

BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem dan Analisa Sistem

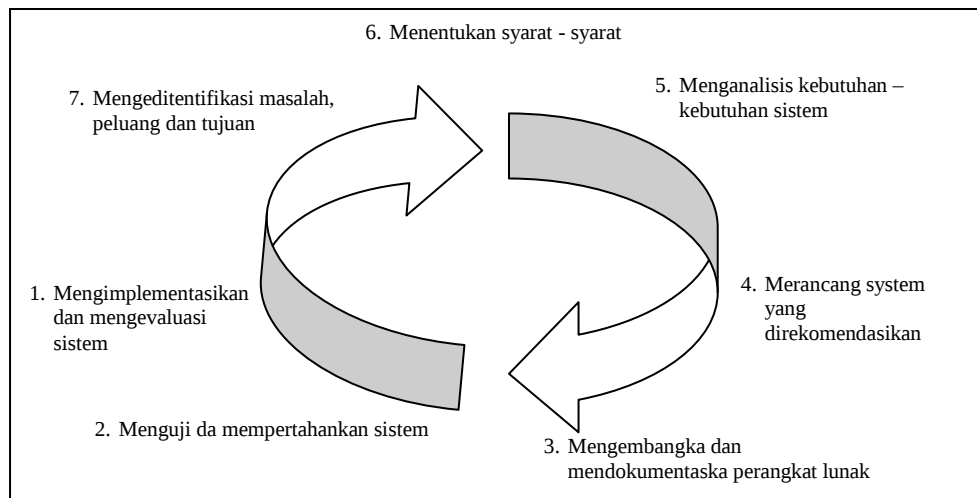
II.1.1. Pengertian Sistem

Sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksud untuk mencapai suatu tujuan (Abdul Kadir. 2009). Sebagai contoh, sistem kendaraan terdiri dari: komponen *starter*, komponen pengapian, komponen penggerak, komponen pengerem, komponen kelistrikan-spedometer, lampau dan lain-lain. Komponen-komponen tersebut diatas memiliki tujuan yang sama yaitu untuk membuat kendaraan tersebut bisa dikendarai dengan nyaman dan aman. Contoh lain yaitu sistem perguruan tinggi, yang terdiri dari dosen, mahasiswa, kurikulum, dan lain-lain. Sistem ini bertujuan untuk menghasilkan mahasiswa-mahasiswa yang memiliki kemampuan di bidang ilmunya. (Abdul Kadir ; 2009: 11).

II.1.2. Siklus Hidup Pengembangan Sistem

Siklus hidup pengembangan sistem ini adalah pengembangan melalui beberapa tahap untuk menganalisis dan merancang sistem yang dimana sistem tersebut telah di kembangkan dengan sangat baik melalui pengembangan siklus kegiatan penganalisisan dan pemakaian secara spesifik (Abdul Kadir ; 2008 : 11)

Adapun siklus hidup pengembangan sistem data pada gambar II.1 :

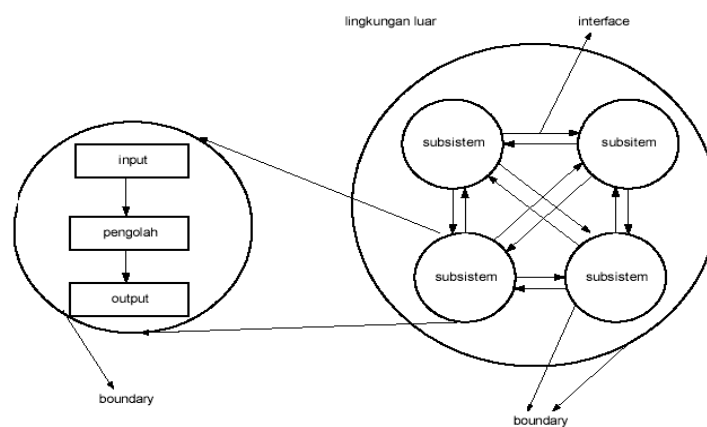


Gambar II.1: Siklus Hidup Pengembangan Sistem

Sumber : Abdul Kadir ; 2008 : 11

II.1.3. Karakteristik Sistem

Karakteristik sistem dapatlah digambarkan sebagai berikut :



Gambar II.2 : Karakteristik Sistem

Sumber : Jogiyanto ; 2005 : 3

Keterangan Gambar II.1 :

1. Komponen Sistem (*Components*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sistem tidak peduli betapapun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Jadi, dapat dibayangkan jika dalam suatu sistem ada subsistem yang tidak berjalan/berfungsi sebagaimana mestinya. Tentunya sistem tersebut tidak akan berjalan mulus atau mungkin juga sistem tersebut rusak sehingga dengan sendirinya tujuan sistem tersebut tidak tercapai.

2. Batas Sistem (*Boundary*)

Batas sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem (*Environments*)

Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem

dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. Sedang lingkungan luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan mengganggu kelangsungan hidup dari sistem.

4. Penghubung (*Interface*)

Penghubung merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke yang lainnya. Keluaran (output) dari satu subsistem akan menjadi masukan (input) untuk subsistem lainnya dengan melalui penghubung. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainnya membentuk satu kesatuan.

5. Masukan (*Input*)

Masukan adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran. Sebagai contoh didalam sistem komputer, program adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

6. Keluaran (*Output*)

Keluaran adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supersistem. Misalnya untuk sistem komputer, panas yang dihasilkan adalah keluaran yang tidak berguna dan merupakan hasil sisa pembuangan, sedang informasi adalah keluaran yang dibutuhkan.

7. Pengolah (*Process*)

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan-bahan yang lain menjadi keluaran berupa barang jadi. Sistem akuntansi akan mengolah data-data transaksi menjadi laporan-laporan keuangan dan laporan-laporan lain yang dibutuhkan oleh manajemen.

8. Sasaran (*Objectives*) atau Tujuan (*Goal*)

Suatu sistem pasti mempunyai tujuan atau sasaran. Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya. Perbedaan suatu sasaran (*objectives*) dan suatu tujuan (*goal*) adalah, *goal* biasanya dihubungkan dengan ruang lingkup yang lebih luas dan sasaran dalam ruang lingkup yang lebih sempit. Bila merupakan suatu

sistem utama, seperti misalnya sistem bisnis perusahaan, maka istilah *goal* lebih tepat diterapkan. Untuk sistem akuntansi atau sistem-sistem lainnya yang merupakan bagian atau subsistem dari sistem bisnis, maka istilah *objectives* yang lebih tepat. Jadi tergantung dari ruang lingkup mana memandang sistem tersebut. Seringkali tujuan (*goal*) dan sasaran (*objectives*) digunakan bergantian dan tidak dibedakan.

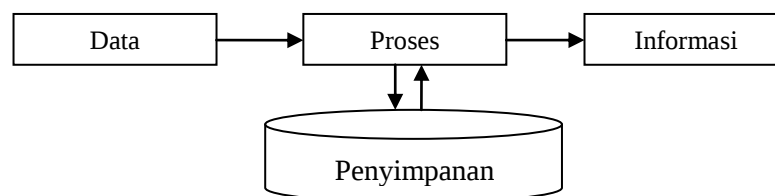
II.1.4. Pengertian Informasi

Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang (Davis, 1999). Jadi dengan adanya informasi, tingkat kepastian menjadi meningkat. Agar informasi yang penulis sajikan lebih bermanfaat maka terlebih dahulu dibuat aliran informasi yang lebih jelas dan lengkap. Berkaitannya dengan penyedia informasi bagi manajemen dalam mengambil suatu keputusan, yang diperoleh harus berkualitas, maka kualitas dari informasi tergantung pada.

1. Relevansi : relevansi berarti bahwa informasi benar – benar berguna bagi suatu tindakan dan keputusan oleh seseorang.
2. Ketepatan waktu : tepat waktu berarti bahwa informasi yang datang pada penerimaan tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi. Karena informasi merupakan landasan dalam pengambilan keputusan. Dewasa ini mahal nya nilai informasi disebabkan harus cepatnya informasi itu di dapat.

3. Keakurasian (Akurat) : akurat berarti bahwa informasi harus bebas dari kesalahan – kesalah dan tidak biasa (menyesatkan) dan jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerimaan informasi kemungkinan banyak terjadi gangguan yang dapat merubah informasi atau merusak informasi tersebut.

Untuk lebih jelasnya informasi merupakan hasil atau output dari proses informasi data. Hal ini dapat dilihat seperti gambar II.3 dibawah ini :



Gambar II.3. Proses Data Menjadi Informasi

Sumber : Fathansyah, Manajemen Sistem Informasi (2005: 9)

II.2. Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) adalah salah satu model sistem informasi yang banyak digunakan untuk membuat berbagai keputusan, perencanaan dan analisis. Kemampuan dari SIG menjelaskan di mana, bagaimana, dan apa yang akan terjadi secara keruangan yang diwujudkan dalam gambaran peta dengan berbagai penjelasan secara deskriptif, tabular, dan grafis. SIG juga di kenal sebagai sebuah rangkaian pengolahan data sistem yang memanfaatkan teknologi digital untuk melakukan analisa spasial yang mamfaatkan perangkat keras dan lunak dari komputer (Eko Budiyanto : 2006).

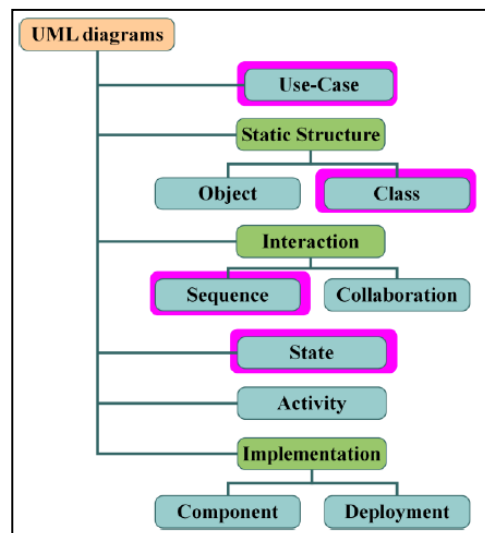
II.3. Alat Bantu Pengembangan Sistem

Alat bantu pengembangan sistem yang digunakan dalam penulisan skripsi ini adalah sebagai berikut :

II.3.1. UML (Unified Modelling Language)

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.

Secara filosofi UML diilhami oleh konsep yang telah ada yaitu konsep permodelan Object Oriented karena konsep ini menganalogikan sistem seperti kehidupan nyata yang didominasi oleh obyek dan digambarkan atau dinotasikan dalam simbol-simbol yang cukup spesifik. (Haviluddin : 2011)



Gambar II.4 : Diagram UML

Sumber : Haviluddin; 2011

II.3.2. Tujuan Pemanfaatan UML

Tujuan dari penggunaan diagram seperti diungkapkan oleh Schmuller J, *“The purpose of the diagrams is to present multiple views of a system; this set of multiple views is called a model”*. Berikut tujuan utama dalam desain UML adalah :

1. Menyediakan bagi pengguna (analisis dan desain sistem) suatu bahasa pemodelan visual yang ekspresif sehingga mereka dapat mengembangkan dan melakukan pertukaran model data yang bermakna.
2. Menyediakan mekanisme yang spesialisasi untuk memperluas konsep inti.
3. Karena merupakan bahasa pemodelan visual dalam proses pembangunannya maka UML bersifat independen terhadap bahasa pemrograman tertentu.
4. Memberikan dasar formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar terhadap penggunaan alat desain sistem yang berorientasi objek (OO).
6. Mendukung konsep pembangunan tingkat yang lebih tinggi seperti kolaborasi, kerangka, pola dan komponen terhadap suatu sistem.
7. Memiliki integrasi praktik terbaik.

II.3.3. Komponen – Komponen UML

Sejauh ini para pakar merasa lebih mudah dalam menganalisa dan mendesain atau memodelkan suatu sistem karena UML memiliki seperangkat aturan dan notasi dalam bentuk grafis yang cukup spesifik (Sugrue J. 2009).

Komponen atau notasi UML diturunkan dari 3 (tiga) notasi yang telah ada sebelumnya yaitu Grady Booch, OOD (Object-Oriented Design), Jim Rumbaugh, OMT (Object Modelling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering). (Haviluddin; 2011)

Pada UML versi 2 terdiri atas tiga kategori dan memiliki 13 jenis diagram yaitu :

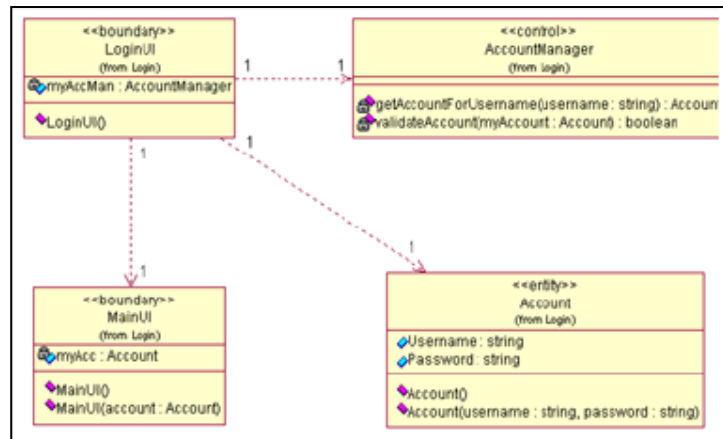
1. Struktur Diagram.

Menggambarkan elemen dari spesifikasi dimulai dengan kelas, obyek, dan hubungan mereka, dan beralih ke dokumen arsitektur logis dari suatu sistem. Struktur diagram dalam UML terdiri atas :

- a. Class diagram Class diagram menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas. Class diagram membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, class diagram berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat.

Class memiliki tiga area pokok :

- 1) Nama (*stereotype*)
- 2) Atribut
- 3) Metoda



Gambar II.5 : Notasi Class Diagram

Sumber : Haviluddin; 2011

b. *Object diagram*

Object diagram menggambarkan kejelasan kelas dan warisan dan kadang-kadang diambil ketika merencanakan kelas, atau untuk membantu pemangku kepentingan non-program yang mungkin menemukan diagram kelas terlalu abstrak.

c. *Component diagram*

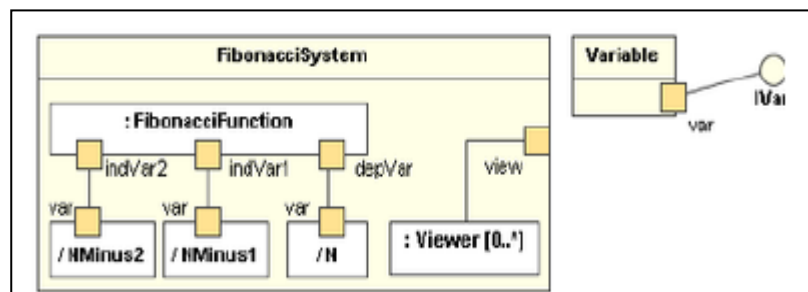
Component diagram menggambarkan struktur fisik dari kode, pemetaan pandangan logis dari kelas proyek untuk kode aktual di mana logika ini dilaksanakan.

d. *Deployment diagram (Collaboration diagram in version 1.x)*

Deployment diagram memberikan gambaran dari arsitektur fisik perangkat lunak, perangkat keras, dan artefak dari sistem. Deployment diagram dapat dianggap sebagai ujung spektrum dari kasus penggunaan, menggambarkan bentuk fisik dari sistem yang bertentangan dengan gambar konseptual dari pengguna dan perangkat berinteraksi dengan sistem.

e. *Composite structure diagram*

Sebuah diagram struktur komposit mirip dengan diagram kelas, tetapi menggambarkan bagian individu, bukan seluruh kelas. Kita dapat menambahkan konektor untuk menghubungkan dua atau lebih bagian dalam atau ketergantungan hubungan asosiasi.



Gambar II.6 : Notasi Composite Diagram.

Sumber : Haviluddin : 2011

f. *Package diagram*

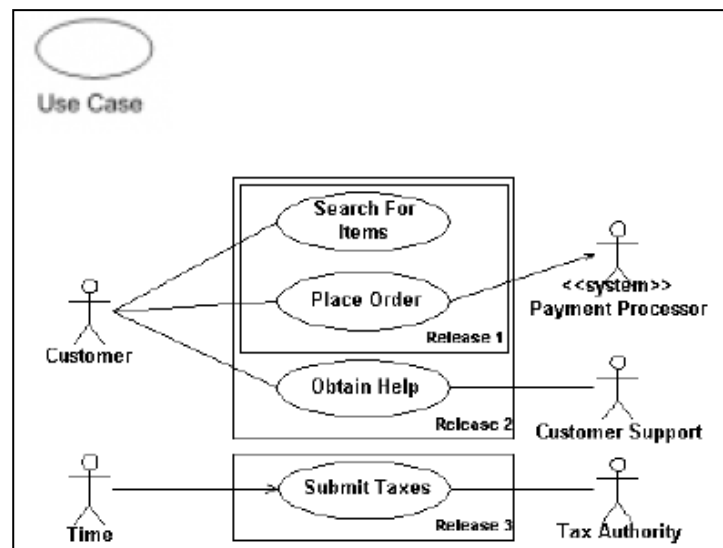
Paket diagram biasanya digunakan untuk menggambarkan tingkat organisasi yang tinggi dari suatu proyek software. Atau dengan kata lain untuk menghasilkan diagram ketergantungan paket untuk setiap paket dalam Pohon Model.

2. Behavior Diagram

Menggambarkan ciri-ciri *behavior*/metode/ fungsi dari sebuah sistem atau *business process*. Behavior diagram dalam UML terdiri atas :

a. Use case diagram

Diagram yang menggambarkan actor, use case dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah use case digambarkan sebagai elips horizontal dalam suatu diagram UML use case.

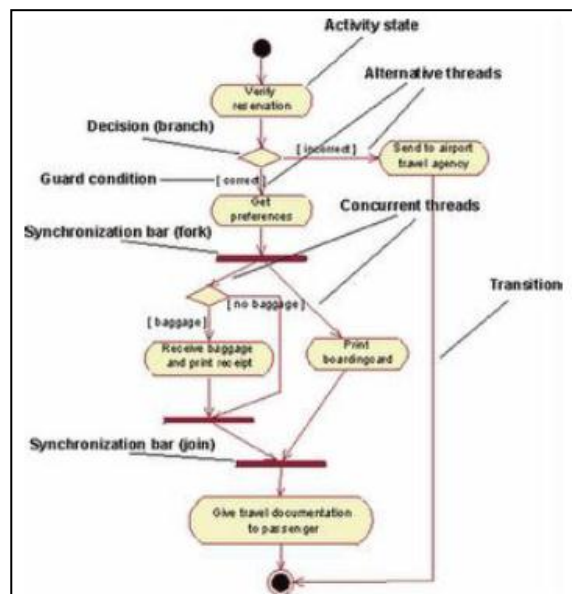


Gambar II.7 : Use Case Diagram

Sumber : Haviluddin : 2011

b. Activity diagram

Menggambarkan aktifitas-aktifitas, objek, state, transisi state dan event. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas



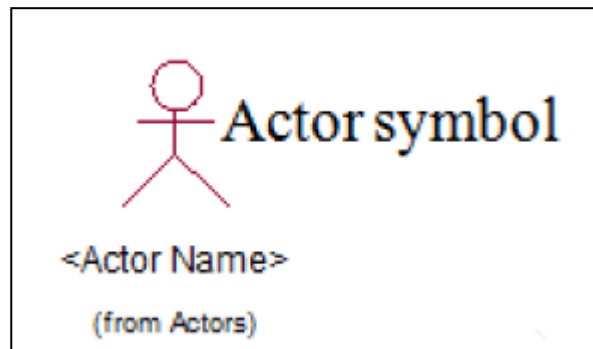
Gambar II.8 : Notasi Activity Diagram
Sumber : Haviluddin : 2011

c. State Machine diagram (State chart diagram in version 1.x)

Menggambarkan state, transisi state dan event.

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 :

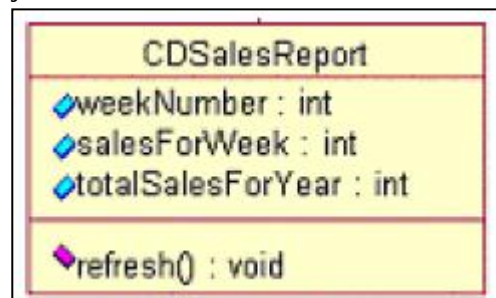
1. Actor; menentukan peran yang dimainkan oleh user atau sistem lain yang berinteraksi dengan subjek. Actor adalah segala sesuatu yang berinteraksi langsung dengan sistem aplikasi komputer, seperti orang, benda atau lainnya. Tugas actor adalah memberikan informasi kepada sistem dan dapat memerintahkan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



Gambar II.9 : Notasi 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.



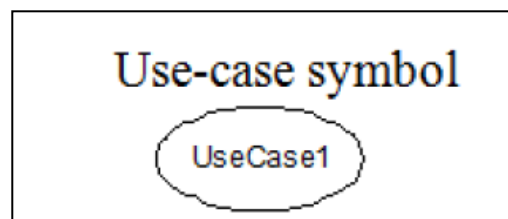
Gambar II.10 : Notasi Class

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, yaitu kebutuhan fungsional sistem dan tidak untuk menentukan kebutuhan non-fungsional, misalnya: sasaran kinerja, bahasa pemrograman dan lain sebagainya.



Gambar.II.11. Notasi 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.
5. *Interaction*; *Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek.
6. *Dependency*; *Dependency* merupakan relasi yang menunjukkan 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.

7. *Package*; *package* adalah mekanisme pengelompokkan yang digunakan untuk menandakan pengelompokkan elemen-elemen model.

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

II.3.4. Flowchart

Flowchart atau diagram alir adalah sekumpulan simbol-simbol atau skema yang menunjukkan atau menggambarkan rangkaian kegiatan-kegiatan mulai dari awal hingga akhir. (Abdul Kadir, 2008 : 83)

II.4. Sistem Database

Database merupakan bagian yang paling tinggi dari sistem data. *Database* adalah kumpulan beberapa *file-file* yang mempunyai kaitan antara satu *file* dengan *file* yang lain sehingga membentuk suatu data. Sedangkan sistem *database* adalah suatu landasan atau kerangka dasar yang terdiri dari sistem dan sub sistem yang digunakan untuk melayani aplikasi tertentu guna mencapai tujuan. Manajemen sebuah sistem *database* adalah suatu sistem perangkat lunak komputer. (Bunafit Nugroho, 2009 : 6)

Sebelum merancang sebuah *database* perlu untuk memahami data, untuk itu harus mengetahui apa saja komponen dasar yang berhubungan dengan *database*, yaitu :

1. *Enterprise*

Merupakan elemen data atau objek data dalam bentuk organisasi, lembaga atau perusahaan.

2. *Entity*

Merupakan segala sesuatu yang membentuk *enterprise* atau unsur yang membentuk *enterprise*. *Entity* dapat berupa orang, benda, tempat, peristiwa yang dapat memberikan informasi.

3. *Attribute*

Merupakan semua informasi yang menerangkan suatu *entity*. Setiap *entity* mempunyai *attribute* untuk mewakili suatu *entity*. *Attribute* disebut juga elemen data *field* dan data item.

4. *Data Key*

Merupakan elemen yang ditentukan sebagai kunci untuk mendapatkan data elemen suatu *entity*.

5. *Data Value*

Merupakan nilai akurat yang diisikan pada suatu elemen.

6. *Record*

Merupakan kumpulan elemen yang saling berkaitan dan menginformasikan tentang suatu *entity* tertentu.

7. File

Merupakan kumpulan dari beberapa *record* yang sejenis.

Database management system atau sistem pengolahan data secara fisik tidak dilakukan oleh pemakai secara langsung tetapi dilayani oleh perangkat lunak (*software*) khusus. Dengan menggunakan suatu sistem manajemen *database* dapat menempatkan *database* dalam suatu *file* dan mengorganisasikannya menjadi tabel dan mengkoordinir data tersebut menjadi laporan. Jenis perangkat lunak yang termasuk DBMS adalah *dbase III+*, *dbse IV*, *Fox Base*, *MS-Access*, *Borland Interbase*, dan *My SQL Server*.

II.5. MySQL

MySQL (My Structure Query Language) adalah sebuah perangkat lunak sistem manajemen basis data *SQL (Database Management System)* atau DBMS dari sekian banyak DBMS, seperti *Oracle*, *MS SQL*, *Porstagre SQL*, dan lain-lain. *MySQL* merupakan DBMS yang multithread, multi *user* yang bersifat gratis dibawah lisensi *GNU General Public Licence (GPL)*. Tidak seperti *Apache* yang merupakan *software* yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki penulisnya masing-masing (Anhar, 2010: 21).

MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia *MySQL AB*, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL AB* adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius. (Yeni Kustiyahningsih; 2011 : 145)

II.5.1. Sistem Manajemen Basis Data Relasional

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). 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 sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, *MySQL* mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, *MySQL* dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (*wordpress*), CMS, dan sejenisnya. (Yeni Kustiyahningsih; 2011 : 152)

II.5.2. Kelebihan MySQL

MySQL memiliki beberapa keistimewaan, antara lain :

1. MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
2. Perangkat lunak sumber terbuka. MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
3. Multi-user. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. '*Performance tuning*', MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. Ragam tipe data. MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double, char, text, date, timestamp*, dan lain-lain.
6. Perintah dan Fungsi. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).
7. Keamanan. MySQL memiliki beberapa lapisan keamanan seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.

8. Skalabilitas dan Pembatasan. *MySQL* mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. Konektivitas. *MySQL* dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, *Unix socket* (UNIX), atau *Named Pipes* (NT).
10. Lokalisasi *MySQL* dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. Antar Muka. *MySQL* memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).
12. Klien dan Peralatan. *MySQL* dilengkapi dengan berbagai peralatan (*tool*) yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.
13. Struktur tabel. *MySQL* memiliki struktur tabel yang lebih fleksibel dalam menangani *Alter Table*, dibandingkan basis data lainnya semacam *PostgreSQL* ataupun *Oracle*.

Berikut beberapa fungsi *MySQL* yang sering digunakan beserta deskripsi singkat dari fungsi-fungsi tersebut :

1. *MySQL_conneccct ()*

Sebelum melakukan operasi apapun pada *MySQL*, hal pertama yang harus dilakukan adalah membuka koneksi dengan *MySQL*. Untuk menjalankan fungsi tersebut, maka kita harus menjalankan fungsi *MySQL_connect()* dengan mengetikkan *hostname*, *username* dan *password*.

2. *MySQL_create_db ()*

Untuk membuat *database* yang hanya dapat diakses oleh *username* dan *password*, digunakan fungsi *MySQL_create_db()* setelah koneksi ke *MySQL* dibuka.

3. *MySQL_query ()*

Fungsi *MySQL_query()* digunakan untuk mengirim *query* kepada *database* aktif yang sudah dipilih.

4. *MySQL_fetch_row ()*

Digunakan untuk mengambil hasil *query* dari *database* sebagai data numerik *array*. Fungsi ini mengambil data baris per baris (*record*) dari *database* secara berulang-ulang sampai kondisi *query* menjadi *false*.

5. *MySQL_fetch_array ()*

Digunakan untuk mengambil hasil *query* dari *database* sebagai data *array* asosiatif.

6. *MySQL_num_rows ()*

Digunakan untuk menghitung jumlah baris hasil *query*

II.6. *PHP*

PHP merupakan singkatan dari *PHP Hypertext Preprocessor* yang merupakan bahasa berbentuk skrip yang di tempatkan dalam server dan diproses di server. *PHP* secara khusus di rancang untuk membentuk aplikasi web dinamis. Artinya ia dapat membentuk suatu tampilan berdasarkan permintaan terkini (Abdul Kadir, 2008).

Contoh terkenal dari aplikasi *PHP* adalah *PHPBB* dan *MediaWiki* (software di belakang *Wikipedia*). *PHP* juga dapat dilihat sebagai pilihan lain dari *ASP.NET/C#/VB.NET Microsoft*, *ColdFusion Macromedia*, *JSP/Java Sun Microsystems*, dan *CGI/Perl*. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan *PHP* adalah *Mambo*, *Joomla!*, *Postnuke*, *Xaraya*, dan lain-lain. (Yeni Kustiyahningsih; 2011 : 113)

II.6.1. Sejarah *PHP*

Pada awalnya *PHP* merupakan kependekan dari *Personal Home Page* (Situs Personal). *PHP* pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu *PHP* masih bernama *FI (Form Interpreted)*, yang wujudnya berupa sekumpulan script yang digunakan untuk mengolah data *form* dari web.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya *PHP/FI*. Dengan perilsan kode sumber ini menjadi *open source*, maka banyak *programmer* yang tertarik untuk ikut mengembangkan *PHP*.

Pada November 1997, dirilis *PHP/FI 2.0*. Pada rilis ini *interpreter PHP* sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan *PHP/FI* secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang *interpreter PHP* menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis *interpreter* baru untuk *PHP* dan meresmikan rilis tersebut sebagai *PHP 3.0* dan singkatan *PHP* dirubah menjadi akronim berulang *PHP: Hypertext Preprocessing*. Pada pertengahan tahun 1999, Zend merilis *interpreter PHP* baru dan rilis tersebut dikenal dengan *PHP 4.0*. *PHP 4.0* adalah versi *PHP* yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis *PHP 5.0*. Dalam versi ini, inti dari *interpreter PHP* mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam *PHP* untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

II.6.2. Kelebihan *PHP*

1. Bahasa pemrograman *PHP* adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. *Web Server* yang mendukung *PHP* dapat ditemukan dimana - mana dari mulai *apache*, *IIS*, *Lighttpd*, *nginx*, hingga *Xitami* dengan konfigurasi yang relatif mudah.

3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahaman, *PHP* adalah bahasa scripting yang paling mudah karena memiliki referensi yang banyak.
5. *PHP* adalah bahasa open source yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah sistem.

II.6.3. Konsep Dasar *PHP*

PHP adalah bahasa *script* yang menyatu dengan HTML dan dijalankan pada *server side*. Artinya semua sintaks yang kita berikan akan sepenuhnya akan dijalankan pada *server* sedangkan yang dikirimkan ke *browser* hanya hasilnya saja.

Untuk menuliskan dan memperkenalkan *PHP*, anda harus memulainya dengan tanda `<?PHP`, setelah tanda tersebut dilanjutkan dengan kode program di dalamnya. Untuk mengakhiri kode program yang dibuat, dapat ditutup dengan tanda `?>`.

II.7. Website

Website atau situs juga dapat diartikan sebagai kumpulan halaman yang menampilkan informasi data teks, data gambar diam atau gerak, data animasi, suara, video dan atau gabungan dari semuanya, baik yang bersifat statis maupun

dinamis yang membentuk satu rangkaian bangunan yang saling terkait dimana masing-masing dihubungkan dengan jaringan-jaringan halaman (*hyperlink*). Bersifat statis apabila isi informasi *website* tetap, jarang berubah, dan isi informasi searah *web* dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain. (Yeni Kustiyahningsih; 2011 : 4)

II.8. ArcView

ArcView merupakan salah satu perangkat lunak desktop Sistem Informasi Geografis dan pemetaan yang telah di kembangkan oleh ESRI. Dengan *ArcView*, pengguna dapat memiliki kemampuan-kemampuan untuk melakukan visualisasi, meng-explore, menjawab query (baik basis data spasial maupun non-spasial), menganalisa data secara geografis, dan sebagainya (Eddy Prahasta, 2009).

Secara umum kemampuan *ArcView* dapat dilihat melalui uraian berikut :

1. Pertukaran data, membaca dan menuliskan data dari dan ke dalam format perangkat lunak SIG lainnya.
2. Melakukan analisis statistik dan operasi-operasi matematis.
3. Menampilkan informasi (basisdata) spasial maupun atribut.
4. Menghubungkan informasi spasial dengan atribut-atributnya yang terdapat (disimpan) dalam basisdata atribut.
5. Melakukan fungsi-fungsi dasar SIG seperti analisis sederhana spasial.
6. Membuat peta tematik.
7. Meng-*customize* aplikasi dengan menggunakan bahasa skrip atau bahasa pemrograman sederhana.

8. Melakukan fungsi-fungsi SIG khusus lainnya (dengan menggunakan *extension* yang ditujukan untuk mendukung penggunaan perangkat lunak SIG *ArcView*).

ArcView mengorganisasikan sistem perangkat lunaknya sedemikian rupa sehingga dapat dikelompokkan ke dalam beberapa komponen penting sebagai berikut:

1. *Project*

Project merupakan suatu unit organisasi tertinggi di dalam *ArcView*. *Project*, di dalam *ArcView*, mirip *project* yang dimiliki oleh bahasa-bahasa pemrograman komputer (C/C++, Pascal/Delphi, Basic dan sebagainya), atau paling tidak merupakan suatu file kerja yang dapat digunakan untuk menyimpan, mengelompokkan dan mengorganisasikan semua komponen-komponen program: *view*, *theme*, *table*, *chart*, *layout*, dan *script* dalam satu kesatuan yang utuh.

2. *Theme*

Themes merupakan suatu bangunan dasar sistem *ArcView*. *Themes* merupakan kumpulan dari beberapa layer *ArcView* yang membentuk suatu tematik tertentu. Sumber data yang dapat direpresentasikan sebagai theme adalah *shapfile*, *coverage* (ArcInfo) dan citra raster.

3. *View*

View mengorganisasikan theme. Sebuah *view* merupakan representasi grafis informasi spasial dan dapat menampung beberapa *layer* atau *theme* informasi spasial (titik, garis, poligon dan citra raster). Sebagai contoh, posisi-

posisi kota (titik), sungai-sungai (garis), dan batas propinsi (poligon) dapat membentuk theme dalam sebuah view.

4. *Table*

Sebuah tabel merupakan representasi data *ArcView* dalam bentuk sebuah tabel. Sebuah tabel akan berisi informasi deskriptif mengenai layer tertentu. Setiap baris data (*record*) mendefinisikan sebuah entry (misalnya informasi mengenai salah satu poligon batas propinsi) di dalam basis data spasialnya. Setiap kolom (*field*) mendefinisikan atribut atau karakteristik dari entri (misalnya nama, luas, keliling, atau populasi suatu propinsi) yang bersangkutan. Dari sisi pengguna, tanpa memperhatikan sumber-sumbernya, semua tabel adalah sama. *ArcView* mendefinisikan *template standart* untuk merujuk tabel yang diakses.

5. *Chart*

Chart merupakan representasi grafis dari *resume* tabel data. Chart juga merupakan hasil suatu *query* terhadap suatu tabel data. Bentuk chart yang didukung oleh *ArcView* adalah *line*, *bar*, *column*, *xy scatter*, *area*, dan *pie*.

6. *Layout*

Layout digunakan untuk menggabungkan semua dokumen (*view*, *tabel* dan *chart*) ke dalam suatu dokumen yang siap cetak (biasanya dipersiapkan untuk pembuatan *hardcopy*).

7. *Script*

Script merupakan bahasa (semi) pemrograman sederhana (makro) yang digunakan untuk mengotomasi kerja *ArcView*. *ArcView* menyediakan bahasa sederhana ini dengan sebutan *Avenue*. Dengan *Avenue*, pengguna dapat

memodifikasi tampilan (*user interface*) *ArcView*, membuat program, menyederhanakan tugas-tugas yang kompleks, dan berkomunikasi dengan aplikasi-aplikasi lain (misalnya dengan *ArcInfo*, basis data *relational* atau lembar kerja elektronik). Singkatnya, dengan *script*, *ArcView* dapat *dicustomized* sedemikian rupa hingga dapat secara optimal memenuhi kebutuhan pengguna untuk tugas-tugas dan aplikasi tertentu (Eddy Prahasta, 2009: 5-7).

II.8.1. Kelebihan *ArcView*

Kelebihan penggunaan software *Arcview GIS* dalam dunia pemetaan atau kartographic adalah dengan terdapatnya fasilitas *tools provide* yang komplet. Sistem mampu mengkreasikan kualitas peta yang prima, dimana user dapat berkreasi dan memperkaya detail detail yang diperlukan untuk lebih atraktif, dan mampu secara efektif menampilkan peta dan informasi lebih komunikatif yang di dapat dari hasil analisa data.

ArcView diperkaya dengan seting dan komposisi *tools* dan *wizard*, yang mempercepat mempresentasikan pekerjaan terutama bidang pemetaan. Lingkup pekerjaan pemetaan mempunyai variasi symbol dan warna juga template yang dapat di pergunakan setiap saat. Kelebihan *ArcView GIS* adalah komposisi dari tool nya yang mampu melakukan features dan digitasi on screen yang dapat dipergunakan untuk membedakan satu obyek dengan obyek lainnya, yakni dengan mempergunakan color ramp, shading data, graduate symbol, chart symbol, untuk kreasi dan memperindah bentuk tampilan peta lebih berkualitas.

Kelebihan lain dari Arcview adalah kemampuannya dalam melakukan exporting data yang bisa dibaca dalam berbagai macam fasilitas software lainnya antara lain *Write/Export data to : ESRI shapefiles, DXF, placeable WMF, Windows metafile, Windows bitmap, PostScript, CGM (binary, character, clear text), and JPEG. Extended data support from optional ArcView GIS extensions TIN, GRID (as raster data), DOQ, IRS-1C, Landsat TM, RPF, SPOT, and GeoSpot.*