendiktean *(dictation)*: dari suara ke teks.
d. Rangkuman suara *(speech summarization)*: dari suara ke teks sederhana.
e. Pengkategorian suara *(speech categorization)*: dari suara ke label kelas.
f. Pengertian suara *(speech understanding)*: dari suara ke representasi makna suara.
g. Pemrosesan dialog *(dialog processing)*: dari suara ke makna suara yang interaktif.
h. Penerjemah suara *(speech translation)*: dari suara ke suara dalam bahasa lain.

Kekurangan pengenalan suara yaitu:

1. Rawan terhadap gangguan sinyal suara lain terutama di tempat yang ramai.
2. Kata-kata yang diucapkan dapat sulit dikenali karena cara pengucapan yang berbeda walaupun oleh pembicara yang sama. Intonasi, logat, dan kecepatan pengucapan sangat mempengaruhi.
3. Bahasa lisan seringkali diucapkan tidak sesuai dengan kaidah tata bahasa yang baku.
4. Jumlah kata yang dapat dikenali masih terbatas.

2.8 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android menyediakan platform yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka [28].

2.8.1 Arsitektur Android

Secara garis besar Arsitektur Android dapat dijelaskan dan digambarkan sebagai berikut :

1. *Applications* dan *Widgets*

Applications dan *Widgets* ini adalah *layer* di mana kita berhubungan dengan aplikasi saja, di mana biasanya kita *download* aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut.

2. *Applications Frameworks*

Applications Frameworks adalah *layer* di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada *layer* inilah aplikasi dapat dirancang dan dibuat, seperti *content-providers* yang berupa sms dan panggilan telepon.

3. *Libraries*

Libraries adalah *layer* di mana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya.

4. *Android Run Time*

Android Run Time adalah *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.

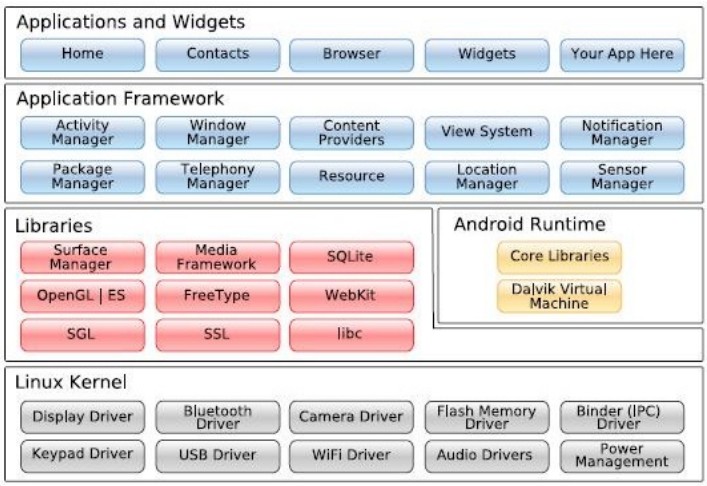
Di dalam *Android Run Time* dibagi menjadi dua bagian yaitu :

a. *Core Libraries*, Aplikasi Android dibangun dalam bahasa java, sementara *Dalvik* sebagai virtual mesinnya 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 adalah *layer* di mana inti dari *operating system* dari Android itu berada. Berisi file-file sistem yang mengatur sistem *processing, memory, resource, drivers*, dan sistem-sistem operasi android lainnya.



Gambar 2.1 Arsitektur Android

2.8.2 Fitur Android

Beberapa fitur Android yang paling penting adalah:

1. *Framework*, aplikasi yang mendukung penggantian komponen dan *reusable*.
2. Mesin Virtual Dalvik, dioptimalkan untuk perangkat *mobile*.
3. *Intergrated Browser*, berdasarkan engine open source WebKit.
4. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional akselerasi *hardware*).
5. SQLite, untuk penyimpanan data (database).
6. Media Support yang mendukung audio, video, dan gambar (MPEG4, MP3, AAC, AMR, JPG, PNG, GIF). (tergantung hardware).
7. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware).
8. Kamera, GPS, kompas (tergantung hardware).

9. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat emulator, *tools* untuk *debugging*, profil dan kinerja memori, dan *plugin* untuk IDE Eclipse.

2.8.3 Versi Android

Telepon pertama yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Adapun versi-versi Android yang pernah dirilis adalah sebagai berikut:

1. Android versi 1.1

   Dirilis pada 9 Maret 2009, versi ini dilengkapi dengan pembaruan estetis pada aplikasi jam, alarm, *voice search*, pengiriman pesan dengan *Gmail*, dan pemberitahuan *email*.

2. Android Versi 1.5 (cupcake)

   Dirilis pada pertengahan Mei 2009, adanya penambahan fitur seperti kemampuan merekam dan menonton video dengan modus kamera, meng-upload video ke *YouTube* dan gambar *Picasa* langsung dari telepon, animasi layar dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

3. Android versi 1.6 (donut)

   Dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah kemampuan dial kontak, teknologi *text to change speech* (tergantung ponsel).

4. Android versi 2.0/2.1 (eclair)

   Dirilis 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 baru, dukungan *flash* untuk kamera 3.2MP, *digital zoom*, dan bluetooth 2.1.

5. Android versi 2.2 (froyo)

   Dirilis pada Mei 2010, dengan fitur kerangka aplikasi memungkinkan penggunaan dan penghapusan komponen yang tersedia, Dalvik Virtual Machine dioptimalkan untuk perangkat mobile, grafik di 2D dan grafis 3D berdasarkan libraries opengl, SQLite sebagai penyimpanan data.

6. Android versi 2.3 (gingerbread)

   Dirilis pada Desember 2010, memiliki kemampuan seperti : *SIP-based VoIP, Near Field Communication* (NFC), *Gyroscope* dan sensor, *Multiple cameras support, Mixable audio effects, Download manager*.

7. Android versi 3.2 (honeycomb)

   Dirilis pada 22 Februari 2011, terdiri dari beberapa fitur baru diantaranya : *New UI framework for creating great tablet apps, High-performance 2D and 3D graphics, Support for multicore processor architectures, Rich multimedia and connectivity, Enhancements for enterprise, Compatibility with existing apps*.

8. Android versi 4.0 (ice cream sandwich)

   Dirilis pada 19 Oktober 2012, memiliki beberapa fitur seperti : *Refined, evolved UI, Multitasking, Home screen folders and favorites tray, Resizable widgets, New lock screen actions, Quick responses for incoming calls, Improved text input and spell-checking, Powerful voice input engine, Control over network data, Live Effects for transforming video*.

9. Android versi 4.1/4.2/4.3 (jelly bean)

   Dirilis pada 9 Juli 2013, memiliki fitur seperti : *Faster, Smoother, More Responsive, OpenGL ES 3.0 for High-Performance Graphics, Enhanced Bluetooth Connectivity, Support for Restricted Profiles, Optimized Location and Sensor Capabilities, Modular DRM framework, VP8 encoder, Video encoding from a surface, Media muxer, Playback progress and scrubbing in remote control clients, Support for International Users*.

10. Android versi 4.4 (KitKat)

    Dirilis pada 22 November 2013, memiliki fitur seperti : *Just say "Ok Google", A work of art, Immerse yourself, Faster multitasking, The future is calling, A smarter caller ID, All your messages in the same place, Emoji everywhere, Print wherever, whenever, Pick afile and any file, Your office anywhere.*

2.8.4 Kelebihan dan Kekurangan Android

Kelebihan dan kekurangan sistem operasi Android adalah sebagai berikut:

1. Kelebihan Android

a. Lengkap (*Complete Platform*): para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan *platform* Android. Android merupakan sistem operasi yang aman dan banyak menyudiakan *tools* dalam membangun *software* dan memungkinkan untuk peluang pengembangan aplikasi.

b. Terbuka (*Open Source)*: *Platform* Android disediakan melalui lisensi *open source*. Pengembang dapat dengan bebas untuk mengembangkan aplikasi.

c. Free (*Free Platform*): Android adalah *platform*/aplikasi yang bebas untuk dikembangkan. Tidak ada lisesnsi atau biaya royalti untuk dikembangkan pada *platform* Android. Tidak ada biaya keanggotaan diperlukan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Android dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

2. Kekurangan Android

a. Kurangnya ketersediaan

Berdasarkan peringkat harian agregat pada semester pertama di tahun ini, dari 50 aplikasi terlaris iPad baik yang berbayar maupun gratis di AppStore rupanya sebesar 30 persen diantaranya tidak tersedia di Google Play.

b. Ancaman dari malware/virus

Karena android adalah *open source* dan pengguna dapat menginstall aplikasi yang bukan berasal dari Google Play, perangkat Android jauh lebih rentan terkena malware dan aplikasi bajakan palsu.

c. Sering terdapat iklan

Karena mudah dan gratis, kadang sering diboncengi iklan. Secara tampilan memang tidak mengganggu kinerja aplikasi itu sendiri, karena memang kadang berada di bagian atas atau bawah aplikasi.

d. Terhubung dengan internet

Android bisa dibilang sangat memerlukan koneksi internet yang aktif. Setidaknya harus ada koneksi internet GPRS di daerah anda, agar perangkat siap untuk online sesuai dengan kebutuhan kita.

2.8.5 Peringkat Sistem Operasi

Peringkat sistem operasi untuk tahun 2013 sebagai berikut:

Worldwide Smartphone Sales to End Users by Operating System in 2Q13 (Thousands of Units)

Operating System	2Q13 Units	2Q13 Market Share (%)	2Q12 Units	2Q12 Market Share (%)
Android	177,898.2	79.0	98,664.0	64.2
iOS	31,899.7	14.2	28,935.0	18.8
Microsoft	7,407.6	3.3	4,039.1	2.6
BlackBerry	6,180.0	2.7	7,991.2	5.2
Bada	838.2	0.4	4,208.8	2.7
Symbian	630.8	0.3	9,071.5	5.9
Others	471.7	0.2	863.3	0.6
Total	**225,326.2**	**100.0**	**153,772.9**	**100.0**

Source: Gartner (August 2013)

Gambar 2.2 Peringkat Sistem Operasi 2013 [10]

2.9 Tools

Pada sub bab ini akan dijelaskan *tools* yang digunakan dalam pembangunan aplikasi lafal bahasa Indonesia ini.

2.9.1 Java

Bahasa Java dikembangkan oleh *Sun Microsystem* tahun 1991 sebagai bagian dari suatu proyek penelitian untuk mengembangkan software bagi konsumer barang – barang elektronik seperti televise, *VCR*, *toaster* dan mesin – mesin lainnya yang dapat dibeli di swalayan. Tujuan penciptaan Java pada waktu itu adalah menjadi suatu program yang berukuran kecil, efisien, dan *portable* di segala jenis *hardware*. Tujuan yang sama ini membuat Java menjadi satu bahasa yang ideal untuk mendistribusikan program – program yang dapat dijalankan melalui *www* dan juga suatu bahasa pemrograman untuk segala tujuan untuk mengembangkan program – program yang dapat digunakan dengan mudah dan *portable* di berbagai *platform* yang berbeda [13].

Sekarang, *Sun* telah mengeluarkan berbagai program Java yang dapat digunakan seperti *Java API,* atau JDK atau *JAVA Developer Kit* . Selain itu, banyak juga program-program lain yang dapat digunakan untuk membuat program Java, seperti *Eclipse, NetBeans, JBuilder, JCreator, J++,* dan sebagainya.

Saat ini distribusi Java dan kelas pendukungnya dibagi dalam tiga bagian yang masing-masing memiliki konsentrasi tersendiri yaitu [13]:

1. *Java 2 Standart Edition* (J2SE), untuk aplikasi *desktop*
2. *Java 2 Enterprise Edition* (J2EE), untuk aplikasi *server*
3. *Java 2 Micro Edition* (J2ME), untuk piranti dengan kemampuan terbatas.

2.9.2 *Application Programming Interface* (API)

Google API bisa dikatakan bagian dari *framework* google. Antarmuka pemrograman aplikasi (*application programming interface*) adalah 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 [11].

Google menyediakan berbagai API 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.

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.
8. Speech API: memanfaatkan fitur speech yang ada pada Google.

2.9.3 *Speech Recognition*

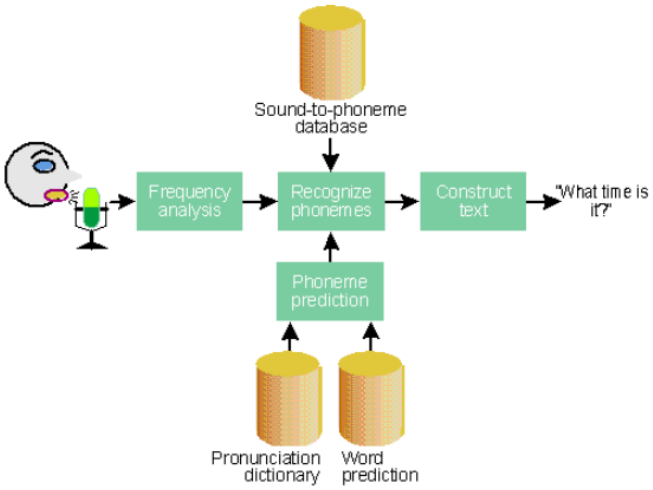
Ketika pengguna berbicara, gelombang suara diubah menjadi audio digital dengan *sound card* komputer. Biasanya, sampel audio berada di 11 KHz dan 16 bit [31]. Mentahan audio pertama-tama diubah oleh modul analisis frekuensi ke format yang lebih berguna. Ini melibatkan banyak pemrosesan sinyal digital yang terlalu rumit untuk dijelaskan. Tantangan yang paling dasar adalah untuk mengekstrak informasi suara yang bearti dari data mentahan audio. Jika anda mengatakan kata “saya”, dan kemudian berkata “saya” lagi, dan melihat bentuk gelombang yang dihasilkan, gelombang tersebut akan terlihat sama, tetapi tidak ada cara anda bisa membandingkan gelombang tersebut yang secara konsisten akan mengenali gelombang tersebut sebagai suara yang sama, tanpa menerapkan beberapa teknik matematika dengan menggunakan transformasi *Fourier*.

Audio yang telah dikonversi selanjutnya dipecah menjadi fonem oleh modul *phoneme recognition*. Modul ini mencari *database* suara-ke-fonem untuk fonem yang paling sesuai sengan suara yang didengar. Setiap *database* yang masuk berisi *template* yang menggambarkan seperti apa fonem tertentu yang terdengar. Seperti *text-to-speech*, sebuah tabel biasanya memiliki beberapa ribu masukan. Sementara tabel fonem secara teori sama dengan yang digunakan untuk TTS, dalam prakteknya mereka berbeda karena mesin SR dan TTS biasanya datang dari vendor yang berbeda.

Karena membandingkan data audio terhadap beberapa ribu fonem memakan waktu yang lama, mesin *speech recognition* berisi sebuah modul *phoneme prediction* yang mengurangi jumlah kandidat dengan memprediksi fonem yang mungkin terjadi dalam konteks tertentu. Sebagai contoh, beberapa fonem jarang terjadi pada awal kata, seperti suara “ft” di akhir kata “raft”. Fonem lain tidak pernah terjadi berpasangan. Dalam bahasa inggris, suara “f” tidak pernah terjadi sebelum suara “s”. Tetapi bahkan dengan optimasi ini, *speech recogntion* masih membutuhkan waktu terlalu lama.

Database *word prediction* digunakan untuk mengurangi daftar calon fonem dengan menghilangkan fonem yang tidak menghasilkan kata-kata yang valid. Setelah mendengar, “mas a,” *recognizer* akan mendengarkan “a” dan “k” karena “masa” dan “masak” adalah kata-kata yang valid. Hal ini juga akan mendengarkan “m” dalam kasus ketika anda berkata “Yemen”. Ini tidak akan mendengarkan “k” karena “yek” bukanlah kata yang valid. (kecuali pada *baby-talk*, yang saat ini tidak didukung) daftar kandidat dapat dikurangi lebih jauh jika aplikasi tersebut menetapkan bahwa hanya mengharapkan kata-kata tertentu. Jika aplikasi hanya ingin mengetahui jika pengguna mengatakan”yes” atau “no”, fonem *recognizer* tidak perlu mendengarkan “n” berikut “y eh,” meskipun “yen” adalah sebuah kata. Setelah fonem diakui, fonem-fonem tersebut dipecah menjadi kata-kata, dikonversi ke teks *string*, dan diteruskan ke aplikasi.

Seperti yang dibayangkan, baik *teks-to-speech* dan *speech recognition* melibatkan sedikit pengolahan/proses, tetapi *speech recognition* lebih sulit karena biasanya membutuhkan pengolahan/proses lebih untuk kepuasan pengguna.



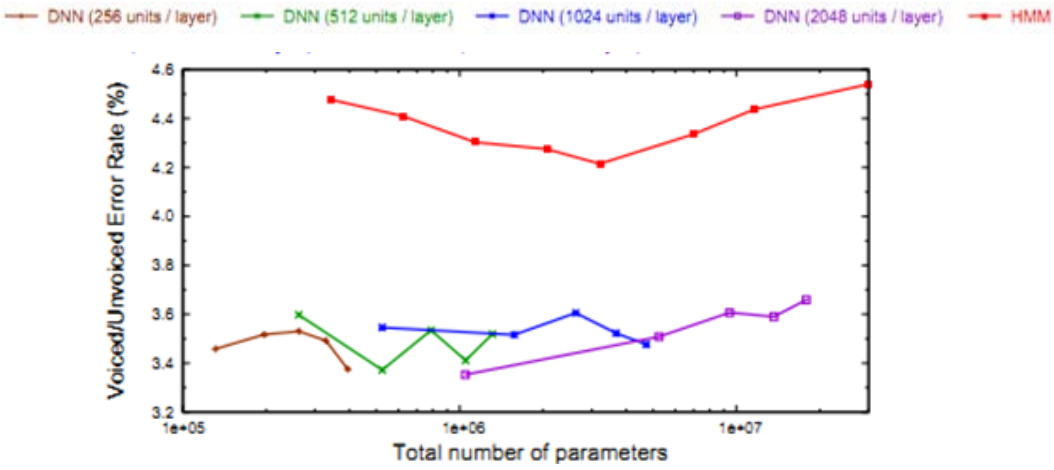
Gambar 2.3 *Speech Recognition*

2.9.4 Deep Neural Network

Menurut Vanhoucke [15] neural network memberikan peneliti cara menganalisis banyak pola dalam kasus android, spektogram dari kata yang diucapkan kemudian memprediksi pola baru apa yang mungkin terjadi. Google memproses perintah suara menggunakan algoritma neural network untuk mencari tahu apa yang baru dikatakan.

Naural network adalah suatu sistem yang terkomputerisasi yang berperilaku seperti otak manusia. Menurut jurnal [14] penggunaan naural network mengungguli sistem berbasis *Hidden Markkov Model (*HMM*)* dalam *voice recognition.*

Berikut grafik *voice error Hidden Markov Model* dan *Naural Nertwork*:



Gambar 2.4 *voice error Hidden Markov Model* dan *Naural Nertwork*

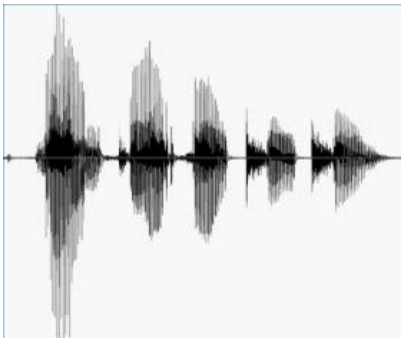
2.9.5 Transformasi Fourier

Transformasi Fourier merupakan metode untuk merubah suara waktu menjadi suara frekuensi (*analog to digital*) [22]. Artinya proses perekaman suara disimpan dalam bentuk *digital* berupa gelombang spectrum suara berbasis frekuensi. Transformasi penting dalam proses analisa suara. Setiap karakteristik suara dalam frekuensi lebih mudah diamati dari pada dalam suara waktu [12].

2.9.5.1 Proses Pencocokan Pola Suara

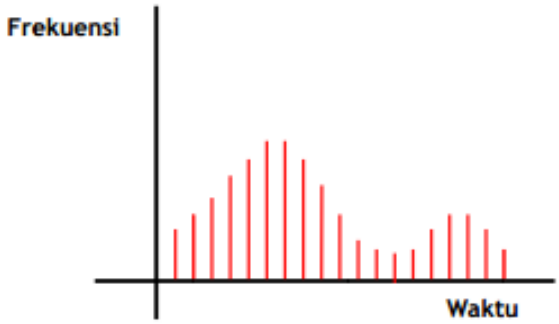
Secara umum, *speech recognizer* memproses sinyal suara yang masuk dan menyimpannya dalam bentuk digital [12].

Proses yang pertama kali dilakukan ialah memproses gelombang kontinu spektrum suara ke dalam bentuk distrit. Berikut contoh gambaran spektrum suara:



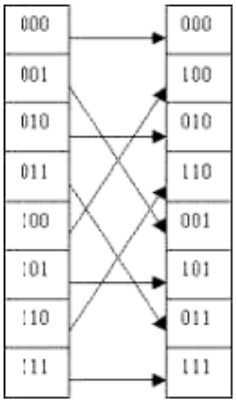
Gambar 2.5 spektrum suara

Berikut gambar spektrum suara yang sudah dikonversi dalam bentuk distrit:



Gambar 2.6 suara dalam bentuk distrit

Dari tiap elemen data tersebut, dikonversi ke dalam bentuk biner. Data biner tersebut yang nantinya akan dibandingkan dengan data suara yang ada di database. Berikut gambar gelombang dari hasil distrit menjadi bentuk biner:



Gambar 2.7 suara dalam bentuk biner

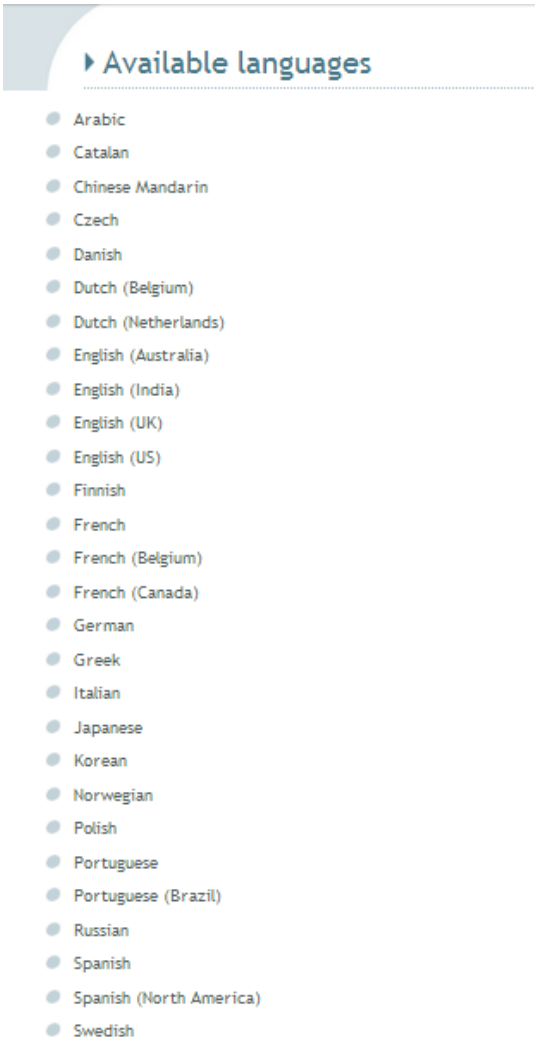
2.9.6 *Developing Speech Mobile*

Ada beberapa pengembang *speech* API diantaranya :

1. *Acapela*

   Acapela online text-to-speech mobile, memberi pengembang *mobile* akses penuh dengan suara kualitas tertinggi dan bahasa dengan control dan biaya rendah, bebas dari segala masalah.

   Bahasa yang tersedia dalam acapela sebagai berikut :



Gambar 2.8 Languages Acapela API [1].

2. AT & T

   AT & T Speech API memberikan solusi mudah *speech-to-text* dan *text-to-speech* untuk aplikasi anda. Bahasa yang tersedia dalam API sebagai berikut:

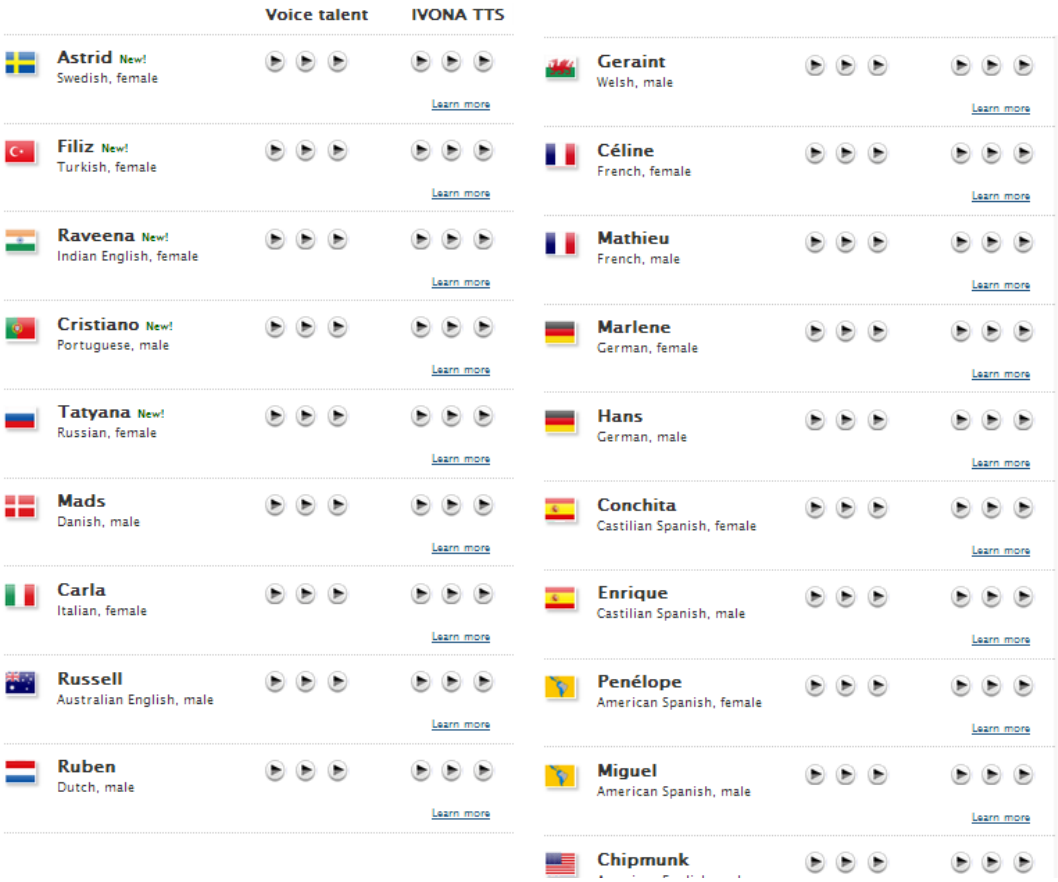
- en-US: English in the United States of America.
- es-US: Spanish in the United States of America.
- es-ES: Spanish in Spain.
- en-GB: English in Great Britain.
- fr-FR: French in France.
- it-IT: Italian in Italy.
- de-DE: German in Germany.
- zh-CN: Chinese in China. No Pronunciation Lexicon Specification (PLS) supported.
- ja-JP: Japanese in Japan. No Pronunciation Lexicon Specification (PLS) supported.
- ko-KR: Korean in Korea.
- pt-PT: Portuguese in Portugal.
- pt-BR: Portuguese in Brazil.
- uk-UA: Ukrainian in Ukraine.
- ru-RU: Russian in Russia.
- id-ID: Indonesian in Indonesia.
- tr-TR: Turkish in Turkey.
- ar-EG: Arabic in Egypt.
- nl-NL: Dutch in the Netherlands.
- sv-SE: Swedish in Sweden.

Gambar 2.9 Languages AT & T API [3].

3. Ivona

   Ivona API speech kualitas tinggi untuk setiap sistem atau interpretasi suara platform. Suara interprestasi natural dalam bentuk teks panjang. Pendekatan yang sangat fleksibel untuk setiap pelaksanaan.

   Adapun bahasa yang disediakan dari Ivona API sebagai berikut :



Gambar 2.10 Languages Ivona API [18].

2.9.7 Android SDK (*Software Development Kit*)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk memulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Sebagai platform aplikasi-netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan yang bukan merupakan aplikasi bawaan *smartphone*.

2.9.8 Eclipse

Pada bagian ini akan dijelaskan mengenai pengertian dan sifat Eclipse, berbagai versi Eclipse, serta Arstitektur Eclipse.

2.9.6.1 Pengertian dan Sifat Eclipse

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

1. Multi-platform : Target sistem operasi Eclipse adalah 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, Python, Perl, PHP, dan lain sebagainya.
3. Multi-role : Selain sebagai IDE untu pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

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

2.9.6.2 Versi Eclipse

Sejak tahun 2005, Eclipse Foundation mengkoordinasikan peluncuran Eclipse secara rutin dan simultan yang dikenal dengan nama *Simlutaneous Release*. Tujuannya adalah untuk menyediaan distribusi Eclipse dngan fitur dan versi yang terstandarisasi.

Tabel 2.1 Versi Eclipse [9]

Kode Peluncuran	Tanggal Peluncuran	Platform
Eclipse 3.0	28 Juli 2004	3.0
Eclipse 3.1	28 Juni 2005	3.1
Callisto	30 Juni 2006	3.2
Europa	29 Juni 2007	3.3

Tabel 2.1 Versi Eclipse [9] (lanjutan)

Kode Peluncuran	Tanggal Peluncuran	Platform
Ganymede	25 Juni 2008	3.4
Galileo	24 Juni 2009	3.5
Helios	23 Juni 2010	3.6

2.9.6.3 ADT/Plugin Eclipse

Android Development Tools (ADT) adalah *plugin* yang didesain untuk IDE Eclipse yang memberikan kita kemudahan dalam mengembangkan aplikasi android dengan menggunakan IDE Eclipse. Dengan menggunakan ADT untuk Eclipse akan memudahkan kita dalam membuat aplikasi project android, membuat GUI aplikasi, dan menambakan komponen-komponen yang lainnya, begitu juga kita dapat melakukan running aplikasi menggunakan Android SDK melalui Eclipse. Dengan ADT juga kita dapat melakukan pembuatan package android (.apk) yang digunakan untuk distribusi aplikasi android yang kita rancang.

2.9.6.4 Android Virtual Device (AVD)

AVD adalah emulator untuk menjalankan program aplikasi android yang kita buat. Dengan AVD ini, kita bisa mengembangkan dan mencoba aplikasi Android tanpa harus menggunakan perangkat Android yang sebenarnya. Kita bisa menentukan karakteristik AVD, misalkan menentukan versi Android, jenis dan ukuran layar, besarnya memory, dan lain sebagainya. AVD bisa dibuat sebanyak yang kita inginkan.

2.9.7 Pemrograman Berorientasi Objek

Pada bagian ini dijelaskan tentang pengertian, bahasa pemrograman, dan konsep dasar pemograman berorientasi objek.

2.9.7.1 Pengertian Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengolalaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis

berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [30].

2.9.8 UML

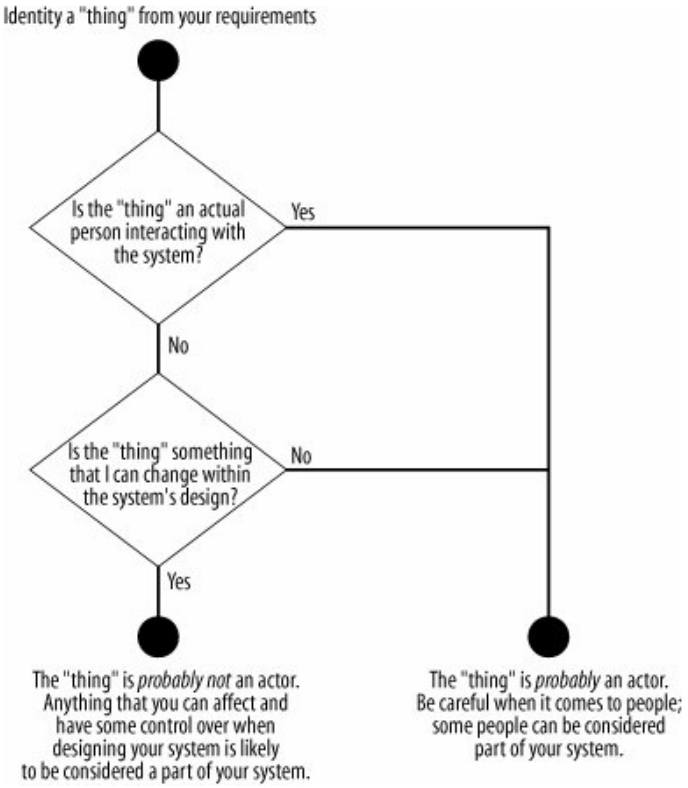
Pada bagian ini akan dijelaskan tentang pengertian UML dan diagram UML.

2.9.8.1 Pengertian UML

Unified Modeling Language (UML) adalah bahasa pemodelan standar untuk perangkat lunak dan sistem pengembang [21].

2.9.8.2 Tricky Actors

Tidak semua aktor jelas sistem eksternal atau orang – orang yang berinteraksi dengan sistem anda. berikut adalah beberapa pertanyaan untuk bertanya ketika mencoba untuk mengidentifikasi aktor.



Gambar 2.11 Mengidentifikasi Aktor

2.9.8.3 Diagram Use Case

Sebuah *use case* adalah sebuah kasus (atau situasi) dimana sistem digunakan untuk memenuhi satu atau lebih dari kebutuhan pengguna. Sebuah *use*

case menangkap sebagian fungsionalitas yang disediakan sistem [21]. Sebuah *use case* merupakan titik awal yang sangat baik untuk hampir semua aspek pengembangan berorientasi objek, desain, pengujian, dan dokumentasi. *Use case* menggambarkan persyaratan sistem yang ketat, juga menentukan nilai yang sistem sampaikan kepada pengguna. *Use case* hanya menentukan apa yang seharusnya dilakukan oleh sistem, yaitu kebutuhan fungsional sistem. *Use* case tidak menentukan apa yang tidak seharusnya dilakukan, seperti kebutuhan non fungsional sistem.

a. *Actor*/ Aktor

Aktor didefinisikan sebagai sesuatu yang berinteraksi dengan sistem dan bukan merupakan bagian dari sistem. Aktor tidak harus orang yang sebenarnya, sementara seorang aktor mungkin seseorang, bisa juga pihak ketiga sistem. Seorang aktor digambarkan dalam notasi UML baik menggunakan “stick man” atau kotak stereotip dan diberi label dengan nama yang sesuai. Pendekatan terbaik dalam penamaan aktor adalah dengan menggunakan nama yang dapat dipahami oleh pelanggan anda dan desainer sistem anda.



Gambar 2.12 Aktor dalam UML

b. Use Cases

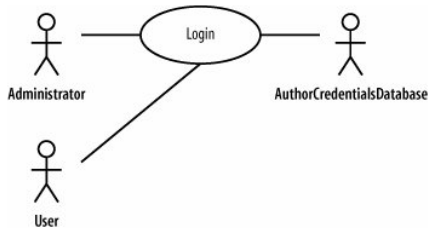
Sebuah *use case* dapat diidentifikasi dari kebutuhan pengguna. Hal terpenting untuk diingat adalah bahwa *use case* dari sudut pandang pengguna selengkapnya digunakan pada sistem. Sebuah *use case* di UML digambarkan dalam bentuk oval dengan nama yang menggambarkan interaksi yang mewakili. Sebuah *use case* adalah sesuatu yang menyediakan beberapa hasil terukur kepada pengguna atau sistem eksternal.



Gambar 2.13 *Use Case* dalam UML

c. *Communication Lines*/ Garis Komunikasi

Communication lines menghubungkan antara aktor dan use case untuk menampilkan bahwa aktor berpartisipasi di dalam use case. Ada potensi untuk memiliki beberapa pelaku yang terlibat dalam sebuah use case. Tidak ada batasan teoritis tentang jumlah pelaku yang dapat berpartisipasi dalam use case. Tujuan dari *communication lines* adalah untuk menunjukan bahwa seorang aktor hanya terlibat dalam *use case*, tidak berarti pertukaran informasi dalam arah tertentu atau aktor memulai *use case*.



Gambar 2.14 *Communication Lines* dalam UML

d. *Use Case Descriptive*/ Deskripsi Use Case

Deskripsi *use case* digunakan untuk memberikan detail informasi tentang siapa aktor dan langkah-langkah apa saja yang terlibat dalam *use case*.

Tabel 2.2 Beberapa jenis informasi yang dapat dimasukkan dalam deskrispi use case

Deskripsi *Use Case*	Keterangan
Related Requirements/ Kebutuhan yang terkait	Beberapa indikasi yang sebagian atau seluruhnya dipenuhi oleh use case.
Goal in Context/ Tujuan	Tempat use case didalam sistem dan mengapa use case ini penting.
Precondition/ Prakondisi	Kondisi sebelum use case diekskusi.
Successful end condition/ kondisi akhir yang sukses	Bagaimana kondisi sistem jika use case dijalankan dengan sukses
Failed end condition/ kondisi akhir yang gagal	Bagaimana kondisi sistem jika use case dijalankan dengan gagal
Primary actor/ aktor utama	Para aktor utama yang berpartisipasi dalam use case

Tabel 2.2 Beberapa jenis informasi yang dapat dimasukkan dalam deskrispi use case (lanjutan)

Deskripsi *Use Case*	Keterangan
Secondary actor/ aktor sekunder	Aktor yang berpartisipasi tetapi bukan aktor utama dalam use case
Trigger	Pemicu oleh Aktor yang menyebabkan use case dieksekusi
Main Flow/ arus utama	Tempat untuk menggambarkan setiap langkah penting dalam eksekusi normal use case
Extensions/ ekstensi	Penjelasan tentang langkah-langkah alternatif yang dijelaskan dalam *Main flow.*

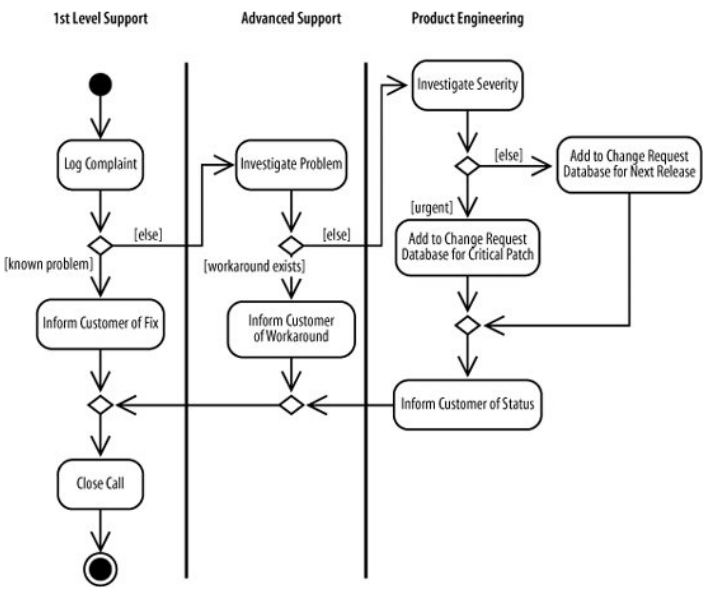
Contoh Deskripsi *Use Case* adalah sebagai berikut :

Use case name	**Create a new Blog Account**	
Related Requirements	Requirement A.1.	
Goal In Context	A new or existing author requests a new blog account from the Administrator.	
Preconditions	The system is limited to recognized authors and so the author needs to have appropriate proof of identity.	
Successful End Condition	A new blog account is created for the author.	
Failed End Condition	The application for a new blog account is rejected.	
Primary Actors	Administrator.	
	Secondary Actors	Author Credentials Database.
	Trigger	The Administrator asks the CMS to create a new blog account.
Main Flow	Step	Action
	1	The Administrator asks the system to create a new blog account.
	2	The Administrator selects an account type.
	3	The Administrator enters the author's details.
	4	The author's details are verified using the Author Credentials Database.
	5	The new blog account is created.
	6	A summary of the new blog account's details are emailed to the author.
Extensions	Step	Branching Action
	4.1	The Author Credentials Database does not verify the author's details.
	4.2	The author's new blog account application is rejected.

Gambar 2.15 Deskripsi *Use Case*

2.9.8.4 Diagram Aktifitas

Diagram aktivitas memungkinkan untuk menentukan bagaimana sistem akan mencapai tujuannya. Diagram aktivitas menunjukkan tindakan tingkat tinggi berantai bersama-sama untuk mewakili suatu proses yang terjadi pada sistem. Diagram aktivitas adalah salah satu diagram UML yang paling mudah karena menggunakan simbol-simbol yang mirip dengan notasi flowchart. Oleh karena itu, diagram aktifitas berguna untuk menggambarkan proses untuk khalayak luas.

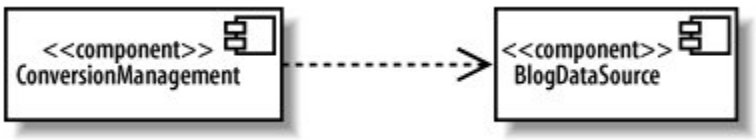


Gambar 2.16 Contoh Diagram Aktivitas

2.9.8.5 Diagram Komponen

Diagram komponen berfungsi untuk mengatur sistem menjadi dapat dikelola dan dapat digunakan kembali. Sebuah diagram komponen merupakan bagian enkapsulasi, dapat digunakan kembali, dan bisa mengganti bagian dari perangkat lunak. Oleh karena itu, diagram komponen dapat diukur dari yang berukuran relatif kecil, sekitar ukuran kelas, sampai subsistem yang lebih besar.

Kandidat yang baik untuk diagram komponen adalah item-item yang menampilkan sebuah fungsi kunci dan akan sering digunakan di seluruh sistem. Perangkat lunak, seperti *loggers*, XML, *parsers*, atau *online shopping charts*, adalah komponen yang mungkin sudah digunakan. Sebuah diagram komponen digambarkan dengan bentuk persegi panjang dengan stereotype **<<component>>** dan tab ikon persegi panjang di sudut sebelah kanan atas.



Gambar 2.17 Contoh *Diagram* Komponen

2.9.8.6 Diagram Deployment

Diagram deployment dalam UML menunjukkan tampilan fisik (*the physical view*) dari sistem, membawa perangkat lunak ke dunia nyata dengan menunjukkan bagaimana perangkat lunak akan ditugaskan untuk hardware dan bagaimana bagain-bagian tersebut berkomunikasi. Diagram deployment berfokus pada tampilan fisik (*the physical view*) dari sistem. Tampilan fisik (*the physical view*) bersangkutan dengan unsur-unsur fisik dari sistem, seperti *executable software files* dan hardware yang dijalankan.

Diagram deployment harus berisi rincian tentang sistem yang penting bagi pengguna. Jika penting untuk menunjukkan *hardware*, *firmware*, sistem operasi, *runtime environments*, atau bahkan perangkat *driver* dari sistem, maka seharusnya dimasukkan kedalam diagram deployment. Notasi diagram deployment dapat digunakan untuk model dari semua jenis hal. Jika ada fitur dari sistem yang tidak penting, maka itu tidak layak dimasukkan kedalam diagram deployment karena dapat dengan mudah mengalihkan perhatian dari fitur-fitur desain yang penting.

a. Nodes

Sebuah node merupakan sumber daya perangkat keras atau perangkat lunak yang dapat meng-host perangkat lunak atau file terkait. Node perangkat lunak dapat dianggap sebagai konteks aplikasi; umumnya bukan bagian dari perangkat lunak yang dikembangkan, tapi lingkungan pihak ketiga yang menyediakan layanan untuk perangkat lunak. Contoh node hardware : server, dekstop pc, disk drive. Contoh node execution environment : sistem operasi, *J2EE container*, *web server*, aplikasi server. Sebuah node digambar dengan bentuk kubus dan didalamnya ditulis berdasarkan jenisnya.



Gambar 2.18 Contoh *Hardware nodes*

b. Artefak

Artefak adalah file fisik yang melaksanakan atau digunakan oleh perangkat lunak. Contoh artefak pada umumnya seperti : file executable (.exe atau file jar),

file library (DLL. Atau file jar), sumber file (.java atau file cpp), file konfigurasi saat runtime (.xml, properti, atau .txt). Artefak ditampilkan dengan bentuk persegi panjang dengan stereotip <<artefak>>, atau ikon dokumen di sudut kanan atas, atau keduanya. Artefak akan ditampilkan dengan kedua stereotip <<artefak>> dan ikon dokumen.



Gambar 2.19 Contoh *artefak*

c. Komunikasi antar Node

Jalur komunikasi digunakan untuk menunjukkan bahwa node berkomunikasi satu sama lain pada saat runtime. Sebuah jalur komunikasi digambar dengan bentuk garis utuh yang menghubungkan dua node. Jenis komunikasi ditunjukkan dengan menambahkan stereotip pada jalur.



Gambar 2.20 Contoh komunikasi antar node