BAB II

TINJAUAN PUSTAKA

II.1. Sistem Informasi

II.1.1. Pengertian Sistem

Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen atau variabel–variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Prajudi Atmosudirdjo menyatakan bahwa suatu sistem terdiri atas objek – objek, atau unsur – unsur, atau komponen – komponen yang berkaitan dan berhubungan satu sama lainnya sedemikian rupa sehingga unsur – unsur tersebut merupakan suatu kesatuan pemprosesan atau pengolahan yang tertentu. (Tata Sutabri ; 2004 : 2).

II.I.2. Karakteristik Sistem

Model umum sebuah sistem terdiri dari input, proses, dan output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana mengingat sebuah sistem dapat mempunyai beberapa masukan dan keluaran sekaligus. Selain itu sebuah sistem juga memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. Adapun karakteristik yang dimaksud menurut (Tata Sutabri ; 2004 : 11) adalah sebagai berikut :

a. Komponen sistem *(Components)*

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk suatu kesatuan. Komponen- komponen sistem

tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat-sifat dari sistem yanng menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem yang lebih besar, yang disebut dengan supra sistem.

a. Batasan sistem *(Boundary)*

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lainnya atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisah-pisahkan.

b. Lingkungan luar sistem *(Environtment)*

   Bentuk apapun yang ada di luar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut dengan lingkungan luar sistem.

c. Penghubung sistem *(Interface)*

   Sebagai media yang menghubungkan sistem dengan subsistem yang lain disebut dengan penghubung sistem atau *interface*. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lain. Keluaran suatu subsistem akan menjadi masukan untuk subsistem yanng lain dengan melewati penghubung. Dengan demikian terjadi suatu integrasi sistem yang membentuk satu kesatuan.

d. Masukan sistem *(Input).*

   Energi yang dimasukan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan *(maintenance input)* dan sinyal *(signal input).*

e. Keluaran sistem *(Output)*

Hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain.

f. Pengolahan sistem *(Process)*

Suatu sistem dapat mempunyai suatu proses yanng akan mengubah masukan menjadi keluaran.

g. Sasaran sistem *(Objective)*

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministik. Kalau suatu sistem tidak memiliki sasaran, maka operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan.

II.1.3. Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan (Tata Sutabri : 2005 : 23).

II.1.4. Sistem Informasi

Sistem informasi secara sederhana dapat diartikan sebagai kumpulan dari beberapa komponen yang saling berinteraksi untuk mencapai hasil dari satu tujuan. Pengertian sederhana ini sesuai dengan pendapat O'Brian(2006:5)" Sistem informasi dapat merupakan kombinasi teratur apapun dari orang-orang, hardware, software, jaringan komunikasi, dan sumber data yang mengumpulkan, mengubah dan menyebarkan informasi dalam sebuah organisasi". Defenisi

tersebut dapat dijelaskan sistem informasi adalah susunan dari orang, data, pemprosesan dan teknologi informasi yang dibutuhkan untuk mendukung sebuah organisasi(Henny Hendarti:2011:1)

II.2. Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) merupakan suatu rancangan sistem informasi untuk mengerjakan data berunsur atau koordinat geografis. Teknologi SIG menyatu dengan operasi database seperti pencarian data dan analisa statistik dan analisa geografis yang disajikan dalam bentuk peta. (Riadi dan Kifli Nasution;2008:4).

perangkat keras dan sistem operasi yang dibutuhkan dalam software ArcView adalah :

1. Komputer pentium atau yang lebih tinggi dengan mikro prosesor Intel.
2. RAM *(Random Access Memory)* minimal 64 MB atau memori utama yang paling baik 1GB.
3. Monitor berwarna SVGA dengan kualitas 256 warna atau lebih besar.
4. Mouse dan keyboard standar.
5. Digitizer, printer dan plotter bersifat opsional. Digitizer diperlukan apabila melakukan digitasi peta. Printer dan plotter dibutuhkan bila melakukan percetakan.
6. ArcView GIS 3.2, dapat dijalankan pada informasi Windows NT, Windows Seven, Windows XP, atau LINUX. (Riadi dan Kifli Nasution; 200:9).

II.2.1. Sistem Development Life Cycle (SDLC)

Menurut **James R. Clageet(Pengenalan Sistem Komputer,2001:45)** Proses pengembangan sistem mempunyai beberapa tahapan mulai dari sistem itu direncanakan sampai dengan sistem tersebut diterapkan, dioperasikan dan dipelihara. Adapun tahapan-tahapan tersebut adalah sebagai berikut:

A. Perencanaan Sistem *(system planning)*

1. Penerimaan untuk studi sistem
2. Pengamatan awal
3. Studi kelayakan

B. Analisa Sistem *(system analysis)* meliputi beberapa langkah:

1. Mendefenisikan kembali masalah
2. Memahami sistem yang ada
3. Menentukan permintaan pemakai dan kendala yang ada
4. Membuat logika dan menyelesaikan

C. Desain Sistem *(system design)* meliputi:

1. Desain output
2. Desain input

D. Implementasi Sistem *(system implementation) meliputi beberapa langkah:*

1. Membangun sistem
2. Pembuatan program

II.3. *Database* MySQL

Menurut Supardi (2007:97), perangkat lunak MySQL adalah perangkat lunak basis data server yang terkenal dan bersifat open-source dengan dukungan driver yang luas dari berbagai vendor. Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya. SQL (*Structured Query Language*). SQL adalah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak pengelolah basis data kompetitor lainnya.

II.4. Java

Java adalah suatu teknologi di dunia *software* komputer, yang merupakan suatu bahasa pemrograman, dan sekaligus suatu *platform*. Sebagai bahasa pemrograman, Java dikenal sebagai bahasa pemrograman tingkat tinggi. Java mudah dipelajari, terutama bagi *programmer* yang telah mengenal C/C++. Java merupakan bahasa pemrograman berorientasi objek yang merupakan paradigma pemrograman masa depan. sebagai bahasa

Pemrograman Java dirancang menjadi handal dan aman. Java juga dirancang agar dapat dijalankan di semua *platform*. Dan juga dirancang untuk menghasilkan aplikasi – aplikasi dengan performansi yang terbaik, seperti aplikasi

database Oracle 8i/9i yang core-nya dibangun menggunakan bahasa pemrograman Java. Sedangkan Java bersifat *neutral architecture,* karena *Java Compiler* yang digunakan untuk mengkompilasi kode program Java dirancang untuk menghasilkan kode yang netral terhadap semua arsitekture perangkat keras yang disebut sebagai *Java Bytecode*.

Sun membagi arsitektur Java membagi tiga bagian, yaitu:

1. **Enterprise Java (J2EE)** untuk aplikasi berbasis web, aplikasi system tersebar dengan beraneka ragam klien dengan kompleksitas yang tinggi merupakan superset dari Standar Java
2. **Standar Java (J2SE),** ini adalah yang biasa dikenal sebagai bahasa Java.
3. **Micro Java (J2ME)** merupakan subset dari J2SE dan salah satu aplikasinya yang banyak dipakai adalah untuk *wireless device / mobile device.* (Widianto; 2011:).

II.4.1. Sejarah Java

Java diciptakan oleh suatu tim yang dipimpin oleh Patrick Naughton dan James Gosling dalam suatu proyek dari *Sun Microsystem* yang memiliki kode *Green* dengan tujuan untuk menghasilkan bahasa komputer sederhana yang dapat dijalankan di peralatan sederhana dengan tidak terikat pada *arsitekture* tertentu. Mulanya disebut OAK, tetapi karena OAK sendiri merupakan nama dari bahasa pemrograman computer yang sudah ada. Maka Sun mengubahnya menjadi Java.

Sun kemudian meluncurkan *browser* dari Java yang disebut *Hot Java* yang mampu menjalankan *applet*. Setelah itu teknologi Java diadopsi oleh *Netscape* yang memungkinkan program Java dijalankan di *browser Netscape* yang

kemudian diikuti *Internet Explorer*. Karena keunikanya dan kelebihanya, teknologi Java mulai menarik banyak *vendor* seperti IBM,*Symantec, Inprise*.

Sun merilis versi awal Java secara resmi pada awal tahun 1996 yang kemudian terus berkembang hingga muncul JDK 1.1, kemudian JDK 1.2 yang mulai disebut sebagai versi Java2 karena banyak mengandung peningkatan dan perbaikan. Perubahan utama adalah adanya *Swing* yang merupakan teknologi GUI (Graphical User Interface) yang mampu menghasilkan window yang *portabel*. Dan pada tahun 1998 – 1999 lahirlah teknologi J2EE (Java 2 Enterprise Edition) yang berbasis J2SE yang diawali dengan servlet dan EJB kemudian diikuti JSP. Java juga menjadi lebih cepat populer di lingkungan *server side* dikarenakan kelebihanya di lingkungan network dan terdistribusi serta kemampuan *multithreading*. Sedangkan J2ME (Java 2 Micro Edition) dapat menghasilkan aplikasi *mobile* baik *games* maupun *software* yang dapat dijalankan di peralatan *mobile* seperti ponsel. (Widianto; 2011 :).

II.4.2. Java Virtual Machine (JVM)

Java Virtual Machine (JVM) adalah sebuah mesin *imajiner* (maya) yang bekerja dengan menyerupai aplikasi pada sebuah mesin nyata. *Java Virtual Machine* (JVM) menyediakan *spesifikasi hardware* dan *platform* dimana kompilasi kode Java terjadi. Spesifikasi inilah yang membuat aplikasi berbasis Java menjadi bebas dari *platform* manapun karena proses kompilasi diselesaikan oleh *Java Virtual Machine* (JVM).

Aplikasi program Java diciptakan dengan *file* teks berekstensi *.java*. Program ini dikompilasi menghasilkan satu berkas *bytecode* berekstensi *.class*

atau lebih. *Bytecode* adalah serangkaian instruksi serupa instruksi kode mesin. Perbedaannya adalah kode mesin harus dijalankan pada sistem komputer dimana kompilasi ditujukan, sementara *bytecode* berjalan pada *java interpreter* yang tersedia di semua *platform* sistem komputer dan sistem operasi (*Jakaria, 2010, Pengenalan Java*).

II.5. Aplikasi Eclipse

Menurut *Widianto Pratama (Tutorial Android Programming, 2011)* Eclipse merupakan komunitas *open source* yang bertujuan menghasilkan *platform* pemrograman terbuka. Eclipse terdiri dari *framework* yang dapat dikembangkan lebih lanjut, peralatan bantu untuk membuat dan memanage *software* sejak awal hingga diluncurkan. *Platform* Eclipse didukung oleh ekosistem besar yang terdiri dari vendor tekonologi, *start-up inovatif*, universitas, riset institusi serta individu.

Banyak orang mengenal Eclipse sebagai IDE (*Integrated Development Environment*) untuk bahasa Java, tapi Eclipse lebih dari sekedar IDE untuk Java. Komunitas Eclipse memiliki lebih dari 60 proyek *open source*. Proyek-proyek ini secara konsep terbagi menjadi 7 kategori :

1. Enterprise Development
2. Embedded and Device Development
3. Rich Client Platform
4. Rich Internet Applications
5. Application Frameworks
6. Application Lifecycle Management (ALM)
7. Service Oriented Architecture (SOA)

Secara umum Eclipse digunakan untuk membangun *software inovatif* berstandar industri, dan alat bantu beserta *frameworknya* membantu pekerjaan menjadi lebih mudah *(Widianto Pratama, Tutorial Android Programming, 2011).*

II.5.1. Eclipse

Eclipse menggunakan EPL (*Eclipse Public License*), yaitu lisensi yang memungkinkan organisasi untuk menjadikan Eclipse sebagai produk komersialnya, dan pada saat yang sama meminta orang yang melakukan perubahan untuk mengkontribusikan hasilnya kembali kepada komunitas.

II.5.2. Menjalankan Eclipse

a. Cari file bernama *eclipse.exe* (pada Windows) atau eclipse (pada Ubuntu) kemudian *double-click*
b. Pada saat Eclipse pertama kali dijalankan, Eclipse akan menanyakan *workspace*, yaitu folder tempat proyek dan data diletakkan. Anda bisa menempatkan di mana saja asalkan jangan di dalam folder Eclipse itu sendiri.
c. *Click Browse* dan pilih folder yang ada inginkan. Tik "*Use this as default and do not ask again*"
d. Halaman pembuka akan muncul. Klik "*Workspace*", tombol paling kanan berbentuk anak panah untuk masuk ke dalam *workspace* Anda.

II.5.3. Membuat Program Java

a. Klik "**File** -> **New** -> **Java Project**"
b. Isi nama proyek (misalnya SelamatDatang), kemudian klik "**Finish**"

c. Setelah Eclipse membuat proyek untuk Anda, di bagian kiri *workspace* Anda akan melihat struktur direktori proyek Anda yang dimulai dengan nama proyek, folder src, dan folder JRE *System Library*

d. Klik kanan pada folder src, kemudian "**New** -> **Package**"

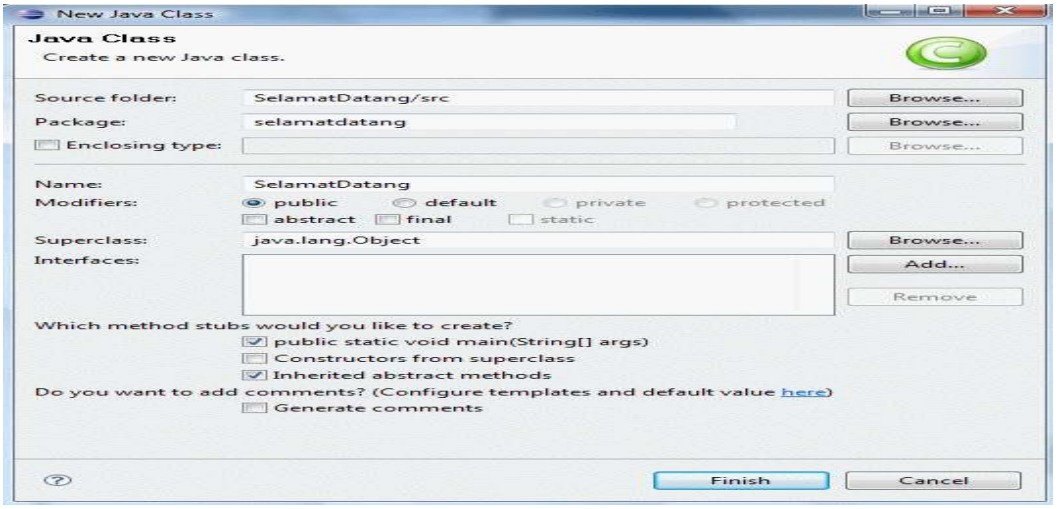
e. Isi nama *package* (misalnya selamatdatang), kemudian klik "**Finish**"

f. Klik kanan lagi pada folder selamatdatang, kemudian "**New** -> **Class**"

g. Isi nama *class* (misalnya SelamatDatang)

h. Karena *class* ini adalah *class* utama yang akan langsung dijalankan oleh JRE (*Java Runtime Environment*), *click* "*public static void main*(*String[] args*)" pada bagian "Which method stubs would you like to create?"

Berikut ini adalah form untuk membuat *project* baru dalam aplikasi eclipse yang dapat dilihat pada gambar II.1. berikut.



Gambar II.1. New Java Class

Sumber : *Widianto Pratama, Tutorial Android Programming, 2011*

i. Klik "**Finish**"

j. Eclipse akan membuat program kosong yang berisi *package* dan class sesuai dengan nama yang Anda masukkan pada tahap sebelumnya

k. Sekarang ketik program berikut di bawah "// TODO"

```
System.out.println("Selamat Datang!");
```

l. Kemudian simpan hasilnya.

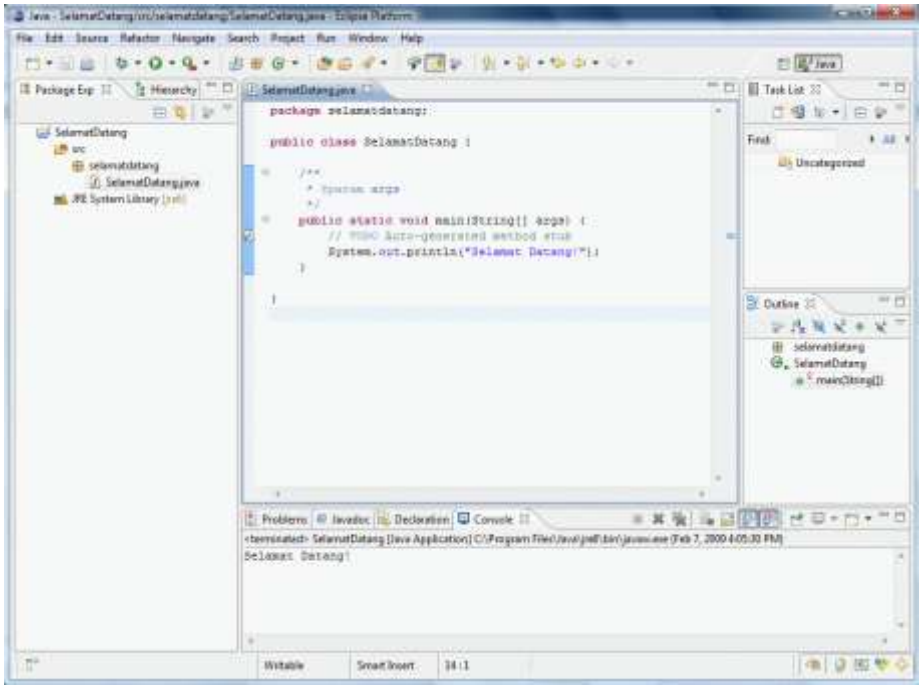
II.5.4. Menjalankan Program Java

a. Untuk menjalankan program Anda, klik "Run -> Run"

b. Di bagian bawah pada tab yang berjudul "Console" hasil program Anda ditampilkan

c. Program ini akan menampilkan tulisan Selamat Datang! seperti pada gambar berikut.

Berikut ini adalah form untuk menjalankan *project* baru dalam aplikasi eclipse yang dapat dilihat pada gambar II.2. dibawah ini



Gambar II.2. Layar Utama Eclipse

Sumber : Widianto Pratama, Tutorial Android Programming, 2011

II.6. Sistem Operasi Android

Android adalah sistem operasi mobile yang berjalan pada kernel **Linux**, yang dirilis pada 21 Oktober 2008. Awalnya, sistem operasi ini dikembangkan oleh Android, Inc, yang kemudian dibeli oleh *Google*, dan yang terakhir, sistem operasi ini dibeli oleh *Open Handset Alliance*, sebuah consortium dari 47 perusahaan *hardware*, *software*, dan *telecom* yang didirikan untuk membuat open standard bagi perangkat lunak mobile. Sistem operasi ini bersifat *free* dan *open source*. Perangkat mobile yang mendukung sistem operasi ini di antaranya adalah HTC Dream dan HTC Magic, ponsel keluaran vendor asal Taiwan, HTC.

Awalnya, *Google* membeli Android, 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*. *Google* sendiri ternyata mempunyai alasan cukup kuat untuk melirik pangsa ini, karena perkembangan teknologi telepon seluler sewasa ini sudah bukan merupakan evolusi lagi, melainkan sebuah revolusi. Babak baru dalam dunia telekomunikasi *nirkabel* ini terus bergulir dengan cepat. Jika sekarang seseorang mempunyai PC di rumah, dan notebook untuk ke kantor atau kuliah, serta berkomunikasi melalui telepon seluler. Maka pergerakkan yang kemudian terjadi sekarang adalah, orang mulai berpikir bagaimana menyatukan semuanya dalam satu genggaman.

Sebenarnya hal tersebut telah mulai dipenuhi dengan munculnya PDA/*smartphone*, di mana seseorang dapat merangkum semua kebutuhan

komputasinya dalam satu genggaman. Dan perkembangan inilah yang membuat *Google* berambisi untuk menguasai pangsa ini. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Adapun Fitur-fitur Android adalah sebagai berikut :

a. *Framework Aplikasi* yang mendukung penggantian komponen dan *reusable.*
b. Mesin *virtual Dalvik* dioptimalkan untuk perangkat *mobile*.
c. *Integrated browser* berdasarkan *engine open source WebKit*.
d. Grafis yang dioptimalkan dan didukung oleh perpustakaan grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional akselerasi hardware).
e. SQLite untuk penyimpanan data.
f. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC,AMR, JPG, PNG, GIF).
g. GSM Telephony (tergantung hardware).
h. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware).
i. Kamera, GPS, kompas, dan accelerometer (hardware tergantung).

II.6.1. Kelebihan Android

Adapun kelebihan Android adalah sebagai berikut ini *(Widianto Pratama, Tutorial Android Programming, 2011)*

1. Lengkap (*Complete Platform*) : Android dikatakan lengkap karena Android menyediakan tools untuk membangun software yang sangat lengkap dibanding dengan platform lain. Para pengembang dapat melakukan

pendekatan yang komprehensif ketika mereka mengembangkan suatu aplikasi pada platform Android.

2. Terbuka (*Open Source Platform*) : Platform Android diciptakan dibawah lisensi *open source*, dimana para pengembang bebas untuk mengembangkan aplikasi pada platform ini. Android menggunakan Linux kernel 2.6.
3. Bebas (*Free Platform*) : Android adalah *platform mobile* yang tidak memiliki batasan dalam mengembangkan aplikasinya. Tidak ada lisensi dalam mengembangkan aplikasi Android. Android dapat didisribusikan dan diperdagangkan dalam bentuk apapun.
4. Dapat menulis file di SD card.
5. Dapat mengolah database dengan SQLite.
6. *Application framework* berbasis komponen yang memudahkan reuse.
7. *Dalvik virtual machine* dioptimisasi untuk menjalankan program Java di mobile devices
8. *Integrated browser* berbasis WebKit engine
9. *Optimized graphics* tersedia 2D graphics library; dan OpenGL ES 1.0 untuk 3D graphics
10. Media support untuk MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF
11. Support pada GSM Telephony fasilitas telepon.
12. Mendukung penggunakan jaringan Bluetooth, EDGE, 3G, dan WiFi
13. Mendukung penggunaan *device* Kamera, GPS, *compass* dan *accelerometer*
14. Mendukung Multitouch .

II.7. UML

II.7.1. Pengenalan UML *(Unified Modeling Languange)*

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh model tunggal, yang membandu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OOP).

UML merupakan standar yang relatif terbuka yang dikontrol oleh *Object Management Group* (OMG), sebuah konsorium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar-standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem yang berorientasi objek. OMG mungkin lebih dikenal dengan standar-standar CORBA (*Common Object Request Broker Architecture*). (Martin Fowler, 2005 : 1)

II.7.2. Diagram-Diagram UML

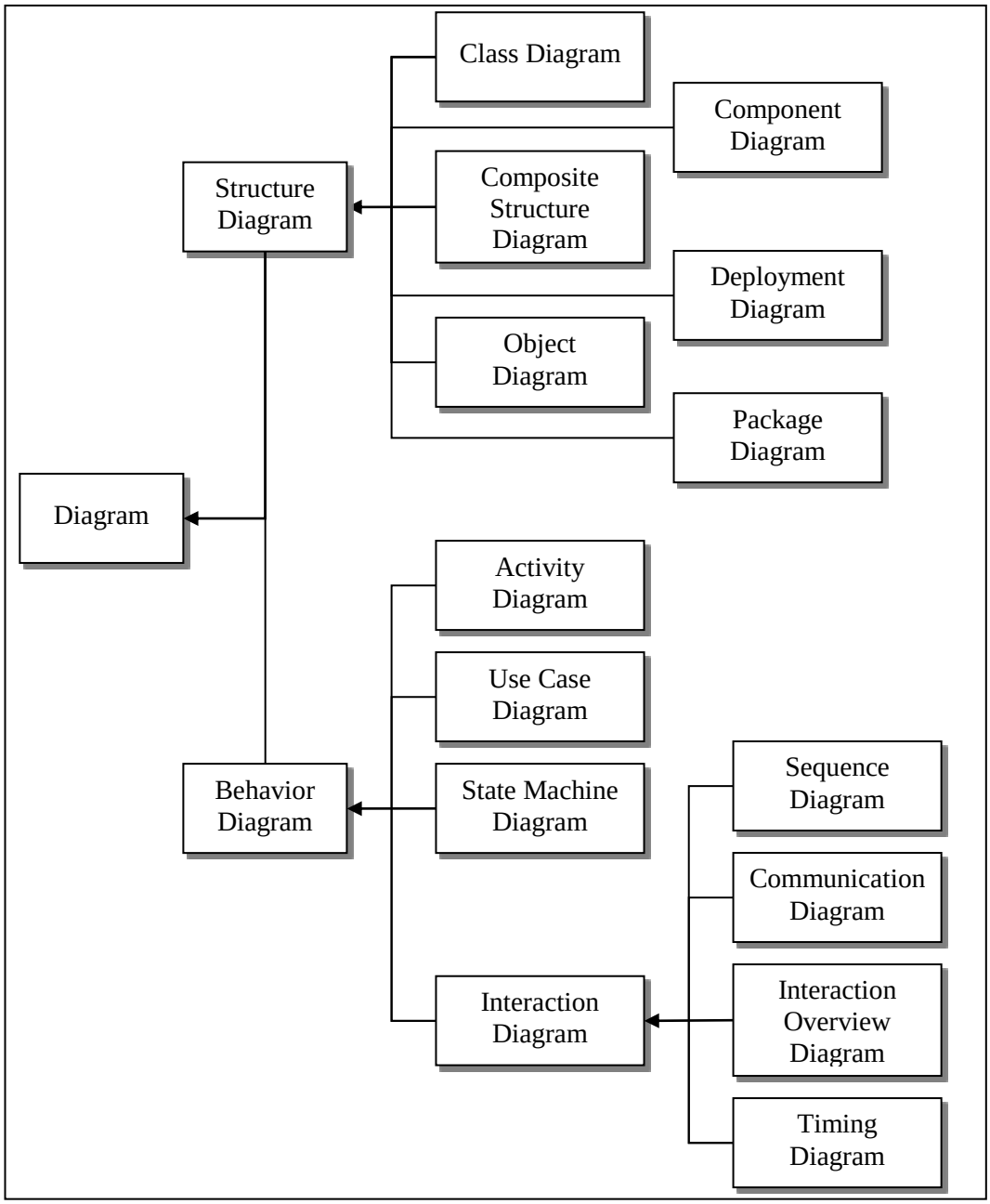
UML terdiri dari diagram, notasi, konsep dan aturan yang digunakan dalam memodelkan sistem. Diagram UML terdiri dari 13 jenis diagram yang memiliki fungsi dan notasi masing-masing. Kesembilan diagram ini dapat dibagi menjadi 2 kategori, yaitu diagram yang menggambarkan struktur yang statis dari sistem dan diagram yang menggambarkan struktur yang dinamis dari sistem.

1. Structur Diagram

   Merupakan diagram yang menggambarkan struktur hubungan statis dari elemen-elemen yang ada dalam sebuah model diantaranya *class*, *package*, dan *relationship* yang terjadi. (Martin Fowler, 2005 : 17)

2. Behavior Diagram

Merupakan kumpulan diagram yang menggambarkan hubungan dinamis antara class yang berada dalam komponen model yang dapat dilihat pada gambar II.3. dibawah ini.



Gambar II.3. Klasifikasi Jenis Diagram UML

Sumber : Martin Fowler (2005:19)

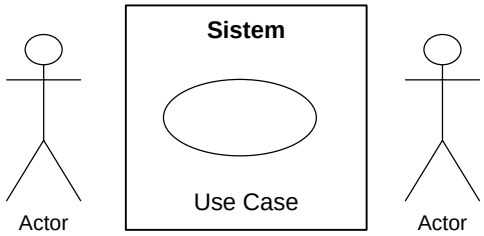
II.7.3. Karakter Diagram UML

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 dan ketika digambarkan biasanya dialokasikan untuk view tertentu.

1. *Use Case Diagram.*

Use case adalah abstraksi dari interaksi antara sistem dan actor. Use case bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah system dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. 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 client.

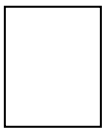
Diagram use case menunjukkan 3 aspek yaitu: aktor, use case dan sistem / sub sistem boundary. Actor mewakili peran orang, sytem yang lain atau laat ketika berkomunikasi dengan use case. Gambar II.4. berikut mengilustrasikan aktor, use case dan boundary dapat dilihat gambar dibawah ini.



Gambar II.4. Use Case Model

(Sumber : Munawar, 2005:64)

Berikut ini merupakan komponen-komponen pada use case diagram. Dapat dilihat pada tabel II.1

No.	Nama Komponen	Keterangan	Simbol
1	Actor	Actor juga dapat berkomunikasi dengan object , maka actor juga dapat diurutkan sebagai kolom. Simbol Actor sama dengan simbol pada Actor Use Case Diagram.	
2	Association	Association Asosiasi digunakan untuk menghubungkan actor dengan use case. Asosiasi digambarkan dengan sebuah garis yang menghubungkan antara Actor dengan use case	
3	System	Menspesifikasikan paket yang menampilkan sistem secara terbatas.	
4	Use Case	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor	

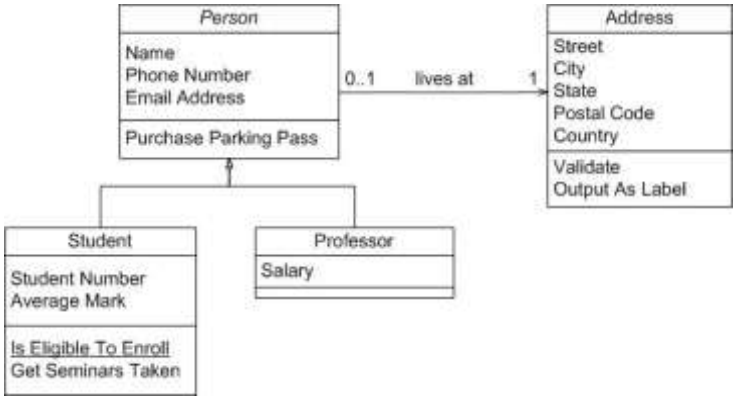
Sumber : Grady Booch (2005 : 30)

Tabel II.1. Komponen Use Case Diagram

2. *Class Diagram.*

Class adalah dekripsi kelompok obyek-obyek dengan property, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya class diagram dapat memberikan pandangan global atas sebuah system. Hal tersebut tercermin dari class- class yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem

biasanya mempunyai beberapa class diagram. Class diagram sangat membantu dalam visualisasi struktur kelas dari suatu sistem. Dapat dilihat pada gambar II.5 berikut.



Gambar II.5. Class diagram

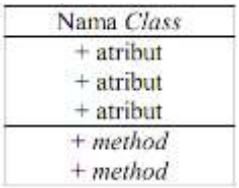
(Sumber : Munawar, 2005:221)

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).

Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain. Class memiliki tiga area pokok, yaitu :

1. Nama (dan stereotype)
2. Atribut
3. Metoda

Berikut ini merupakan komponen-komponen pada class diagram, dapat dilihat pada tabel II.2

No.	Nama Komponen	Keterangan	Simbol
1	Class	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 methodmethod dari sebuah class.	
2	Association	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).	
3	composition	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	

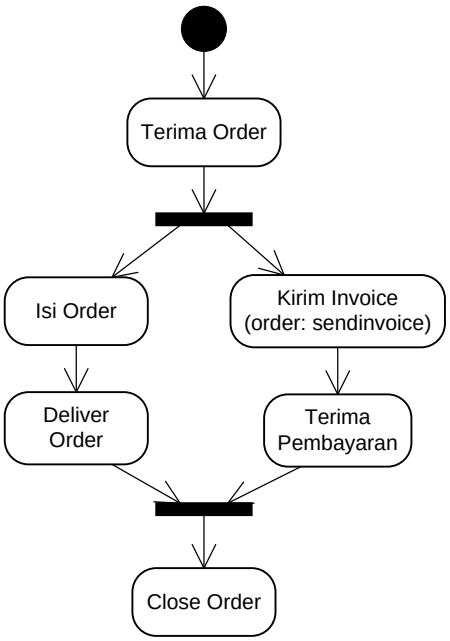
		bergantung tersebut. Sebuah relationship composition digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.	
4	Dependency	Dependency Kadangkala sebuah class menggunakan class yang lain. Hal ini disebut dependency. Umumnya penggunaan dependency digunakan untuk menunjukkan operasi pada suatu class yang menggunakan class yang lain. Sebuah dependency dilambangkan Sebagai panah bertitik-titik.	
5.	Aggregation	Aggregation mengindikasikan keseluruhan bagian relationship dan biasa disebut relasi	

Sumber : Grady Booch (2005 : 34)

Tabel II.2. Komponen Class Diagarm

3. *Activity Diagram.*

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti use caseatau interaksi. Dapat dilihat pada gambar II.6. berikut

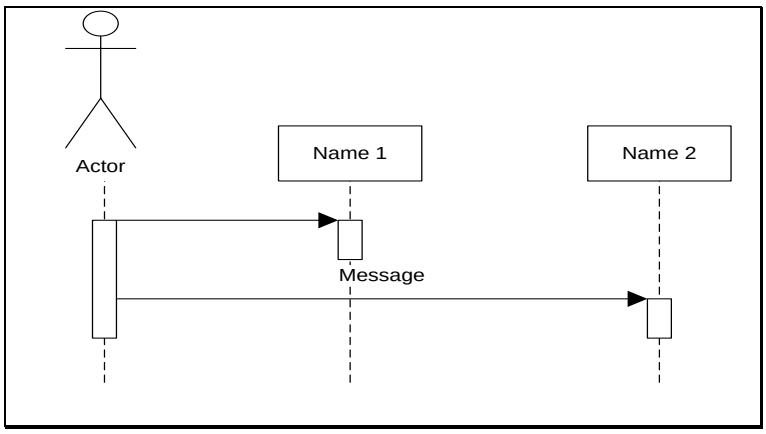


Gambar II.6. Activity diagram modifikasi

(Sumber : Munawar, 2005:112)

4. *Sequence Diagram.*

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara object juga interaksi antaraobject, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem. Dapat dilihat pada gambar II.7 berikut.



Gambar II.7. Simbol-simbol yang ada pada sequence diagram

(Sumber : Munawar, 2005:89)

Berikut ini merupakan komponen-komponen pada sequence diagram. Dapat dilihat pada tabel II.3 sebagai berikut.

Tabel II.3. Komponen Sequence Diagram

No.	Nama Komponen	Keterangan	Simbol
1	Object	Object merupakan instance dari sebuah class dan dituliskan tersusun secara horizontal. Digambarkan sebagai sebuah class (kotak) dengan nama object didalamnya yang diawali dengan sebuah titik koma.	
2	Actor	Actor juga dapat berkomunikasi dengan object , maka actor juga dapat diurutkan sebagai kolom. Simbol Actor sama dengan simbol pada Actor Use Case Diagram.	

Sumber : Grady Booch (2005 : 42)