BAB II
LANDASAN TEORI

II.1 Pengertian *Mobile*

Mobile berasal dari bahasa inggris yang artinya berpindah. *Mobile* dapat di artikan sebagai perpindahan dari suatu tempat ke tempat yang lain. Pada konsep ini, *mobile* lebih cenderung dengan aplikasi yang dapat digunakan kapanpun dan dimanapun dengan menggunakan perangkat *mobile* seperti telepon seluler, *pager*, PDA (*Portable Digital Assistant*), *smartphone* dan sejenisnya.

Aplikasi *mobile* berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju sedangkan *mobile* dapat di artikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

Maka aplikasi *mobile* dapat di artikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah-pindah dari satu tempat ke tempat yang lain serta mempunyai ukuran yang kecil. Aplikasi *mobile* ini dapat di akses melalui perangkat nirkabel, *pager*, PDA (*Portable Digital Assistant*), telepon seluler, *smartphone*, dan perangkat sejenisnya.

II.2 Pengertian Android

Android adalah sistem operasi yang digunakan di *smartphone* dan juga *tablet* PC. Fungsinya sama seperti sistem operasi *Symbian* di Nokia, iOS di *Apple* dan *BlackBerry* OS. Android tidak terikat ke satu merek *handphone* saja, beberapa vendor terkenal yang sudah memakai Android antara lain Samsung, Sony Ericsson, HTC, Nexus, Motorolla, dan lain-lain (Rizky Ari Nugroho; 2015)

II.2.1 Sejarah Android

Android adalah sistem operasi untuk telepon seluler yang berbasiskan Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri sehingga dapat digunakan oleh bermacam peranti penggerak. Awalnya Google Inc. membeli Android Inc. pendatang baru yang membuat *software* (perangkat lunak) untuk telepon genggam. Kemudian untuk mengembangkan Android di bentuklah *Open Handset Alliance* yang merupakan gabungan dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, TMobile, dan NVidia.

Pada saat perilisan perdana Android pada tanggal 5 november 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android dibawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Terdapat dua jenis distributor sistem operasi Android. Pertama yang dapat dukungan penuh dari Google atau *Google Mail Service*

(GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung dari Google atau dikenal sebagai *Open Handset Distribution* (DHD) (Firdan Ardiansyah; 2011).

II.2.2 Kelebihan Android

Setiap sistem operasi memiliki kelebihan dalam kinerja dan fitur-fiturnya, berikut adalah kelebihan sistem operasi android :

1. *User interface* menarik.
2. *Open Source*. Sumber bebas dan terbuka.
3. Multitasking. Bisa menjalankan berbagai aplikasi dalam satu waktu.
4. Kemudahan dalam notifikasi. Setiap ada SMS, Email, atau bahkan artikel terbaru dari RSS *Reader*, akan selalu ada notifikasi di *Home Screen* Ponsel Android.
5. Akses Mudah terhadap Ribuan Aplikasi Android lewat Google Android *App Market*.
6. Pilihan Ponsel yang beranekaragam. Android tersedia di ponsel dari berbagai produsen, mulai dari Sony Ericsson, Motorola, HTC sampai Samsung.
7. Bisa menginstal ROM yang dimodifikasi.
8. *Widget*.
9. Terintegrasi dengan google.
10. Aman dari *virus*. Karena menggunakan kernel dari linux.

II.3 EYD (Ejaan yang disempurnakan)

Berdasarkan etimologi, keta ejaan berasal dari kata dasar eja, yang berarti melafalkan huruf-huruf atau lambang bunyi bahasa. Ejaan adalah kaidah-kaidah cara menggambarkan bunyi-bunyi kata dan kalimat dalam bentuk tulisan (huruf-huruf) dan penggunaan tanda baca (KBBI, 1991:250). Berdasarkan pengertian di atas, maka dapat ditarik kesimpulan bahwa ejaan adalah tata cara penggunaan kata, kalimat dan tanda baca dalam bentuk lisan maupun tertulis. Ejaan yang disempurnakan atau yang lebih dikenal dengan singkatan EYD adalah ejaan yang mulai resmi dipakai dan digunakan di Indonesia tanggal 16 Agustus 1972. Ejaan ini masih digunakan hingga saat ini.

EYD adalah rangkaian aturan yang wajib digunakan dan ditaati dalam tulisan bahasa Indonesia resmi. EYD mencakup penggunaan dalam 12 hal, yaitu penggunaan huruf besar (kapital), tanda koma, tanda titik, tanda seru, tanda hubung, tanda titik koma, tanda tanya, tanda petik, tanda titik dua, tanda kurung, tanda *elips*, dan tanda garis miring.

Analisis kesalahan merupakan penelaahan, penilaian sesuatu yang salah atau menyimpang dari aturan. Kesalahan dalam penggunaan EYD merupakan kesalahan dalam penggunaan atau pemakaian bahasa yang sesuai dengan kaidah bahasa yang sudah ditentukan sejak tanggal 16 Agustus 1972. Kesalahan-kesalahan pada ejaan yang banyak dilakukan dalam menuliskan bahasa Indonesia yang baik dan benar memang merupakan kesalahan umum yang banyak terjadi atau pernah dilakukan oleh siapa saja terutama oleh para siswa. Kesalahan dalam penerapan kaidah EYD, diantaranya a) kesalahan penulisan huruf kapital, b)

kesalahan penulisan huruf miring, c) kesalahan penulisan lambang bilangan, d) kesalahan penulisan tanda baca.

II.4 Kamus Besar Bahasa Indonesia

Kamus Besar Bahasa Indonesia adalah kamus ekabahasa resmi bahasa Indonesia yang disusun oleh Badan Pengembangan dan Pembinaan Bahasa dan diterbitkan oleh Balai Pustaka. Kamus ini menjadi acuan tertinggi bahasa Indonesia yang baku, karena kamus ini merupakan kamus bahasa Indonesia terlengkap dan yang paling akurat yang pernah diterbitkan oleh penerbit yang memiliki hak paten dari pemerintah Republik Indonesia yang dinaungi oleh Kementerian Pendidikan dan Kebudayaan Indonesia (id.wikipedia.org).

Adapun sejarah Kamus Besar Bahasa Indonesia menurut situs resmi (id.wikipedia.org) sebagai berikut :

1. Edisi pertama (1988) : Edisi pertama adalah hasil pengembangan dari Kamus Bahasa Indonesia yang terbit pada tahun 1983. Kamus ini baru memuat 62.100 lema.
2. Edisi kedua (1991) : Edisi kedua adalah revisi pertama KBBI dan memuat 72.000 lema.
3. Edisi ketiga (2005) : Edisi ketiga memuat 78.000 lema. Menurut Dr. Dendy Sugono, Kepala Pusat Bahasa, kamus ketiga ini masih terasa banyak sekali kosakata yang belum masuk. Tetapi harap diingat bahwa KBBI adalah Kamus Umum berisi kosakata umum, sehingga dalam kamus tidak termasuk berbagai istilah. Untuk penggunaan kamus bidang ilmu tertentu Pusat Bahasa juga memiliki kamus Istilah.

4. Edisi keempat (2008) : Edisi keempat memuat lebih dari 90.000 lema. Pada edisi ini KBBI diperkaya kosakata yang berasal dari kamus istilah, pada edisi ini kamus disusun berdasarkan paradigma.
5. Edisi kelima : Edisi kelima kemungkinan besar akan dirilis pada tahun 2016, dengan perkiraan penambahan kata sekitar 2.000 kata

II.5 Eclipse

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

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak pengembangan IBM *Visual Age for Java* 4.0. Produk Eclipse ini diluncurkan oleh IBM pada tanggal 5 November 2001. IBM menginvestasikan US$ 40 juta untuk pengembangannya. Sejak 5 November 2001, konsorsium Eclipse *Foundation* mengambil alih pengembangan Eclipse lebih lanjut.

Sejak tahun 2006, Eclipse *Foundation* mengkoordinasikan peluncuran Eclipse secara rutin dan simultan yang dikenal dengan nama *Simultaneous Release*. Setiap versi peluncuran terdiri dari Eclipse *Platform* dan juga sejumlah proyek yang terlibat dalam proyek Eclipse. Tujuan sistem ini adalah untuk menyediakan distribusi Eclipse dengan fitur-fitur dan versi yang terstandarisasi. Hal ini juga dimaksudkan untuk mempermudah *deployment* dan *maintenance* untuk sistem *enterprise*, serta untuk kenyamanan. Peluncuran simultan dijadwalkan pada bulan Juni setiap tahunnya.

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. *Mulit-language*: Eclipse dikembangkan dengan bahasa pemrograman *Java*, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lain 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, pengujian perangkat lunak, pengembangan web, dan lain sebagainya.

Pada saat ini, Eclipse merupakan salah satu IDE favorit karena gratis dan *open source*. *Open source* 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 membuat komponen yang disebut plugin.

Gambar II.1 Emulator Android

II.6 Pemodelan UML

Unified Modelling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek.

Sejarah UML sendiri terbagi dalam dua *fase* sebelum dan sesudah munculnya UML. Dalam *fase* sebelum, UML sebenarnya sudah mulai diperkenalkan sejak tahun 1990an namun notasi yang dikembangkan oleh para ahli analisis dan desain berbeda-beda, sehingga dapat dikatakan belum memiliki *standarisasi.*

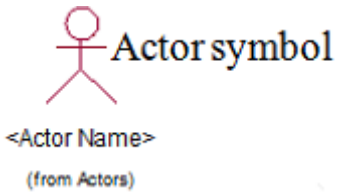
Fase kedua; dilandasi dengan pemikiran untuk mempersatukan metode tersebut dan dimotori oleh *Object Management Group* (OMG) maka pengembangan UML dimulai pada akhir tahun 1994 ketika Grady Booch dengan metode OOD (*Object-Oriented Design*), Jim Rumbaugh dengan metode OMT (*Object Modelling Technique*) mereka ini bekerja pada *Rasional Software Corporation* dan Ivar Jacobson dengan metode OOSE (*Object-Oriented Software Engineering*) yang bekerja pada perusahaan *Objectory Rasional* (Haviluddin; 2011).

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu:

1. *Use Case* Diagram untuk memodelkan proses bisnis.
2. *Conceptual* Diagram untuk memodelkan konsep-konsep yang ada di dalam aplikasi.
3. *Sequence* Diagram untuk memodelkan pengiriman pesan *(message*) antar objek.
4. *Collaboration* Diagram untuk memodelkan interaksi antar objek.
5. *State* Diagram untuk memodelkan perilaku *objects* di dalam sistem.
6. *Activity* Diagram untuk memodelkan perilaku *Use Cases* dan objek di dalam *system.*
7. *Class* Diagram untuk memodelkan struktur kelas.
8. *Object* Diagram untuk memodelkan struktur objek.
9. *Component* Diagram untuk memodelkan komponen *object.*
10. *Deployment* Diagram untuk memodelkan distribusi aplikasi.

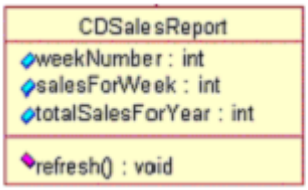
Untuk menggambarkan analisa dan *desain* diagram, UML memiliki seperangkat notasi yang akan digunakan ke dalam tiga kategori diatas yaitu struktur diagram, *behaviour* diagram dan *interaction* diagram. Berikut beberapa notasi dalam UML diantaranya :

1. *Actor*; menentukan peran yang dimainkan oleh *user* atau sistem lain yang berinteraksi dengan subjek. *Actor* adalah segala sesuatu yang berinteraksi langsung dengan sistem aplikasi komputer, seperti orang, benda atau lainnya. Tugas *actor* adalah memberikan informasi kepada sistem dan dapat memerintahkan sistem untuk melakukan sesuatu tugas.



Gambar 2.2 Actor
Sumber : Haviluddin, 2011

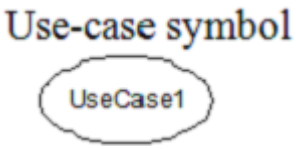
2. *Class* diagram; Notasi utama dan yang paling mendasar pada diagram UML adalah notasi untuk mempresentasikan suatu *class* beserta dengan atribut dan operasinya. *Class* adalah pembentuk utama dari sistem berorientasi objek.



Gambar 2.3 Class Diagram
Sumber : Haviluddin, 2011

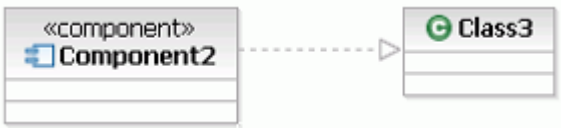
3. *Use Case* dan *use case specification*; *Use case* adalah deskripsi fungsi dari sebuah sistem perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem

dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. *Use case* merupakan awal yang sangat baik untuk setiap *fase* pengembangan berbasis objek, *design*, *testing*, dan dokumentasi yang menggambarkan kebutuhan sistem dari sudut pandang di luar sistem. Perlu diingat bahwa *use case* hanya menetapkan apa yang seharusnya dikerjakan oleh sistem.



Gambar 2.4 Use Case
Sumber : Haviluddin,2011

4. *Realization; Realization* menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah.



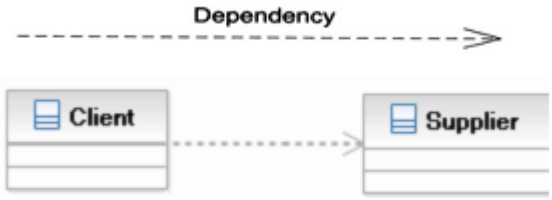
Gambar 2.5 Realization
Sumber : Haviluddin, 2011

5. *Interaction; Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar *obyek* maupun hubungan antar *obyek.*

Gambar 2.6 Interaction
Sumber : Haviluddin, 2011

6. *Dependency; Dependency* merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain.

Terdapat 2 *stereotype* dari *dependency*, yaitu *include* dan *extend*. *Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah). *Extend* menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan ke dalam elemen yang ada di garis dengan panah.

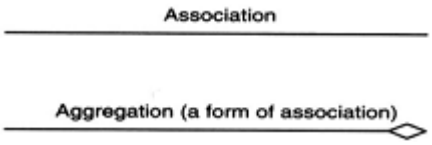


Gambar 2.7 Dependency
Sumber : Haviluddin, 2011

7. *Note; Note* digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. Note ini bisa disertakan ke semua elemen notasi yang lain.

Gambar 2.8 Note
Sumber : Haviluddin, 2011

8. *Association; Association* menggambarkan navigasi antar *class (navigation*), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (*multiplicity* antar *class*) dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*).

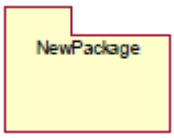


Gambar 2.9 Association
Sumber : Haviluddin, 2011

9. *Generalization; Generalization* menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik.

Gambar 2.10 Generalization
Sumber : Haviluddin,, 2011

10. Package; package adalah mekanisme pengelompokkan yang dgunakan untuk menandakan pengelompokkan elemen-elemen model.



Gambar 2.11 Package
Sumber : Haviluddin, 2011

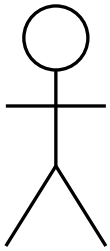
11. *Interface; Interface* merupakan kumpulan operasi berupa implementasi dari suatu class. Atau dengan kata lain implementasi operasi dalam *interface* dijabarkan oleh operasi di dalam *class.*

Gambar 2.12 Interface
Sumber : Haviluddin, 2011

Berikut adalah simbol dan komponen-komponen diagram pada UML :

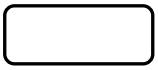
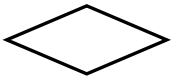
1. *Use Case* Diagram

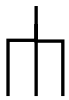
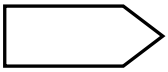
Tabel II.1 Komponen *Use Case* Diagram

Nama Komponen	Keterangan	Simbol
Use Case	*Use case* digambarkan sebagai lingkaran elips dengan nama *use case* dituliskan didalam elips tersebut.	
Actor	*Actor* adalah pengguna sistem. *Actor* tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan *input* atau memberikan *output*, maka aplikasi tersebut juga bisa dianggap sebagai *actor*.	
Association	Asosiasi digunakan untuk menghubungkan *actor* dengan *use case*. Asosiasi digambarkan dengan sebuah garis yang menghubungkan antara *Actor* dengan *Use Case*.	

2. *Activity* Diagram

Tabel II.2 Komponen *Activity* Diagram

Simbol	Keterangan
	Titik awal
	Titik akhir
	*Activity*
	Pilihan untuk mengambil keputusan
	Fork; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan paralel menjadi satu.

	*Rake*; Menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*Flow Final*)

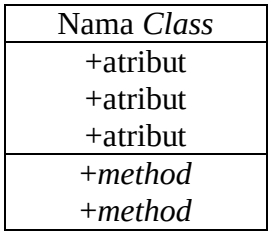
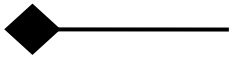
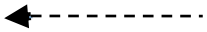
3. *Sequence* Diagram

Tabel II.3 Komponen *Sequence* Diagram

Nama Komponen	Keterangan	Simbol
Lifeline	*Lifeline* mengindikasikan keberadaan sebuah *object* dalam basis waktu. Notasi untuk *Lifeline* adalah garis putus-putus *vertikal* yang ditarik dari sebuah *object* .	
Activation	*Activation* dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah *lifeline*. mengindikasikan sebuah obyek yang akan melakukan sebuah aksi.	
Message	*Message*, digambarkan dengan anak panah horizontal antara *Activation* *Message* mengindikasikan komunikasi antara *object -object* .	

4. *Class* Diagram

Tabel II.4 Komponen *Class* Diagram

Nama Komponen	Keterangan	Simbol
Class	*Class* adalah blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah class digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari class. Bagian tengah mendefinisikan *property*/*atribut class*. Bagian akhir mendefinisikan *method-method* dari sebuah *class*.	
Association	Sebuah asosiasi merupakan sebuah *relationship* paling umum antara 2 *class* dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 *class*. Garis ini bisa melambangkan *tipe-tipe relationship* dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah *relationship*. (Contoh: *One-to-one*, *one-to-many*, *many-to-many*).	
Composition	Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/*solid*.	
Dependency	Kadangkala sebuah *class* menggunakan *class* yang lain. Hal ini disebut *dependency*. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu	

	class yang menggunakan *class* yang lain. Sebuah *dependency* dilambangkan sebagai sebuah panah bertitik-titik.	
Aggregation	*Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.