BAB II

LANDASAN TEORI

2.1 Aplikasi Mobile

Adapun pengertian Aplikasi *Mobile* adalah sebagai berikut

2.1.1 Pengertian Aplikasi

Perangkat lunak aplikasi menurut jogiyanto (Analisis dan Desain Sistem Informasi 1999:12) adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna atau user. Biasanya perangkat lunak system dapat 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 dan lain sebagainya.

Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau suite aplikasi (application suite). Contohnya adalah *Microsoft Office* dan *OpenOffice.org*, 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.

2.1.2 Pengertian Aplikasi Mobile

Pengertian aplikasi menurut Reito Meiler (Profesional Android Application Development 2009:2) madalah program yang digunakan orang untuk melakukan sesuatu pada sistem komputer. *Mobile* dapat diartikan sebagai perpindahan yang mudah dari satu tempat ke tempat yang lain, misalnya telepon *mobile* berarti bahwa terminal telepon yang dapat berpindah dengan mudah dari satu tempat ke tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi. Sistem aplikasi *mobile* merupakan aplikasi yang dapat digunakan walaupun pengguna berpindah dengan mudah dari satu tempat ketempat lain lain tanpa terjadipemutusan atau terputusnya komunikasi. Aplikasi ini dapat diakses melalui perangkat nirkabel seperti pager, seperti telepon seluler dan *PDA.* Adapun karakteristik perangkat *mobile* yaitu:

1. Ukuran yang kecil : Perangkat *mobile* memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

2. Memory yang terbatas : Perangkat *mobile* juga memiliki memory yang kecil, yaitu primary (RAM) dan secondary (disk).
3. Daya proses yang terbatas : Sistem *mobile* tidaklah setangguh rekan mereka yaitu desktop.
4. Mengkonsumsi daya yang rendah : Perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin desktop
5. Kuat dan dapat diandalkan : Karena perangkat *mobile* selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.
6. Konektivitas yang terbatas : Perangkat *mobile* memiliki bandwith rendah, beberapa dari mereka bahkan tidak tersambung.
7. Masa hidup yang pendek : Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala.

2.2 Android

Pembahasan mengenai *android* dan tools yang digunakan dalam pembuatan aplikasi, akan dijelaskan pada sub bab berikut:

2.2.1 Pengertian Android

Menurut *Nasruddin Safaat h* (Pemrograman aplikasi *mobile* smartphone dan tablet PC berbasis *android* 2012:1) *Android* adalah sebuah sistem operasi pada handphone yang bersifat terbuka dan berbasis pada sistem operasi *Linux*. *Android*

bisa digunakan oleh setiap orang yang ingin menggunakannya pada perangkat mereka. *Android* menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang akan digunakan untuk bermacam peranti bergerak.

Awalnya, Google Inc. membeli *Android* Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-*Mobile*, dan Nvidia.

Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama Open Handset Alliance menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode–kode *Android* di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler.

2.2.2 Sejarah dan Perkembangan Android

Pada mulanya terdapat berbagai macam sistem operasi pada perangkat selular, (*Pemrograman aplikasi mobile smartphone dan tablet PC berbasis android* 2012:79)diantaranya sistem operasi *Symbian, Microsoft Windows Mobile, Mobile Linux*, *iPhone*, dan sistem operasi lainnya. Namun diantara sistem operasi yang ada belum mendukung standar dan penerbitan API yang dapat dimanfaatkan secara keseluruhan dan dengan biaya yang murah. Kemudian Google ikut berkecimpung di

dalamnya dengan platform *Android*, yang menjanjikan keterbukaan, keterjangkauan, open source dan framework berkualitas.

Pada tahun 2005, Google mengakuisisi perusahaan *Android Inc*. untuk mamulai perkembangan platform *Android*. Dimana terlibat pengembangan ini adalah *Andy Rubin, Rich Miner, Nick Sears,* dan *Chris White*. Pada pertengahan 2007 sekelompok pemimpin industri bersama-sama membentuk analisis aliansi perangkat selular terbuka, Open Handset Alliance (OHA). Bagian dari tujuan aliansi ini adalah berinovasi dengan cepat dan menaggapi kebutuhan konsumen dengan lebih baik, dengan produk awalnya adalah platform *Android*. Dimana *Android* dirancang untuk melayani kebutuhan operator telekomunikasi, manufaktur handset, dan pengembangan aplikasi, OHA berkomitmen untuk membuat *Android* open source dengan lisensi Apache Versi 2.0.

2.2.3 Versi Android

Android pertama kali Nasruddin Safaat H (*Pemrograman aplikasi mobile smartphone dan tablet PC berbasis android* 2012:130)diluncurkan pada 5 November 2007, dan smartphone pertama yang menggunakan sistem operasi *Android* dikeluarkan oleh T-*Mobile* degnan sebutan G1 pada bulan September 2008. Hingga saat ini *Android* telah merilis beberapa versi *Android* untuk menyempurnakan versi sebelumnya. Selain berdsarkan penomoran, pada setiap versi *Android* terdapat kode

nama berdasarkan nama-nama kue. Hingga saat ini sudah terdapat beberapa versi yang telah diluncurkan, diantaranya:

1. *Android* 1.0 (API level 1) dirilis pada 23 September 2008.
2. *Android* 1.1 (API level 2) dirilis pada 09 Februari 2009.
3. *Android* 1.5 Cupcake (API level 3) dirilis pada 30 April 2009.
4. *Android* 1.6 Donut (API level 4) dirilis pada 15 September 2009.
5. *Android* 2.0 Eclair (API level 5) dirilis pada 26 Oktober 2009.
6. *Android* 2.0.1 Eclair (API level 6) dirilis pada 3 Desember 2009.
7. *Android* 2.1 Eclair (API level 7) dirilis pada 12 Januari 2010.
8. *Android* 2.2 s/d 2.2.3 Froyo (API level 8) dirilis pada 20 Mei 2010, 18 Januari 2011, 22 Januari 2011, 21 November 2011.
9. *Android* 2.3 s/d 2.3.2 Gingerbread (API level 9) dirilis pada 6 Desember 2010, Desember 2010, Januari 2011.
10. *Android* 2.3.3 s/d 2.3.7 Gingerbread (API level 10) dirilis pada 9 Februari 2011, 28 April 2011, 25 Juli 2011, 2 September 2011, 21 September 2011.
11. *Android* 3.0 Honeycomb (API level 11) dirilis pada 22 Februari 2011.
12. *Android* 3.1 Honeycomb (API level 12) dirilis pada 10 Mei 2011.
13. *Android* 3.2 Honeycomb (API level 13) dirilis pada 15 Juli 2011, 20 September 2011, 30 Agustus 2011, Desember 2011, Januari 2012, Februari 2012.

14. *Android* 4.0 s/d 4.0.2 Ice Cream Sandwich (API level 14) dirilis pada 19 Oktober 2011, 21 Oktober 2011, 28 November 2011.
15. *Android* 4.0.3 s/d 4.0.4 Ice Cream Sandwich (API level 15) dirilis pada 16 Desember 2011, 29 Maret 2012.
16. *Android* 4.1 s/d 4.1.2 Jelly Bean (API level 16) dirilis pada 9 Juli 2012, 23 Juli 2012, 9 Oktober 2012.
17. *Android* 4.2 s/d 4.2.2 Jelly Bean (API level 17) dirilis pada 13 November 2012, 27 November 2012, 11 Februari 2013.
18. *Android* 4.3 Jelly Bean (API level 18) dirilis pada 24 Juli 2013.
19. *Android* 4.4 KitKat (API level 19) dirilis pada Oktober 2013.

2.2.4 Kelebihan Android

Sudah banyak platform untuk perangkat selular saat ini, termasuk didalamnya Symbian, iPhone, Windows *Mobile*, BlackBerry, Java *Mobile* Edition, Linux *Mobile* (LiM0), dan banyak lagi. Namun ada beberapa hal yang menjadi kelebihan *Android*. Walaupun beberapa fitur-fitur yang ada telah muncul sebelumnya pada platform lain, *Android* adalah yang pertama menggabungkan hal seperti berikut:

1. Keterbukaan, Bebas pengembangan tanpa dikenakan biaya terhadap system karena berbasiskan *Linux* dan *open source*. Pembuat perangkat menyukai hal ini karena dapat membangun platform yang sesuai yang diinginkan tanpa harus membayar royality. Sementara pengembang software menyukai karena

Android dapat digunakan diperangkat manapun dan tanpa terikat oleh vendor manapun.

2. Arsitektur komponen dasar *Android* terinspirasi dari teknologi internet Mashup. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan service, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, database SQL, browser dan penggunaan peta. Semua itu sudah tertanam pada *android* sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja system menjadi lebih stabil. Pengguna tak perlu kawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.
5. Dukungan grafis dan suarat terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan OpenGL memungkinkan membuat aplikasi maupun game yang berbeda.
6. Portabilitas aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun yang akan datang. Semua program ditulis dengan menggunakan bahas pemrograman Java dan dieksekusi oleh mesin

2.3 Pengertian Lowongan Kerja

Lowongan artinya peluang yang kosong alias belum terisi, dalam hal ini peluang untuk bekerja. Sebenarnya lowongan kerja banyak sekali, tidak terbatas pada pekerjaan kantoran atau kerja di pabrik dan perusahaan swasta lainnya. Dalam bisnis pun orang bekerja, berjualan pun peluang kerja, bahkan jika ada peluang untuk bekerja di rumah sendiri pun misalnya dengan menjalankan bisnis internet, itu juga lowongan kerja.

Menurut data statistik Indonesia, tenaga kerja (manpower) adalah seluruh penduduk dalam usia kerja (berusia 15 tahun atau lebih) yang potensial dapat memproduksi barang dan jasa. Sebelum tahun 2000, Indonesia menggunakan patokan seluruh penduduk berusia 10 tahun ke atas (lihat hasil Sensus Penduduk 1971, 1980 dan 1990). Namun sejak Sensus Penduduk 2000 dan sesuai dengan ketentuan internasional, tenaga kerja adalah penduduk yang berusia 15 tahun atau lebih

2.4 Pengertian Basis Data (*Database*)

Menurut *Yahya* (2001:8), basis data (*database*) didefinisikan sebagai himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah yang disimpan dalam media penyimpanan elektronis.

Menurut pengertian diatas dapat disimpulkan bahwa basis data adalah satu kelompok organisasi data terpusat. Basis data secara umum dianalogikan sebagai lemari dokumen atau sekumpulan lemari dokumen.

2.5 Perangkat Lunak Pendukung

Dibawah ini merupakan penjelasan dari masing-masing perangkat lunak:

2.5.1 Eclipse

Menurut *Wina Noviani*(2011:10), *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, 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.

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.

2.5.2 Arsitektur Eclipse

Menurut *Wina Noviani*(2011:10), Sejak versi 3.0, *Eclipse* pada dasarnya merupakan sebuah kernel. Apa yang dapat digunakan di dalam *Eclipse* sebenarnya adalah fungsi dari plug-in yang sudah dipasang (diinstal). Ini merupakan basis dari *Eclipse* yang dinamakan *Rich Client Platform* (RCP).

Berikut ini adalah komponen yang membentuk RCP:

1) Core platform
2) OSGi

3) SWT (Standard Widget Toolkit)

4) JFace

5) Eclipse Workbench

Secara standar *Eclipse* selalu dilengkapi dengan *JDT* (*Java Development Tools*), plug-in yang membuat *Eclipse* kompatibel untuk mengembangkan program *Java*, dan *PDE (Plug-in Development Environment)* untuk mengembangkan *plug-in* baru. *Eclipse* beserta plug-in-nya diimplementasikan dalam bahasa pemograman *Java*.

Konsep *Eclipse* adalah *IDE* adalah

1. Terbuka (open),

2. Mudah diperluas (*extensible*) untuk apa saja, dan

3. Tidak untuk sesuatu yang spesifik.

Eclipse tidak saja untuk mengembangkan program *Java*, tetapi juga untuk berbagai macam keperluan. Perluasan apapun cukup dengan menginstal *plug-in* yang dibutuhkan. Apabila ingin mengembangkan program *C/C++* maka telah terdapat *plug-in CDT (C/C++ Development Tools)* yang dapat dipasang di Eclipse untuk Eclipse menjadi perangkat untuk pengembangan *C/C++*.

Pengembangan secara visual bukan hal yang tidak mungkin oleh *Eclipse*, *plug-in UML2* tersedia untuk membuat diagram *UML*. Dengan menggunakan *PDE*

setiap orang bisa membuat plug-in sesuai dengan keinginannya. Salah satu situs yang menawarkan plug-in yang gratis adalah *Eclipse downloads by project*.

2.5.3 Pemograman Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh *James Gosling* saat masih bergabung di Sun *Microsystems* saat ini merupakan bagian dari *Oracle* dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada *C dan C++* namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal.

Aplikasi-aplikasi berbasis *java* umumnya dikompilasi ke dalam *p-code (bytecode)* dan dapat dijalankan pada berbagai *Mesin Virtual Java (JVM)*. *Java* merupakan bahasa pemograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Menurut *Noviyanto(2009)* Karakteristik *Java* antara lain :

- Sederhana (Simple)

- Berorientasi Objek (Object Oriented)
- Terdistribusi (Distributed)
- Interpreted
- Robust
- Aman (Secure)
- Architecture Neutral
- Portable
- Performance
- Multithreaded
- Dinamis

Macam-macam *Java 2 Software Developer Kit (J2SDK)* antara lain :

- J2SE (*Java 2 Standard Edition*)
- J2EE (*Java 2 Enterprise Edition*)
- J2ME (*Java 2 Micro Edition*)

Java mempunyai platform yaitu :

- Java Virtual Machine (*Java VM*)
- Java Application Programming Interface *(Java API*)

Aplikasi-aplikasi *java* antara lain :

- Stand alone

- Applets
- Aplikasi berbasis Windows (*GUI*)
- Java Servlet
- Java Server Pages (*JSP*)
- Java Beans dan Enterprise Java Beans
- Java Micro Edition

2.5.4 Arsitektur Java

Secara arsitektur, Java tidak berubah sedikitpun semenjak awal mula bahasa tersebut dirilis. Kompiler Java (yang disebut dengan Javac atau Java Compiler) akan mentransformasikan kode-kode dalam bahsa Java ke dalam suatu bytecode. Apa itu bytecode? Bytecode adalah sekumpulan perintah hasil kompilasi yang kemudian dapat dieksekusi melalui sebuah mesin komputer abstrak, yang disebut dengan *JVM (Java Virtual Machine)*. *JVM* juga sering dinamakan sebagai interpreter, karena sifatnya yang selalu menerjemahkan kode-kode yang tersimpan dalam bytecode dengan cara baris demi baris.

2.6. Unified Modeling Language (UML)

Menurut *Adi Nugroho(2005:110).* "Unified Modeling Language (UML) adalah alat bantu analisis serta perancangan perangkat lunak berbasis objek". Unified Modeling Language (UML) merupakan standard modeling language yang terdiri dari

kumpulan-kumpulan diagram, dikembangkan untuk membantu para pengembang sistem dan software agar bisa menyelesaikan tugas-tugas seperti: Spesifikasi, Visualisasi, Desain Arsitektur, Konstruksi, Simulasi dan testing serta Dokumentasi".

Berdasarkan beberapa pendapat yang dikemukakan diatas dapat ditarik kesimpulan bahwa "Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk menvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis OO (Object Oriented)". UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan system yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan. Bagian-bagian utama dari UML adalah View, Diagram, model element, dan general mechanisme.

1. **View**

   View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. View bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah Diagram. Beberapa jenis View dalam

UML antara lain: *Use case* View, *Logical* View, *Component* View, *concurrency* View, dan *Deployment* View.

2. *Use case* View

Use case View mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan external actors. Actor yang berinteraksi dengan sistem dapat berupa user atau sistem lainnya.View ini digambarkan dalam Use case Diagrams dan kadang-kadang dengan Activity Diagrams. View ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

3. *Logical* View

Logical View mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (Class, object,dan relationship) dan kolaborasi dinamis yang terjadi ketika object mengirim pesan ke object lain dalam suatu fungsi tertentu. View ini digambarkan dalam Class Diagram untuk struktur statis dan dalam *State*, *Sequence*, *Collaboration*, dan *Activity* Diagram untuk model dinamisnya. View ini digunakan untuk perancang (designer) dan pengembang (developer).

4. *Component* View

Component View mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari code module diperlihatkan dengan struktur dan ketergantungannya juga alokasi

sumber daya komponen dan informasi administrative lainnya. View ini digambarkan dalam *Component* View dan digunakan untuk pengembang (*developer*).

5. ***Concurrency* View**

   Concurrency View membagi sistem ke dalam proses dan prosesor. View ini digambarkan dalam Diagram dinamis (*State, Sequence, Collaboration,* dan *Activity* Diagram) dan Diagram implementasi (*Component* dan *Deployment* Diagram) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

6. ***Deployment* View**

   Deployment View mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya.View ini digambarkan dalam *Deployment* Diagram dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

7. **Diagram**

   Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah Diagram merupakan bagian dari suatu View tertentu danketika digambarkan biasanya dialokasikan untuk View tertentu. Adapun jenis Diagram antara lain :

1. *Use Case* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:155, Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami.

2. *Activity* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:161*, Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

3. *Sequence* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:165*, Diagram sekuen atau *sequence* Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikrimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka

harus diketahui objek-objek yang akan terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case.*

4. *Collaboration* Diagram

Collaboration diagram adalah bentuk lain *sequence* diagram, dimana *sequence* diagram diorganisir oleh waktu, sedangkan *collaboration* diagram diorganisir menurut ruang/space.

5. *Class* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:141*, Diagram kelas atau *class diagram* adalah menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas, operasi atau metode adalah fungsi-fungsi yang dimiliki oleh kelas.

Diagram kelas dibuat agar pembuat program atau *progammer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron.

6. *Component* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:148*, Diagram komponen atau *component diagram* dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Diagram komponen berfokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem.

7. *Deployment* Diagram

Menurut *Rosa A.S, M. Shalahudin, Rekayasa Perangkat Lunak, 2013:154*, Diagram deployment atau *deployment diagram* menunjukan konfigurasi komponen dalam proses eksekusi aplikasi. Diagram deployment juga dapat digunakan untuk memodelkan hal-hal seperti, sistem tambahan (*embedded system)* yang mengambarkan rancangan *device, node,* dan *hardware.*

Adapun tujuan penggunaan UML yaitu :

a. Memberikan bahasa pemodelan yang bebas dari berbagai bahas pemrograman dan proses rekayasa.
b. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.
c. Memberikan model yang siap pakai, bahsa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
d. UML bisa juga berfungsi sebagai sebuah (blue print) cetak biru karena sangat lengkap dan detail. Dengan cetak biru ini maka akan bias diketahui informasi secara detail tentang coding program atau bahkan membaca program dan menginterpretasikan kembali ke dalam bentuk Diagram (reserve enginering).

Sedangkan perangkat lunak yang mendukung pembuatan diagaram UML yaitu :

a) StarUML

StarUML adalah sebuah proyek Staropen source untuk mengembangkan cepat, fleksibel, extensible, featureful, dan bebas-tersedia UML / platform MDA berjalan pada platform Win32.Tujuan dari proyek StarUML adalah untuk membangun sebuah alat pemodelan perangkat lunak dan juga platform yang menarik adalah pengganti alat UML komersial seperti Rational Rose, Bersama dan sebagainya.

b) Acceleo

Acceleo adalah generator kode yang mengubah model menjadi kode. Acceleo mudah digunakan dan menyediakan "dari rak" generator (Jee,. Bersih, Php …) dan template editor untuk Eclipse.

c) ArgoUML

ArgoUML adalah Staropen source UML modeling tool terkemuka dan termasuk dukungan untuk semua Diagram UML standar 1,4. Ini berjalan pada setiap platform Java dan tersedia dalam bahasa sepuluh.

2.6.1 SQLite

Sistem database yang digunakan oleh *Android* adalah SQLite. SQLite adalah database berbasis SQL, tetapi memiliki banyak keterbatasan, salah satunya penyediaan tipe data yang terbatas dan tidak mendukung konsep normalisasi.

Database yang dibuat hanya dapat diakses oleh aplikasi itu sendiri. Lalu untuk apa kita menggunakan SQLite? Umumnya digunakan untuk menyimpan data sementara saja. Sedangkan untuk data permanen biasanya digunakan web service.

SQLite adalah mesin database SQL embedded. Tidak seperti kebanyakan database SQL lainnya, SQLite tidak memiliki proses server yang terpisah. SQLite membaca dan menulis secara langsung ke file disk biasa. Sebuah database SQL lengkap dengan beberapa tabel, indeks, pemicu, dan tampilan, yang terkandung dalam sebuah file disk tunggal. Format file database adalah cross-platform - kita dapat dengan bebas menyalin database antara 32-bit dan 64-bit atau antara *big-endian* dan *little-endian* arsitektur. Fitur-fitur ini membuat SQLite menjadi pilihan populer sebagai *Format file Application* . Pikirkan SQLite bukan sebagai pengganti untuk Oracle tetapi sebagai pengganti *fopen ()*