

BAB II

LANDASAN TEORI

2.1 Perancangan

Perancangan adalah langkah pertama dalam fase pengembangan rekayasa produk atau sistem (Pressman, 2010). Perancangan adalah proses penerapan berbagai teknik dan prinsip yang bertujuan untuk mendefinisikan sebuah peralatan, suatu proses atau sistem secara detail yang membolehkan realisasi fisik.

Komponen perancangan secara umum sebagai berikut :

a. Perancangan Model

Analisis sistem dapat merancang model dari sistem informasi yang diusulkan dalam bentuk fisik dan model logika. Model logika dari sistem informasi lebih menjelaskan pada *user*, bagaimana nantinya fungsi-fungsi dari sistem informasi secara logika akan bekerja. Model logika akan digambarkan dengan menggunakan *sequence diagram*.

b. Perancangan Keluaran

Keluaran merupakan produk dari sistem informasi yang dapat dilihat yang berupa tampilan di media atau layar komputer.

c. Perancangan Masukan

Alat masukan dapat dikategorikan ke dalam 2 (dua) golongan yaitu alat *input* langsung dan alat *input* tidak langsung. Alat input langsung berupa alat yang langsung dihubungkan ke CPU (*Central Processing Unit*) sedangkan alat *input* tidak langsung adalah alat yang tidak langsung dihubungkan ke CPU.

2.2 Aplikasi

Menurut Nazrudin Safaat H (2012) Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media. Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau suite aplikasi (*application suite*). Contohnya adalah *Microsoft Office* dan *Open Office*, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan setiap aplikasi. Sering kali, aplikasi ini memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

2.3 Data Dan Informasi

Keterkaitan data dan informasi sangatlah erat sebagaimana antara sebab dan akibat. Data merupakan dasar dari sebuah informasi, sedangkan informasi merupakan elemen yang dihasilkan dari suatu bentuk pengolahan data.

2.3.1 Data

Data merupakan keterangan-keterangan atau fakta-fakta yang dikumpulkan dari suatu populasi atau bagian populasi yang akan digunakan untuk menerangkan ciri-ciri populasi yang bersangkutan (Lungan, 2006).

Data merupakan keterangan-keterangan tentang suatu hal, dapat berupa sesuatu yang diketahui atau dianggap (Hasan, 2009).

dari pendapat para ahli tersebut, dapat disimpulkan bahwa data merupakan fakta atau keterangan yang dikumpulkan dari suatu populasi untuk menjelaskan karakteristik populasi tersebut.

Agar dapat menerangkan ciri-ciri populasi dengan benar, maka menurut lungan, data tersebut harus memiliki kriteria sebagai berikut :

1. Objektif, yaitu data yang benar-benar sama dengan keadaan yang sebenarnya.
2. Mewakili populasi
3. Galat baku(standard error) kecil.
4. Tepat waktu
5. Relavan

2.3.2 Informasi

Menurut Tata Sutabri, informasi adalah data yang telah diklasifikasikan atau diolah untuk digunakan dalam proses pengambilan keputusan(Sutabri, 2005).

Informasi juga merupakan fakta-fakta atau data yang diproses sedemikian atau mengalami proses transformasi data sehingga berubah bentuk menjadi informasi(Sulindawati & Fathoni, 2010).

2.4 Sejarah Toyota

Pendiri Toyota Motor Corporation adalah Kichiro Toyoda, pada tahun 1937 Toyota Motor berlokasi Toyota *city*, Aichi, Jepang. Prodak kendaraannya meliputi mobil penumpang, Truk, dan Bis. Pada tahun 1971 Toyota meresmikan PT.Toyota Astra Motor di Indonesia sebagai importir kendaraan sampai saat ini. Pada tahun 1999 Toyota tersebar di 160 negara di seluruh dunia, dengan dealer-dealer yang menyediakan penjualan dan service kendaraan Toyota. Seiring berjalannya perkembangan zaman, Toyota semakin berkembang dan merupakan salah satu pembuat kendaraan terbesar di dunia. (Team 21, 2005)

2.5 Toyota Avanza

Toyota Avanza pertama kali dikenalkan di publik Indonesia pada tanggal 11 Desember 2003 pada event Gaikindo Auto Expo. Lahir dari pabrikan PT. Astra Daihatsu Motor (ADM) Indonesia, yang berlokasi di Sunter, Jakarta Utara. Selain di Indonesia, Avanza juga diproduksi di Malaysia oleh Ferodua Manufacturing Sdn. dan di Tiongkok oleh FAW Jilin. Mobil ini memiliki saudara kembar yaitu Daihatsu Xenia. Mobil berjenis MPV (Multi Purpose Vehicle) atau kendaraan dengan segala kegunaan dengan kapasitas penumpang 7 orang mobil ini sangat cocok untuk keluarga Indonesia. Toyota Avanza diterima baik di pasaran Indonesia yaitu dengan ditandai terus berjalannya proses produksi dari tahun 2003 hingga sekarang ini serta jumlah penjualan yang terus meningkat.

Pada tahun 2014 pertarungan di kelas low MPV mendapat pendatang baru yang cukup tangguh, yaitu dari Honda yang sudah dikenal sebagai produsen mobil-mobil mewah. Honda memperkenalkan low MPV-nya dengan tampilan eksterior yang sangat eye-catching, yaitu Mobilio. Design eksterior mobil-mobil Honda memang harus diakui setingkat di atas mobil-mobil Jepang yang lainnya, munculnya Mobilio langsung merampas takhta Ertiga di peringkat ke-dua di tahun pertama debutnya. Toyota lantas merombak Avanza dengan melakukan facelift hingga menjadikannya bertabur fitur dan peningkatan yang signifikan. Diantaranya perombakan engine double VVT-I, *Bayond interior*, *Bayond safety*, dll.



Gambar 2.1 Toyota Avanza

2.6 UML (Unified Modelling Language)

UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

Unified Modelling Language (UML) adalah sebuah “bahasa” yang telah menjadi standar untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML, kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras komputer, sistem operasi dan jaringan apapun (Rosa, 2014).

UML terdiri dari 13 macam diagram yang dikelompokkan dalam kategori, yaitu :

1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

Struktur diagram terdiri dari :

- a. *Class diagram*, menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem serta interaksi antar kelas-kelas tersebut.
- b. *Object diagram*, menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem atau *instances* objek yang sebenarnya, yaitu nilai terkini dari *attribute instance*.
- c. *Component diagram*, menunjukkan organisasi dan ketergantungan di antara kumpulan komponen dalam sebuah sistem, menunjukkan model secara fisik atau komponen perangkat lunak pada sistem dan hubungan antar mereka.
- d. *Composite structure diagram*, digunakan untuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) dari *instance* yang saling terhubung.
- e. *Package diagram*, menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram *UML*.

- f. *Deployment diagram*, menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi dan menampilkan rancangan fisik jaringan tempat berbagai komponen akan diletakan.
- 2. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. *Behavior diagrams* terdiri dari :
 - a. *Use case diagram*, pemodelan untuk kelakuan (*behavior*) sistem atau perangkat lunak yang akan dibuat serta menggambarkan interaksi antara *use case* dan aktor dalam sistem tersebut.
 - b. *Activity diagram*, menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.
 - c. *State machine diagram*, menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem.
- 3. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. *Interaction diagrams* terdiri dari:
 - a. *Sequence diagram*, menggambarkan interaksi antar objek/bagian dalam bentuk urutan pengiriman pesan.
 - b. *Timing diagram*, menggambarkan tingkah laku sistem dalam periode waktu tertentu.
 - c. *Interaction overview diagram*, bentuk aktivitas diagram yang setiap titik mempresentasikan diagram interaksi.

Dari ketiga belas macam diagram diatas, hanya empat macam diagram yang akan penulis gunakan pada tahap analisis dan perancangan sistem, yaitu *use case diagram*, *activity diagram*, *sequence diagram*, dan *class diagram* (Rosa,2014).

2.6.1 Use Case Diagram

Use Case adalah abstraksi dari interaksi antara sistem dan aktor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara *user* sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem digunakan. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana

sistem akan terlihat di mata *user*. Sedangkan *use case diagram* memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan klien (Rosa, 2014).

Use case diagram memiliki beberapa komponen didalamnya, diantaranya :

1. *Actor*

Actor adalah sesuatu yang berhubungan dengan sistem dan berpartisipasi dalam *use case*. *Actor* menggambarkan orang, sistem atau *entitas eksternal* yang secara khusus membangkitkan sistem dengan input atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. *Actor* dilukiskan dengan peran yang mereka mainkan dalam *use case*, seperti Admin, Pengajar, Staf, Peserta didik, dan lain-lain.

2. *Use Case*

Use case adalah gambaran *fungsi* dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

3. *Relationship*

Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case diagram*. Relasi antara aktor dan *use case* disebut juga dengan asosiasi (*assosiation*). Asosiasi ini digunakan untuk menggambarkan bagaimana hubungan antara keduanya.

Relasi antara *use case* dengan *use case* :


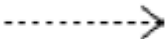

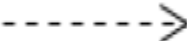
1. *Include*, pemanggilan *use case* oleh *use case* lain atau untuk menggambarkan suatu *use case* termasuk di dalam *use case* lain (diharuskam). Contohnya adalah pemanggilan sebuah fungsi program. *Include* digambarkan dengan garis lurus berpanah dengan tulisan <<include>>.
2. *Extend*, digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak kontrol form dan mendeklarasikan *ekstension* pada *use case* utama. Atau dengan kata lain adalah perluasan dari *use case* lain jika syarat atau kondisi



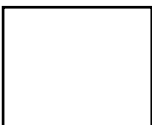



terpenuhi. *Extend* digambarkan dengan garis berpanah dengan tulisan *<<extend>>*.

3. *Generalization/Inheritance*, dibuat ketika ada sebuah kejadian yang lain sendiri atau perlakuan khusus dan merupakan pola berhubungan *base-parent use case*. *Generalization/Inheritance* digambarkan dengan garis berpanah tertutup dari *base use case* ke *parent use case*.

Berikut ini digambarkan dalam tabel simbol-simbol yang digunakan dalam *Use Case Diagram* :

Tabel 2.1 Simbol-Simbol Use Case Diagram

No.	Gambar	Nama	Keterangan
1.		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2.		<i>Dependency</i>	Hubungan di mana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
3.		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
4.		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .

5.		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6.		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu <i>actor</i> .
9.		<i>Collaboration</i>	Interaksi aturan - aturan dan elemen lain yang bekerjasama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10.		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

2.6.2 Class Diagram

Class Diagram merupakan bagian yang paling penting dalam analisa dan perancangan berorientasi objek. Dalam *UML* diagram kelas digunakan untuk memodelkan *static structure* dari sistem informasi (Rossa, 2014).

Diagram kelas menggambarkan struktur objek sistem, dimana diperlihatkan hubungan antar mereka. Diagram kelas merupakan fondasi untuk *component diagram* dan *deployment diagram*.

Secara garis besar terdapat 3 jenis *class*. Ketiga jenis *class* tersebut dikelompokkan berdasarkan fungsi dan karakteristiknya masing-masing, yaitu :

1. *Entity Class Diagram*

Merupakan paket utama dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk sistem dan menjadi landasan untuk menyusun basis data pada model data konseptual.




2. *Control Class Diagram*



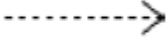


Berisi kumpulan *class* yang menjadi kontrol program termasuk koneksi dengan basis data dan merupakan kelas perantara atau penghubung antara *entity class* dengan kelas antar muka pemakai (*interface*).

3. *Boundary Class Diagram*

Berisi kumpulan *class* yang menjadi *interface* antara pemakai dengan sistem, seperti tampilan form untuk pencetakan

Tabel 2.2 Simbol-Simbol Class Diagram

No.	Gambar	Nama	Keterangan
1.		<i>Class</i>	Kelas pada struktur sistem.
2.		<i>Nary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 Objek
3.		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).


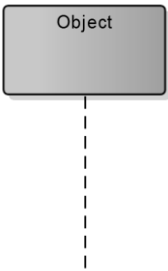

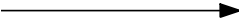
4.		<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5.		<i>Realization</i>	Relasi antar kelas dengan makna generalisasi spesialisasi (umum khusus)
6.		Dependency	Hubungan di mana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
7.		<i>Association</i>	Relasi antar kelas dengan makna umum, asosiasi biasanya disertai dengan multiplicity
8.		<i>Use Case</i>	Deskripsi dari urutan aksi aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor.

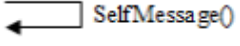
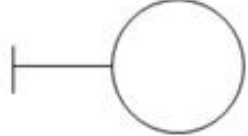


2.6.3 Sequence Diagram

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat *diagram sequence* juga dibutuhkan untuk melihat skenario yang ada pada *use case*.

Banyaknya *diagram sequence* yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada *diagram sequence* sehingga semakin banyak *use case* yang didefinisikan maka *diagram sequence* yang harus dibuat juga semakin banyak (Shalahuddin, M; S, Rosa A., 2016)

Tabel 2.3 Simbol-Simbol Sequence Diagram

No.	Simbol	Nama	Keterangan
1.		<i>Actor</i>	Digunakan untuk menggambarkan <i>user</i> / pengguna.
2.		<i>ObjectLife Line</i>	Menunjukkan keberadaan dari sebuah objek terhadap waktu. Yaitu objek dibuat atau dihilangkan selama suatu periode waktu diagram ditampilkan, kemudian <i>lifeline</i> berhenti atau mulai pada titik yang tepat.
3.		<i>LifeLine Control Class</i>	Objek entity, antarmuka yang saling berinteraksi.
4.		<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.

5.		<i>SelfMessage</i>	Sebuah objek yang mempunyai sebuah operation kepada dirinya sendiri.
6.		<i>Boundary</i>	Digunakan untuk menggambarkan sebuah form.
7.		<i>Control Class</i>	Digunakan untuk menghubungkan boundary dengan tabel.
8.		<i>Entity Class</i>	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan.

2.6.4 Activity Diagram


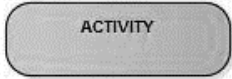

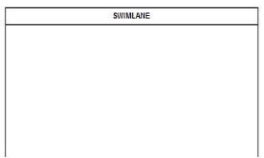

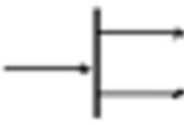
Activity Diagram menurut (Shalahuddin, M; S, Rosa A., 2016) menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau sebuah proses bisnis atau *menu* yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa *diagram* aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi aktivitas yang dapat dilakukan oleh sistem.

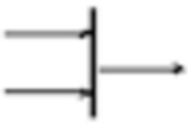

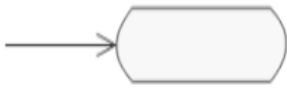
Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut (Shalahuddin, M; S, Rosa A., 2016) :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan tampilan dari sistem atau *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan *menu* yang ditampilkan pada perangkat lunak.

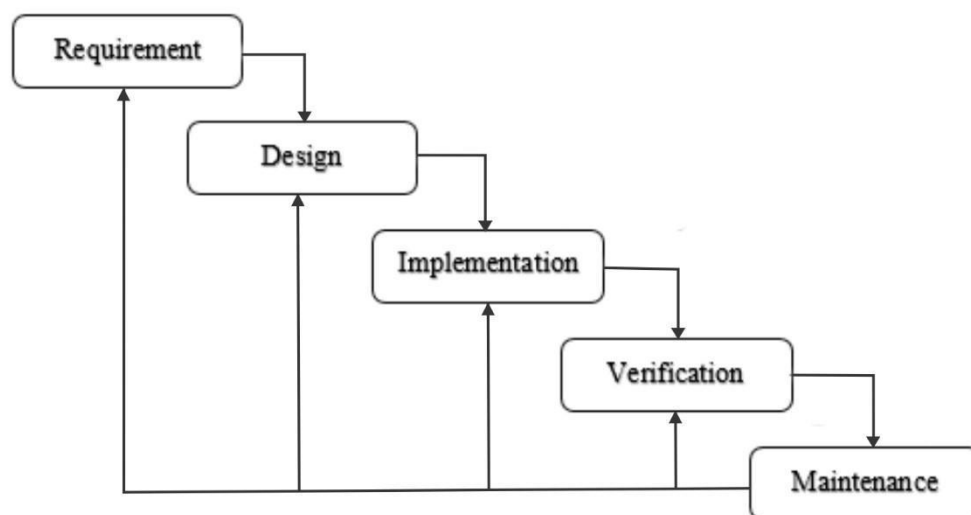
Tabel 2.4 Simbol-Simbol Activity Diagram

No.	Gambar	Nama	Keterangan
1.		<i>Initial Node</i>	Menggambarkan awal dari suatu aktivitas yang berjalan pada system
2.		<i>Activity</i>	Menggambarkan aktivitas yang dilakukan pada system
3.		<i>Decision</i>	Menggambarkan kondisi dari sebuah aktifitas yang bernilai benar / salah.
4.		<i>Swimlane</i>	Menggambarkan pembagian pengelompokkan berdasarkan tugas dan fungsi tersendiri.
5.		<i>Activity Final Node</i>	Menggambarkan akhir dari suatu aktivitas yang berjalan pada system
6.		<i>Fork</i>	Simbol <i>Fork</i> menunjukkan adanya percabangan secara paralel dari aktivitas

7.		<i>Join</i>	Simbol <i>Join</i> menunjukkan adanya penggabungan aktivitas
8.		<i>Miracle Activities</i>	<i>Miracle Activities</i> , tidak ada masukan dan ada keluaran dan dipakai pada waktu start point.
9.		<i>BlackHole Activities</i>	<i>Black Hole Activities</i> , ada masukan dan tidak keluaran

2.7 Metode Waterfall

Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ke tahap analisis, desain, coding, testing / verification, dan maintenance. Disebut dengan waterfall karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Sebagai contoh tahap desain harus menunggu selesainya tahap sebelumnya yaitu tahap requirement. Secara umum tahapan pada model waterfall dapat dilihat pada gambar berikut.



Gambar 2.2 Tahapan Model Waterfall

Pada gambar di atas adalah tahapan umum dari model proses ini. Akan tetapi Pressman (2008) memecah model ini menjadi 6 tahapan meskipun secara garis besar sama dengan tahapan-tahapan model waterfall pada umumnya. Berikut adalah penjelasan dari tahapan - tahapan yang dilakukan di dalam model ini menurut Pressman:

a. *Requirement Analysis*

Tahap ini pengembang sistem diperlukan komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diperoleh melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengguna.

b. *System Design*

Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini dan desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras(*hardware*) dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem secara keseluruhan.

c. *Implementation*

Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya.

Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing.

d. *Integration & Testing*

Seluruh unit yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing unit. Setelah integrasi seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.

e. *Operation & Maintenance*

Tahap akhir dalam model waterfall. Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya.

Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

Kelebihan menggunakan metode air terjun (waterfall) adalah metode ini memungkinkan untuk departementalisasi dan kontrol. proses pengembangan model fase one by one, sehingga meminimalis kesalahan yang mungkin akan terjadi. Pengembangan bergerak dari konsep, yaitu melalui desain, implementasi, pengujian, instalasi, penyelesaian masalah, dan berakhir di operasi dan pemeliharaan.

Kekurangan menggunakan metode waterfall adalah metode ini tidak memungkinkan untuk banyak revisi jika terjadi kesalahan dalam prosesnya. Karena setelah aplikasi ini dalam tahap pengujian, sulit untuk kembali lagi dan mengubah sesuatu yang tidak terdokumentasi dengan baik dalam tahap konsep sebelumnya.

2.8 Aplikasi Pendukung

Adapun aplikasi yang mendukung dalam pembuatan aplikasi pembelajaran bahasa inggris pada penelitian ini, diantaranya :

2.8.1 Pengertian Android

Android merupakan sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti *Smartphone*. Android awalnya dikembangkan oleh Android Inc, dengan dukungan finansial dari *Google*, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya *Open Handset Alliance*, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel Android pertama dijual pada bulan Oktober 2008. Menurut Nazaruddin (2012:1), Android merupakan sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Android umum digunakan di *Smartphone* dan juga tablet PC. Fungsinya sama seperti sistem operasi Symbian di Nokia, iOS di Apple

dan Blackberry OS. Menurut *Gramlich* (2013:8) Android dibangun menggunakan Linux. Lebih detailnya, Android merupakan mesin *Virtual* buatan yang telah dirancang untuk mengoptimasi sumber *memory* dan perangkat keras dalam sebuah perangkat mobilisasi. Android akan menjadi *source* terbuka yang memungkinkan berkembang diluar perusahaan sejalan dengan komunitas pembuat sejalan dengan perkembangan inovasi aplikasi mobilisasi. Menurut Yuniar Supardi (2014:2) menyatakan bahwa Android merupakan sebuah sistem informasi untuk perangkat *mobile* berbasis Linux yang mencakup sistem operasi, *middleware* dan aplikasi. Menurut Yeremias Edward (2012:1) mendefinisikan bahwa Android adalah sistem operasi untuk telepon seluler yang berbasis Linux.

Berdasarkan pengertian diatas, dapat ditarik kesimpulan bahwa Android merupakan sebuah sistem operasi untuk *smartphone* berbasis kernel Linux yang merupakan *platform* terbuka sehingga pengembang dapat dengan bebas mengembangkan aplikasi.

Seperti teknologi lainnya, Android muncul tidak langsung canggih seperti saat ini. Teknologinya yang bersifat *open source*, terus berkembang dan selalu terbuka untuk digunakan dan dikembangkan kapan saja. Mungkin inilah yang membuat Android begitu diminati.

2.8.2 Sejarah Android

Menurut (H, Nazruddin Safaat, 2015) *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. Awalnya *google inc* membeli *android inc* yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel atau *smartphone*. Kemudian untuk mengembangkan *android*, dibutuhkan *open handset alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk *Google*, *Intel*, *Motorola*, *Qualcomm*, *T-mobile*, dan *Nvidia*.

Pada saat perilisan perdana *android*, 5 November 2007, *android* bersama *open handset alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Dilain pihak, *google* merilis kode-kode *android* di bawah

lisensi *apache*, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler.

Di dunia ini terdapat 2 jenis sistem yaitu, sistem operasi distributor dan sistem operasi *android*. Pertama yang mendapat dukungan penuh dari *google* atau *google mail services (GMS)* dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung *google* atau dikenal sebagai *open handset distribution (OHD)*.

Sekitar September 2007 *google* mengenalkan *nexus one*, salah satu jenis *smartphone* yang menggunakan *android* sebagai sistem operasinya. Telepon seluler ini diproduksi oleh *HTC corporation* dan tersedia di pasaran pada 5 Januari 2008. Pada 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 operasi *open handset alliance*, OHA mengumumkan produk perdana merek *android*, perangkat *mobile* yang merupakan modifikasi kernel *linux 2.6*. sejak *android* dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan pembaruan fitur baru.

2.8.3 Generasi Android

Sekitar September 2007 sebuah studi melaporkan bahwa Google mengajukan hak paten aplikasi telepon seluler (akhirnya Google mengenalkan Nexus One, 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). Pada 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* (Apriliyandi, 2011).

Dibawah pimpinan Rubin Tim android pun mengembangkan Sistem Operasi untuk perangkat ponsel yang di kembangkan dari Kernel Linux, dan akhirnya munculah OS android yang berkembang di bawah lisensi GPL dan Apache (Apriliyandi, 2011).

Berikut merupakan Versi android generasi pertama sampai saat ini :

- b. Android Versi 1.1
- c. Android Versi 1.5 (Cupcake)
- d. Android Versi 1.6 (Donut)
- e. Android Versi 2.0/2.1 (Eclair)
- f. Android Versi 2.2 (Froyo : Frozen Yogurt)
- g. Android Versi 2.3 (Gingerbread)
- h. Android Versi 3.0/3.1 (Honeycomb)
- i. Android Versi 4.0 (ICS : Ice cream Sandwich)
- j. Android Versi 4.1 (Jelly Bean)
- k. Android Versi 4.4 KitKat
- l. Android Versi 5.0 Lollipop
- m. Android Versi 6.0 Marshmallow
- n. Android Versi 7.0 Nougat

Dari beberapa versi android adapun versi android yang masih digunakan pada *smartphone* android pada saat ini diantaranya :

- 1. Android Versi 1.1

Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email (Apriliyandi, 2011).



Gambar 2.3 Android Versi 1.1

2. Android Versi 1.5 (Cupcake)

Pada pertengahan Mei 2009, Google kembali merilis telepon seluler 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 pada layar yang dapat disesuaikan dengan sistem (Apriliyandi, 2011).



Gambar 2.4 Logo Android Cupcake

3. Android Versi 1.6 (Donut)

Donut (versi 1.6) dirilis 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 diintegrasikan; CDMA / EVDO, 802.1x, VPN, *Gestures*, dan *Text-to-speech engine*; kemampuan dial kontak; teknologi *text to change speech* (tidak tersedia pada semua ponsel; pengadaan resolusi VWGA) (Apriliyandi, 2011).



Gambar 2.5 Logo Android Donut

4. Android Versi 2.0/2.1 (Eclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), 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 (Apriliyandi, 2011).



Gambar 2.6 Logo Android Éclair

5. Android Versi 2.2 (Froyo : Frozen Yogurt)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, integrasi V8 JavaScript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan WiFi Hotspot portabel, dan kemampuan *auto update* dalam aplikasi Android Market (Apriliyandi, 2011).



Gambar 2.7 Logo Android Froyo

6. Android Versi 2.3 (Gingerbread)

Pada 6 Desember 2010, Android versi 2.3 (Gingerbread) diluncurkan. 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 (Apriliyandi, 2011).



Gambar 2.8 Logo Android Gingerbread

7. Android Versi 3.0/3.1 (Honeycomb)

Android 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 (Apriliyandi, 2011).



Gambar 2.9 Logo Android Honeycomb

8. Android Versi 4.0 (ICS : Ice cream Sandwich)

Diumumkan 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 email secara offline, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galxy Nexus (Apriliyandi, 2011).



Gambar 2.10 Logo Android Ice Cream Sandwich

9. Android Versi 4.1 (Jelly Bean)

Android Jelly Bean yang diluncurkan pada acara Google I/O lalu membawa sejumlah keunggulan dan fitur baru. Penambahan baru diantaranya meningkatkan *input* keyboard, desain baru fitur pencarian, UI yang baru dan pencarian melalui Voice Search yang lebih cepat. Tak ketinggalan Google Now juga menjadi bagian yang diperbarui. Google Now memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca, lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi Android Jelly Bean 4.1 muncul pertama kali dalam produk tablet Asus, yakni Google Nexus 7 (Apriliyandi, 2011).



Gambar 2.11 Logo Android Jelly Bean

10. Android Versi 4.4 KitKat

Android KitKat dirilis pada tanggal 31 Oktober 2013. Android KitKat disebutkan juga lebih bersahabat untuk *smartphone* ataupun *tablet* bertipe *low-end* karena diklaim mampu berjalan lancar dengan memory RAM 512 MB. Hal ini dapat tercapai karena peningkatan memory manajemen dan optimasi di kernel, sistem, *framework*, dan *aplikasi*.



Gambar 2.12 Logo Android Kitkat

11. Android Versi 5.0 Lollipop

Android Lollipop dirilis pada tanggal 15 Oktober 2014. Pada perilsan versi terbarunya, sistem operasi android versi 5.0 Loliipop dibekalkan dengan sector tampilan yang lebih berwarna dan responsive. Selain itu, pihak android juga memberikan jaminan pada para pengguna android jika navigasi dari sistem operasi 5.0 Lollipop lebih mudah dan tidak akan menyulitkan penggunaanya.



Gambar 2.13 Logo Android Lollipop

12. Android Versi 6.0 Marshmallow

Android Marshmallow memperkenalkan model izin yang didesain ulang: sekarang ada hanya delapan kategori izin, dan aplikasi yang tidak lagi secara otomatis diberikan semua hak akses mereka ditentukan pada waktu instalasi. Sebuah sistem opt-in sekarang digunakan, di mana pengguna akan diminta untuk memberikan atau menolak izin individu (seperti kemampuan untuk mengakses kamera atau mikrofon) untuk aplikasi ketika mereka dibutuhkan. Aplikasi mengingat hibah izin mereka, dan mereka dapat disesuaikan oleh pengguna setiap saat. Model izin baru akan digunakan hanya oleh aplikasi yang dikompilasi untuk Marshmallow menggunakan kit pengembangan perangkat lunak (SDK) tersebut, sementara semua aplikasi lainnya akan terus menggunakan model izin sebelumnya.

Marshmallow juga memiliki skema manajemen daya baru bernama *Doze* yang mengurangi tingkat aktivitas aplikasi latar belakang saat perangkat menentukan bahwa itu tidak sedang aktif ditangani oleh pengguna, yang, menurut Google, menggandakan pemakaian baterai perangkat. Hal ini juga memperkenalkan pilihan untuk mengatur ulang semua pengaturan jaringan, tersedia untuk pertama kalinya pada Android, yang membersihkan pengaturan terkait jaringan untuk Wi-Fi, Bluetooth dan koneksi seluler.

Android Marshmallow memberikan dukungan asli untuk pengenalan sidik jari, memungkinkan penggunaan sidik jari untuk membuka perangkat dan otentikasi Play Store dan pembelian Android Pay; API standar juga tersedia untuk melaksanakan otentikasi berbasis sidik jari dalam aplikasi lain. *Android* Marshmallow mendukung USB Type-C, termasuk kemampuan untuk menginstruksikan perangkat untuk mengisi daya perangkat lain melalui USB. Marshmallow juga memperkenalkan "pranala yang diverifikasi" yang dapat dikonfigurasi untuk membuka langsung dalam aplikasi tertentu mereka tanpa petunjuk pengguna lanjut. Versi API

Android yang disediakan oleh Marshmallow adalah 23. Alat pengembang Android Marshmallow tersedia di Pengelola SDK di bawah tingkat API "MNC".



Gambar 2.14 Logo Android Marshmallow

13. Android Nougat

Android Nougat atau android versi 7.0 adalah sistem operasi yang terbaru, memperkenalkan perubahan penting pada platform dan juga pengembangan, termasuk kemampuan untuk menampilkan beberapa aplikasi di layar sekaligus dalam tampilan layar yang terpisah, serta lingkungan berbasis "Java OpenJDK" dan dukungan pada render grafis "Vulkan API", dan pembaruan sistem mulus pada perangkat yang di dukung.



Gambar 2.15 Logo Android Nougat

2.8.4 Fitur Android

Fitur yang tersedia di Android adalah:

1. Kerangka aplikasi: itu memungkinkan penggunaan dan penghapusan komponen yang tersedia.
2. Dalvik mesin virtual: mesin virtual dioptimalkan untuk perangkat telepon seluler.
3. Grafik: grafik di 2D dan grafis 3D berdasarkan pustaka OpenGL.
4. SQLite: untuk penyimpanan data.
5. Mendukung media: audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
6. GSM, Bluetooth, EDGE, 3G, 4G dan WiFi (tergantung piranti keras)
7. Kamera, Global Positioning System (GPS), kompas, NFC dan *accelerometer* (Pratama, 2011).

2.8.5 Arsitektur Android

Secara garis besar arsitektur Android dapat dijelaskan sebagai berikut:

1. *Application dan Widget*

Application dan Widget ini adalah layer dimana kita berhubungan dengan aplikasi saja. Di layer terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis dengan menggunakan bahasa pemrograman JAVA.

2. *Application Framework*

Application Framework adalah layer untuk melakukan pengembangan / pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti content provider yang berupa SMS dan panggilan telepon.

Komponen-komponen yang termasuk di dalam Application Framework adalah sebagai berikut:

- a) *Views*
- b) *Content Provider*

- c) *Resource Manajer*
- d) *Notification Manajer*
- e) *Activity Manajer*

3. *Libraries*

Libraries adalah layer tempat fitur-fitur Android berada, biasanya para pengembang aplikasi mengakses *libraries* untuk menjalankan aplikasinya.

4. *Android Runtime*

Layer yang membuat aplikasi Android dapat dijalankan di mana dalam prosesnya menggunakan implementasi Linux. Dalvik *Virtual Machine* merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam *Android Runtime* dibagi menjadi dua bagian yaitu:

- a) *Core Libraries* Aplikasi Android dibangun dalam bahasa Java, sementara DVM bukan merupakan *virtual machine* untuk Java. Sehingga diperlukan *libraries* yang berfungsi untuk menterjemahkan bahasa Java/C yang ditangani oleh *Core Libraries*.
- b) *Dalvik Virtual Machine* *Virtual Machine* berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat Linux kernel untuk melakukan *threading* dan manajemen tingkat rendah.

5. *Linux Kernel*

Linux Kernel adalah *layer* dimana inti sistem operasi dari Android itu berada. Berisi file system yang mengatur system processing *memory*, *resource*, *drivers*, dan sistem-sistem operasi Android lainnya. Linux Kernel yang digunakan Android adalah Linux Kernel *release 2.6*. (Pratama, 2011)

2.8.6 Platform Android

1. *Android Lengkap (Complete Platform)*: para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan *platform android*. *Android* merupakan sistem

operasi yang aman dan banyak menyediakan *tools* dalam membangun *software* dan memungkinkan untuk peluang pengembangan aplikasi.

2. Terbuka (*Open Source Platform*): pengembang dengan bebas untuk membangun aplikasi. Aplikasi *android* sendiri menggunakan *linux* kernel 2.6.
3. Bebas (*Free Platform*): *android* adalah *platform* atau aplikasi yang bebas digunakan dan dikembangkan, tidak ada lisensi atau biaya *royalty* untuk dikembangkan pada *platform android* (H, Nazruddin Safaat, 2011).

Android merupakan generasi baru *platform mobile*, *platform* memberikan pengembang untuk melakukan pengembangan sesuai dengan yang diharapkannya. Sistem operasi yang mendasari *android* dilisensasikan bahan GNU. *General public* lisensi versi 2 (GPLv2), yang sering dikenal dengan istilah “*copyleft*” lisensi dimana setiap perbaikan pihak ketiga harus terus jatuh di bawah *terms*. *Android* didistribusikan di bawah lisensi *apache software* (ASL atau *apache2*), yang memungkinkan untuk distribusi kedua dan seterusnya. Komersialisasi pengembang (produsen *handset* khususnya) dapat memilih untuk meningkatkan *platform* tanpa harus memberikan perbaikan mereka ke masyarakat *open source*. Sebaliknya pengembang dapat keuntungan dari perangkat tambahan seperti perbaikan dan mendistribusikan ulang pekerjaan mereka di bawah lisensi apapun yang mereka inginkan. Pengembang aplikasi *android* diperbolehkan untuk mendistribusikan aplikasi mereka di bawah skema lisensi apapun yang mereka inginkan (H, Nazruddin Safaat, 2011).

2.8.7 Kelebihan Android

Ada beberapa kelebihan android diantaranya :

- a. Multitasking - keunggulan dari Symbian yang bisa membuka beberapa aplikasi sekaligus, begitu juga Android yang mampu

membuka beberapa aplikasi sekaligus tanpa harus menutup salah satunya.

- b. Kemudahan dalam notifikasi - Setiap ada SMS, Email, atau bahkan artikel terbaru dari RSS Reader, akan selalu ada notifikasi di Home Screen Ponsel Android, tak ketinggalan Lampu LED Indikator yang berkedip-kedip, sehingga tidak akan terlewatkan satu SMS, Email ataupun Misscall sekalipun.
- c. Akses mudah terhadap ribuan Aplikasi Android lewat Google Android App Market kalau gemar install aplikasi ataupun games, lewat Google Android App Market bisa mendownload berbagai aplikasi dengan gratis. Ada banyak ribuan aplikasi dan games yang siap untuk didownload di ponsel Android.
- d. Pilihan ponsel yang beraneka ragan – ponsel Android, akan terasa beda dibandingkan dengan iOS, jika iOS hanya terbatas pada iPhone dari Apple, maka Android tersedia di ponsel dari berbagai produsen, mulai dari Sony Ericsson, Motorola, HTC sampai Samsung. Dan setiap pabrik ponsel pun menghadirkan ponsel Android dengan gaya masing-masing, seperti Motorola dengan Motoblur-nya, Sony Ericsson dengan TimeScape-nya. Jadi anda bisa leluasa memilih ponsel Android sesuai dengan merk favorit.
- e. Bisa menginstal ROM yang dimodifikasi – Tidak puas dengan tampilan standar Android, ada banyak Custom ROM yang bisa dipakai untuk ponsel Android.
- f. Widget – Dengan adanya Widget di *homescreen*, bisa dengan mudah mengakses berbagai setting dengan cepat dan mudah.
- g. Google Maniak – kelebihan Android lainnya jika pengguna setia layanan Google mulai dari Gmail sampai Google Reader, ponsel Android telah terintegrasi dengan layanan Google, sehingga bisa dengan cepat mengecek email dari Gmail.

2.8.8 Kekurangan Android

Kekurangan dari Android adalah sebagai berikut:

- a. Koneksi internet yang terus menerus, kebanyakan ponsel berbasis sistem ini memerlukan koneksi internet yang simultan alias terus menerus aktif. Koneksi internet GPRS selalu aktif setiap waktu, itu artinya anda harus siap berlangganan paket GPRS yang sesuai dengan kebutuhan.
- b. Iklan – Aplikasi di Ponsel Android memang bisa didapatkan dengan mudah dan gratis, namun konsekuensinya disetiap Aplikasi tersebut, akan selalu menampilkan iklan yang terpampang, entah itu bagian atas atau bawah aplikasi.

2.8.9 Android SDK (Software Development Kit)

JDK singkatan dari *java development kit* adalah pengembangan perangkat aplikasi *java*, perangkat ini mutlak diperlukan untuk membuat aplikasi *android*, mengingat aplikasi itu berbasis *java*. Sebagaimana *java* adalah salah satu bahasa pemrograman yang biasa untuk membuat aplikasi (Kadir, Abdul, 2013).

2.8.10 Eclipse

Menurut Nasruddin Safaat h (Pemrograman aplikasi *mobile smartphone* dan tablet PC berbasis android 2015) *Eclipse* adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*). Berikut ini adalah sifat dari Eclipse:

1. *Multi-platform*: Target sistem operasi *Eclipse* adalah *Microsoft Windows, Linux, Solaris, AIX*, dan *Mac OS X*.
2. *Mult-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 untuk pengembangan aplikasi, *Eclipse* pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari *Eclipse* yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in*. *Eclipse* dibuat dari kerja sama antara perusahaan-perusahaan anggota 'Eclipse Foundation' (beserta individu-individu lain).

2.8.11 Java

Java merupakan pemrograman yang menanjak popularitasnya saat ini. Selain sifat grafis, Java mudah di dapatkan dan juga tangguh. Java sendiri diciptakan pada tahun 1991 yang diprakasai oleh tim Sun Microsystem melalui proyek bernama Green yang dipimpin oleh James Gosling.

Java merupakan fitur-fitur dari bahasa pemrograman yang lain, dimana fitur itu dianggap merupakan suatu kelebihan oleh tim Sun Microsystem. Misalnya Java Virtual Machine (JVM) atau Java Runtime Enviroment (JRE) yang merupakan mesin maya pada bahasa pascal, sintaks, dan Exception Handling diambil dari C/C++ dan lain sebagainya.

Java Virtual Machine (JVM) atau *Java Runtime Enviroment* (JRE) merupakan fitur java yang membuatnya dapat berjalan pada semua *platform* sistem operasi. Apabila anda membuatnya dapat pada Windows, anda dapat menjalankannya pada sistem operasi Linux, Mac OS dan lainnya.

Java memiliki beberapa keunggulan yang tidak dimiliki oleh bahasa pemrograman lain berdasarkan *White Paper* resmi dari Sun Microsystem, Java memiliki karakteristik-karakteristik sebagai berikut (Andi, 2011) :

a. Sederhana

Bahasa pemrograman Java menggunakan sintaks mirip dengan C++, namun sintaks Java telah banyak diperbaiki terutama menghilangkan penggunaan *pointer* yang rumit dan *multiple inheritance*. Java juga menggunakan *automatic memory allocation* dan *memory garbage collection*.

b. Berorientasi Objek

Java menggunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antara objek-objek tersebut.

c. Dapat Didistribusikan dengan Mudah

Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya *libraries networking* yang terintegrasi pada Java.

d. *Interpreter*

Program Java dijalankan menggunakan *intrepreter* yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *source code* Java yang telah dikompilasi menjadi *Java bytecodes* dapat dijalankan pada *platform* yang berbeda-beda.

e. *Robust*

Java mempunyai realibitas yang tinggi. *Compiler Java* mempunyai kemampuan untuk mendeteksi *error* secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai *Runtime Exception Handling* untuk membantu mengatasi *error* pada program.

f. Aman

Sebagai bahasa pemrograman untuk aplikasi *internet* dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.

g. *Architecture Neutural*

Program Java merupakan *platform* independen. Program cukup mempunyai satu buah versi yang dapat dijalankan pada *platform* yang berbeda dengan *Java Virtual Machine* (JVM).

h. *Portable*

Source code maupun program Java dapat dengan mudah dibawa ke *platform* yang berbeda-beda tanpa harus dikompilasi ulang.

i. *Multithreading*

Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.

j. *Dinamis*

Java di desain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu *class* dengan menambahkan *properties* ataupun *method* dapat dilakukan pada tanpa mengganggu program yang menggunakan *class* tersebut.

2.8.12 Enterprise Architect

Sistem informasi dan sistem komunikasi, dalam sebuah organisasi jika tidak di bangun berdasarkan desain atau rancangan yang jelas pada awal pembangunannya akan merusak keharmonisan dari sistem tersebut. Upaya untuk menghindari terjadinya gangguan pada keharmonisan sistem pada saat pengembangan sistem tersebut adalah dengan melakukan perencanaan dari sistem tersebut secara jelas sebelum sistem tersebut di bangun. Perencanaan sistem secara menyeluruh (melingkupi seluruh aspek dalam organisasi) inilah yang di kenal dengan istilah *Enterprise Architecture*.

Enterprise adalah gambaran bisnis pada suatu organisasi dalam bentuk yang kompleks, gambaran ini memiliki jangkauan yang sangat luas meliputi manusia (pelanggan, staff dan kontraktor), proses dan asset yang digunakan untuk mengembangkan dan menghasilkan produk-produk dan *service-service*, data dan informasi yang disimpan untuk digunakan dalam bisnis, dan mekanisme untuk menyediakan komunikasi dan sekuriti. (Nasruddin Safaat h, 2012).

2.8.13 Adobe Photoshop

Adobe Photoshop adalah *software* garis berbasis *bitmap (pixel)* yang biasa dipakai untuk mengedit foto, membuat ilustrasi bahkan desain *web*, sehingga

banyak digunakan di *studio* foto, percetakan, *production house*, biro arsitektur, pabrik tekstil dan bidang yang berkaitan dengan Teknologi Informasi (IT).

Namun *photoshop* tidak cocok untuk *me-layout* brosur, publikasi dengan halaman banyak, desain logo, spanduk ukuran besar, *image* 3D, animasi dan lain-lain. Untuk itu anda perlu menguasai *coreldraw*, *adobe illustrator*, *indesign*, *image ready*, *after effects* atau lebih baik lagi jika menguasai 3D *Studio Max*.

Software sejenis antara lain *paint for windows*, *gimp*, *corelpaint*, *photo impact*, *photostudio*, *photostyler*, *ACDSee*, *paintbrush* dan lain-lain. Namun semuanya kalah *popular* di bandingkan dengan *photoshop*, sehingga umumnya desainer profesional wajib menguasai *adobe photoshop* (Hendratman, Hendi, 2014).

2.8.14 Microsoft Office (Word)2013

Microsoft Office 2013 merupakan seri aplikasi terbaru dari *Microsoft*. Banyak perubahan yang ditampilkan oleh *Microsoft office* 2013, salah satunya aplikasi pengolahan kata yang lebih baik, terutama untuk perkantoran.

Microsoft office 2013 mampu mengintergrasikan secara mendalam antara aplikasi *desktop* dan data *online*. Artinya, anda dapat menggunakan dan mengakses *Microsoft* 2013 kapan saja dengan PC, *tablet*, dan ponsel *windows phone*. *Microsoft office* 2013 dapat membantu anda dalam membuat *dokumen word* dan tampilan baru pada *dokumen word*. Selain itu *microsoft office* 2013 memungkinkan anda menyimpan data *office* secara *online* dan dapat diakses kapan saja tanpa takut kehilangan data (Komputer, Wahana, 2013).

2.9 Pengujian Black box

Black Box Testing menurut (Shalahuddin, M; S, Rosa A., 2016) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi masukan dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan.

Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan

spesifikasi yang dibutuhkan. Kasus uji yang dibuat dengan kasus benar dan kasus salah, misalkan untuk kasus proses *login* maka kasus uji yang dibuat adalah :

1. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password* yang benar).
2. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password*) yang salah, misalnya nama pemakai benar tapi kata sandi salah atau sebaliknya atau keduanya.