BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem merupakan kumpulan dari unsur-unsur yang saling berkaitan, berinteraksi / berhubungan, dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai tujuan tertentu.

Menurut Ludwig Von Bartalanfy, sistem merupakan seperangkat unsur yang saling terikat dalam suatu antar relasi di antara unsur-unsur tersebut dengan lingkungan.

Direktur Sistem Informasi
Manajer Pengembangan Sistem
Analis Sistem
Programmer
Manajer Komputer dan Operasi

Gambar II.1 : Struktur Manajemen Sistem Informasi

(Sumber : Asbon Hendra, 2012)

II.1.1. Syarat-Syarat Sistem

1. Sistem harus dibentuk untuk menyelesaikan tujuan.
2. Adanya hubungan di antara elemen sistem.
3. Elemen sistem harus mempunyai rencana yang ditetapkan.

4. Unsur dasar dari proses (arus informasi, energi dan material) lebih penting daripada elemen sistem.
5. Tujuan organisasi lebih penting dari pada tujuan elemen.

II.1.2. Karakteristik Sistem

1. Komponen (*Component*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, dan bekerja sama membentuk satu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai proses sistem yang lebih besar yang disebut *supra system*.

2. Batas Sistem (*Boundary*)

Merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan, karena dengan batas sistem ini fungsi dan tugas dari subsistem yang satu dengan yang lain berbeda tetapi tetap saling berinteraksi. Batas suatu sistem menunjukkan ruang lingkup dari sistem tersebut.

3. Lingkungan Luar Sistem (*Environment*)

Merupakan segala sesuatu yang berada di luar batas sistem yang mempengaruhi operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan dan merugikan. Lingkungan luar yang

menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan atau dikendalikan agar tidak mengganggu operasi sistem.

4. Penghubung Sistem (*Interface*)

Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya untuk membentuk satu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain, *output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainnya.

5. Masukan Sistem (*Input*)

Merupakan energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan, yaitu energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Masukan sinyal adalah energi yang diproses untuk didapatkan keluaran.

6. Keluaran Sistem (*Output*)

Merupakan hasil dari energi yang diolah oleh sistem, meliputi *output* yang berguna, contohnya informasi yang dikeluarkan oleh komputer. Dan *output* yang tidak berguna dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan komputer.

7. Pengolah Sistem (*Process*)

Pengolah sistem merupakan bagian yang memproses masukan untuk menjadi keluaran yang diinginkan contoh *CPU* pada komputer.

8. Tujuan Sistem (*Goal*)

Setiap sistem mempunyai tujuan yang mempengaruhi *input* yang dibutuhkan dan *output* yang dihasilkan. Dengan kata lain, suatu sistem itu mengenai tujuannya. Jika sistem tidak mempunyai tujuan, maka operasi sistem tidak akan ada gunanya.

II.1.3. Klasifikasi Sistem

1. Sistem Abstrak (*Abstract System*)

Sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sebagai contoh sistem Teologia yang merupakan suatu sistem yang menggambarkan hubungan Tuhan dengan manusia.

2. Sistem Fisik (*Physical System*)

Sistem yang ada secara fisik sehingga setiap makhluk dapat melihatnya. Contohnya sistem komputer, sistem akuntansi, sistem produksi, dan lain-lain.

3. Sistem Alamiah (*Natural System*)

Sistem yang terjadi melalui proses alam. Dalam artian tidak dibuat oleh manusia. Seperti sistem tata surya, sistem galaksi, sistem reproduksi, dan lain-lain.

4. Sistem Buatan Mansuia (*Human Made System*)

Sistem yang dirancang oleh manusia. Sistem buatan manusia yang melibatkan interaksi manusia dengan mesin disebut *human machine system*. Contohnya sistem informasi.

5. Sistem Tertentu (*Deterministic System*)

Beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi bagian-bagiannya dapat dideteksi dengan pasti sehingga keluaran dari sistem dapat diramalkan. Contohnya sistem komputer.

6. Sistem Tak Tentu (*Probabilistic System*)

Sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas. Contohnya sistem manusia.

7. Sistem Tertutup (*Closed System*)

Sistem yang tidak berhubungan dan tidak terpengaruh dengan sistem luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak luarnya. Secara teori, sistem tersebut ada. Tetapi pada kenyataannya tidak ada sistem yang benar-benar tertutup. Yang ada hanyalah relatif tertutup atau tidak benar-benar tertutup (*relatively closed system*).

8. Sistem Terbuka (*Open System*)

Sistem yang berhubungan dan terpengaruh dengan lingkunan luarnya. Lebih spesifik dikenal juga yang desebut dengan sistem terotomasi, yang merupakan bagian dari sistem buatan manusia dan berinteraksi dengan kontrol oleh satu atau lebih komputer sebagai bagian dari sistem yang digunakan dalam masyarakat modern. (Asbon Hendra;2012: 157-162).

II.2. Informasi

Informasi merupakan data yang telah diproses menjadi bentuk yang memiliki arti bagi penerima dan dapat berupa fakta, suatu nilai yang bermanfaat. Jadi, ada suatu proses transformasi data menjadi suatu informasi = *input* – proses – *output.*

Data merupakan bahan baku (*raw material*) untuk suatu informasi. Perbedaan informasi dan data sangat relatif, tergantung pada nilai gunanya bagi manajemen yang memerlukan. Suatu informasi bagi level manajemen tertentu bisa menjadi data bagi manajemen level di atasnya, atau sebaliknya. (Asbon Hendra; 2012 : 167).

II.3. Sistem Informasi Geografis

Geografi adalah ilmu tentang lokasi serta persamaan dan perbedaan (variasi) keruangan atau fenomena fisik dan manusia diatas permukaan bumi. Kata geografi berasal dari Bahasa Yunani yaitu '*ge*' (bumi) dan '*graphein*' (menulis / menjelaskan). Geografi tidak hanya menjawab apa dan di mana di atas muka bumi, tapi juga mengapa di situ dan di tempat lainnya, kadang diartikan dengan "lokasi pada ruang", geografi mempelajari hal ini, baik yang disebabkan oleh alam atau manusia. Juga mempelajari akibat yang disebabkan dari perbedaan yang terjadi, istilah ini digunakan karena GIS dibangun berdasarkan pada geografi atau spasial.

Objek ini mengarah pada spesifikasi lokasi dalam suatu space. Objek bisa berupa fisik, budaya atau ekonomi alamiah. Penampakan tersebut ditampilkan

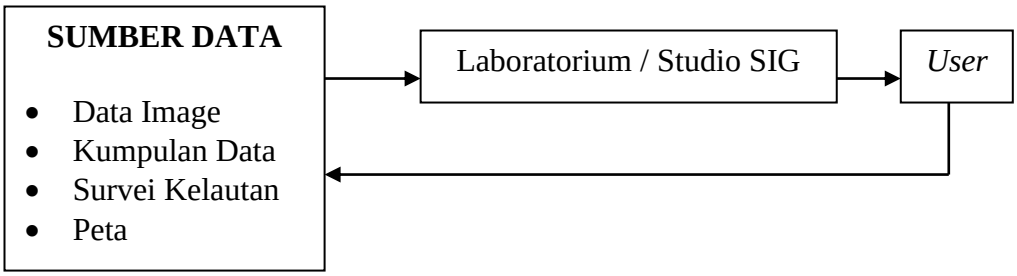
pada suatu peta untuk memberikan gambaran yang representatif dari spasial suatu objek sesuai dengan kenyataannya dibumi. Simbol, warna dan gaya garis digunakan untuk mewakili setiap spasial yang berbeda pada peta dua dimensi. (Eko Budianto; 2009 : 1)

Sistem Informasi Berbasis Pemetaan dan Geografis adalah sebuah alat bantu manajemen berupa informasi berbantuan komputer berkait erat dengan sistem pemetaan dan analisis terhadap segala sesuatu serta peristiwa-peristiwa yang terjadi dipermukaan bumi. Teknologi GIS mengintergrasikan operasi pengolahan data berbasis database yang digunakan saat ini pengambilan data berdasarkan kebutuhan, serta analisa statistik dengan menggunakan visualisasi yang khas berbagai keuntungan yang mampu ditawarkan melalui analisis geografi melalui gambar-gambar petanya. Teknologi Sistem Informasi Geografis dapat digunakan untuk investigasi ilmiah, pengolahan sumber daya perencanaan pembangunan, kartografi dan perencanaan rute, misalnya, SIG bisa membantu perencana untuk secara cepat menghitung waktu tanggap darurat saat terjadi bencana alam, atau SIG dapat digunakan untuk mencari lahan basah (*wetlands)* yang membutuhkan perlindungan dari polusi. (Eko Budianto; 2009 : 6)

II.3.1. Konsep SIG

Sumber data untuk keperluan SIG dpat berasal dari citra, data lapangan, survey kelautan, peta, sosial ekonomi dan GPS. Selnjutnya diolah di laboratorium atau studio SIG dengan *software* tertentu sesuai dengan kebutuhannya untuk

menghasilkan produk berupa informasi yang berguna, bisa berupa peta konvensional, maupun peta digital sesuai keperluan user.



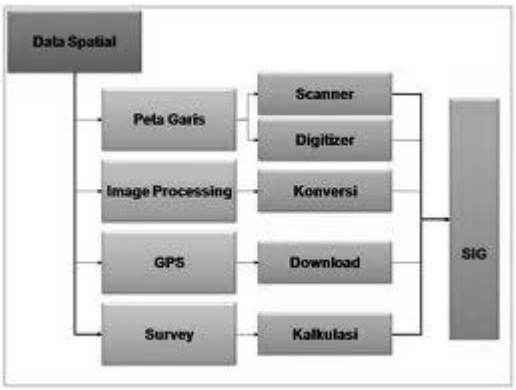
Gambar II.2 : Contoh Konsep SIG

(Sumber : Eko Budiyanto, 2009)

Untuk mendukung suatu sistem informasi geografis, pada prinsipnya terdapat dua jenis data, yaitu :

1. Data Spasial

   Data yang berkaitan dengan aspek keruangan dan merupakan data yang menyajikan lokasi geografis atau gambaran nyata suatu tempat/wilayah dipermukaan bumi. Umumnya direpresentasikan berupa grafik, peta, ataupun gambar dengan format digital dan disimpan dalam bentuk koordinat x,y (vektor) atau dalam bentuk image (raster) yang memiliki nilai tertentu.

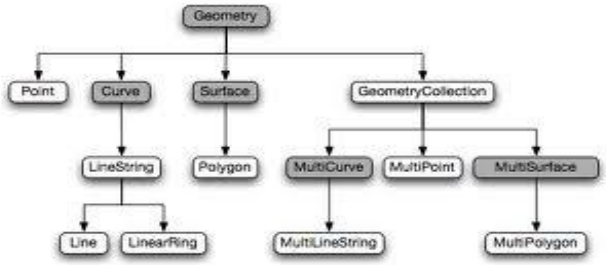


Gambar II.3 : Contoh Data Spasial

(Sumber : Eko Budiyanto, 2009)

2. Data Non Spasial

Disebut juga data atribut, yaitu data yang menerangkan keadaan atau informasi dari suatu objek (lokasi dan posisi) yang ditunjukkan oleh data spasial. Salah satu komponen utama dari sistem informasi geografis adalah perangkat lunak. Perangkat lunak berfungsi sebagai alat yang dapat membantu dalam memvisualisasikan, mengeksplorasi, menjawab *query* dan menganalisis data secara geografis. (Eko Budianto; 2009 : 10)



Gambar II.4 : Contoh Data Non Spasial

(Sumber : Eko Budiyanto, 2009)

II.3.2. Komponen Sistem Informasi Geografis

Secara umum, Sistem Informasi Geografis bekerja berdasarkan integrasi komponen, yaitu :

1. *Hardware*

Sistem Informasi Geografis (SIG) memerlukan spesifikasi komponen *hardware* yang sedikit lebih tinggi dibanding spesifikasi komponen sistem informasi lainnya. Hal tersebut disebabkan karena data-data yang digunakan dalam SIG, penyimpanannya membutuhkan ruang yang besar dan dalam proses analisanya membutuhkan memori yang besar dan

prosesor yang cepat. Beberapa *hardware* yang sering digunakan yaitu : *personal computer*, *mouse*, *digitilizer*, *printer*, *plotter*, dan *scanner*.

2. *Software*

Sebuah *software* SIG haruslah menyediakan fungsi dan *tool* yang mampu melakukan penyimpanan data, analisis, dan menampilkan informasi geografis.

3. Data

Hal yang merupakan komponen penting dalam SIG adalah data. Secara fundamental, SIG bekerja dengan 2 tipe model data geografis, yaitu model data vektor dan model data raster. Dalam model data vektor, informasi posisi *point*, garis, dan poligon disimpan dalam bentuk koordinat x,y. Data raster terdiri dari sekumpulan *grid* atau sel seperti peta hasil *scanning* maupun gambar. Masing-masing *grid* memiliki nilai tertentu yang bergantung pada bagaimana gambar tersebut digambarkan.

4. Manusia

Komponen manusia memegang peranan yang sangat menentuka. Karena tanpa manusia maka sistem tersebut tidak dapat diaplikasikan dengan baik. Jadi manusia menjadi komponen yang mengendalikan suatu sistem sehingga menghasilkan suatu analisa yang dibutuhkan.

5. Metode

Metode yang digunakan dalam SIG akan berbeda untuk setiap permasalahan. SIG yang baik tergantung pada aspek desain dan aspek *real*nya. (Eko Budianto; 2009 : 10)

II.4. ArcView

ArcView adalah *software* sistem informasi geografis (SIG). *Software* SIG mempunyai kemampuan untuk menampilkan, memanipulasi dan merubah data SIG. Data SIG mempunyai dua komponen, yaitu komponen spasial atau geografis dan komponen atribut atau table. Data spatial menampilkan lokasi geografis dari suatu *features.* Pada umumnya *features* tersebut ditampilkan dalam bentuk titik (*point),* garis (*line),* polygon (*polygon).* Seperti gambar di bawah :

Gambar II.5 : Contoh *ArcView*

(Sumber : Eko Budiyanto, 2009)

ArcView memiliki struktur dan istilah yang harus dipelajari dan dipahami agar dapat mempermudah pekerjaan kita dalam mengolah data SIG dengan menggunakan *ArcView*. Beberapa struktur dan istilah dalam *ArcView* GIS, yaitu :

1. *ArcView Project*

   File *ArcView* Project (*.apr) mengandung sebuah set perintah yang menjelaskan bagaimana tampilan data *ArcView* dan bagaimana data tersebut harus ditampilkan.

2. *Views*

View adalah sebuah workspace yang dapat menganalisis data, memanipulasi data dan menampilkan data. *Layer-layer* yang terdapat pada peta disebut themes, themes ditampilkan disisi kiri workspace, list tersebut disebut dengan *table of content* (TOC).

3. *Tables*

Table merupakan representasi data *ArcView* yang menampilkan data tabular. *Table* menyajikan informasi deskriptif yang menjelaskan *feature-feature* tentang *layer* tertentu pada suatu *view*.

4. *Chart*

Chart menampilkan data tabular secara visual dalam bentuk grafik. *Chart* juga bisa merupakan hasil dari suatu *querry* terhadap tabel data.

5. *Layout*

Menyediakan teknik-teknik untuk menggabungkan dan menyusun dokumen-dokumen dalam *project* (*view*, *table*, *chart*) dan komponen peta lainnya seperti arah utara dan skala guna menciptakan peta akhir untuk dicetak atau diplot.

6. *Scripts*

Script merupakan bahasa (semi) pemograman sederhana (makro) yang digunakan untuk otomatisasi kerja *ArcView*.

7. *Active*

Active merupakan *themes active* yang akan diedit atau dianalisa oleh *ArcView*.

8. Shapefile

Shapefile adalah format data yang menyimpan lokasi geometrik dan informasi atribut dari suatu feature geografis. (Eko Budianto; 2009 : 1)

II.5. Website

Website adalah sebutan bagi sekelompok halam web (*web page*), dan umumnya merupakan bagian dari suatu nama domain (*domain name*), atau subdomain dalam *World Wide Web* (WWW) di internet. WWW terdiri dari seluruh situs web yang tersedia kepada publik. Jika Anda sering menggunakan fasilitas internet dan mengunjungi *Yahoo, Google, Friendster*, atau *Facebook*. Maka nama-nama itu menunjukkan suatu domain di internet (*www.detik.com*, *www.google.co.id*, *www.jawapos.com*, *www.yahoo.co.id*). (MADCOMS;2009;1)

Ada beberapa *software* yang dapat digunakan untuk merancang *interface* sebuah *website* pribadi. Di antaranya adalah *Adobe Photoshop* dan *Adobe Fireworks*. Rancangan yang dibuat dalam *Photoshop* dapat disimpan menjadi html, kemudian dapat diolah kembali menggunakan *Adobe Dreamweaver*. (MADCOMS;2009;1)

II.6. PHP

PHP adalah bahasa pemograman (*script*) yang digunakan untuk membuat halaman *web* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru atau *up to date*. Semua script

PHP dieksekusi pada server dimana script tersebut dijalankan. (Rulianto Kurniawan; 2010 : 3)

II.6.1. Kelebihan *PHP* dari Bahasa Pemrograman *Web*

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* hingga *xitami* dengan konfigurasi yang relatif mudah.
3. Dalam isi 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 referesi yang banyak.
5. PHP adalah bahasa bahasa *open source* yang dapat digunakan di berbagai mesih *(Linux, Unix Macintosh, Windows)* dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

II.6.2. Fungsi Dalam *PHP*

Fungsi merupakan hal yang paling penting dalam membuat aplikasi *web*. Dengan membagi kode-kode yang ada ke dalam fungsi-fungsi akan memudahkan kita apabila kita menggunakan kembali kode tersebut. Atau apabila kita ingin membuat *website* dengan fitur yang sama dengan *website* yang pernah kita buat maka kita cukup menggunakan fungsi-fungsi yang pernah kita buat.

Hal ini akan sangat menghemat waktu dan mempercepat proses pembuatan *website.* Fungsi pada *PHP sintax*-nya adalah *function namafungsi()* dimana *namafungsi* merupakan nama fungsi tersebut dan bisa kita ganti sesuai yang kita inginkan.

II.7. MySQL

MySQL adalah salah satu jenis database server yang sangat terkenal. *MySQL* termasuk jenis *RDBMS* (*Relational Database Management System*). *MySQL* ini mendukung bahasa pemograman *PHP*. *MySQL* juga mempunyai query atau bahasa *SQL* (*Structured Query Language*) yang simpel dan menggunakan *escape character* yang sama dengan *PHP*. (Rulianto Kurniawan; 2010 : 17)

MySQL merupakan database server yang bersifat multiuser dan multi-threaded. *SQL* adalah bahasa database standar yang memudahkan penyimpanan, pengubahan dan akses informasi. Pada *MySQL* dikenal istilah database dan tabel. Tabel asalah sebuah struktur data dua dimensi yang terdiri dari baris-baris *record* dan kolom. (Faisal S.Si; 2011 : 126)

II.8. Unified Modeling Language (*UML*)

UML (*Unified Modeling Language*) adalah sebuah ”bahasa” yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. *UML* menawarkan sebuah standar untuk merancang model sebuah sistem. Seperti bahasa-bahasa lainnya, *UML* mendefenisikan notasi dan *sintax/semantic*. Notasi *UML* merupakan sekumpulan

bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan *UML sintax* mendefinisikan bagaimana bentuk – bentuk tersebut dapat dikombinasikan.

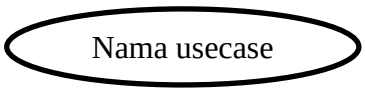
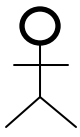
Karena tergolong bahasa visual, *UML* lebih mengedepankan penggunaan diagram untuk menggambarkan aspek dari sistem yang sedang dimodelkan. Bahasa visual lebih dekat ke mental model pikiran kita, sehingga pemodelan menggunakan bahasa visual bias lebih mudah dan lebih cepat dipahami dibandingkan sebuah bahasa pemrograman. *Unified Modeling Language* biasa digunakan untuk :

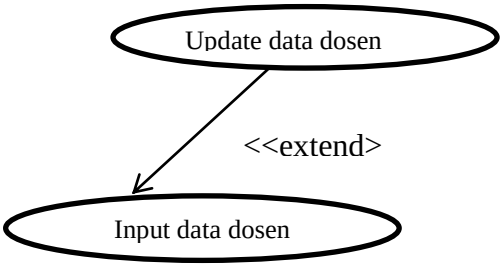
1. Menggambarkan batasan sistem dan fungsi – fungsi sistem secara umum, di buat dengan *use case* dan *actor*.
2. Menggambarkan kegiatan atau proses bisnis yang di laksanakan secara umum, di buat dengan *interaction diagrams*.
3. Menggambarkan representasi struktur *static* sebuah sistem dalam bentuk *class diagram*s.
4. Membuat model behavior “yang menggambarkan kebiasaan atau sifat sebuah sistem” dengan *state transition diagrams*.
5. Menyatakan arsitektur implementasi fisik menggunakan *component and development diagrams*.
6. Menyampaikan atau memperluas *fungsionality* dengan *stereotypes*. (Yuni Sugiarti; 2013 :36)

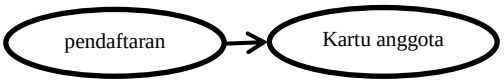
II.8.1. Use Case Diagram

Use case diagrams merupakan pemodelan untuk menggambarkan kelakuan sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi–fungsi tersebut. Terdapat beberapa simbol dalam menggambarkan diagram *use case*, yaitu *use case*, *actor* dan relasi. Berikut adalah sismbol – simbol yang ada pada diagram *use case*. (Yuni Sugiarti; 2013: 42).

Tabel II.1 : Simbol – simbol pada Use Case

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau *actor*; biasanya ditanyakan dengan menggunakan kata kerja di awal frase nama *use case*.
Aktor  nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari *acto*r adalah gambar orang, tapi *actor* belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal frase nama *actor*.
Asosiasi/ *association*	Komunikasi antara actor daan use case yang

———————	berpartisipasi pada use case atau use case memiliki interaksi dengan kator.
Extend <<extend>>	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, arah panah menunjuk pada use case yang dituju. Contoh : 

Include <<include>> ⟶	Relasi use case tambahan kesebuah use case dimana use case yang yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. Ada dua sudut pandang yang cukup besar mengenai include di use case, *include* berarti use case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan, contoh :  <<include>

(Sumber: Yuni Sugiarti; 2013)

Diagram *use case* adalah sebuah diagram yang menjelaskan apa yang harus dilakukan oleh sistem pada level konseptual sehingga kita akan memahami apakah keputusan yang diambil oleh sistem adalah benar atau tidak. (Yuni Sugiarti; 2013: 45)

II.8.2. *Class* Diagram

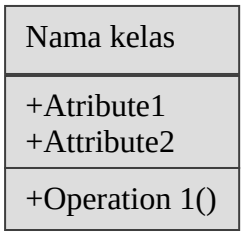
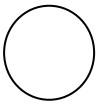
Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefenisian kelas – kelas yang akan di buat untuk membangun sistem. Kelas memiliki apa yang di sebut atribut dan metode atau operasi.

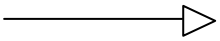
1. Atribut merupakan variabel- variabel yang di miliki oleh suatu kelas.

2. Atribut mendeskripsikan properti dengan sebaris teks di dalam kotak kelas tersebut.
3. Operasi atau metode adalah fungsi-fungsi yang di miliki oleh suatu kelas.

Diagram kelas mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. Diagram kelas juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. Diagram kelas menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *contaiment,* pewarisan, asosiasi, dan lain-lain. (Yuni Sugiarti; 2013: 57)

Tabel II.2 : Simbol – simbol Class Diagram

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / interface  **Interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi 1 1..*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*

Asosiasi berarah/directed asosiasi 	Relasi antar kelas dengan makna kelas yang satu di gunakan oleh kelas yang lain, asosiasi biasanya juga di sertai dengan *multiplicity*.
Generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum khusus).
Kebergantungan / defedency	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi 	Relasi antar kelas dengan makna semua bagian (*whole-part*)

(Sumber : Yuni Sugiarti ; 2013)

II.8.3. *Activity* Diagram

Activity diagram menggambarkan aktifitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga digunakan untuk mendefenisikan hal-hal berikut :

1. rancangan proses bisnis dimana setip ukuran aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujiannya.

Activity diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya

state sebelumnya. Karena itu *activity diagram* tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. (Yuni Sugiarti; 2013: 75)

II.8.4. *Sequence Diagram*

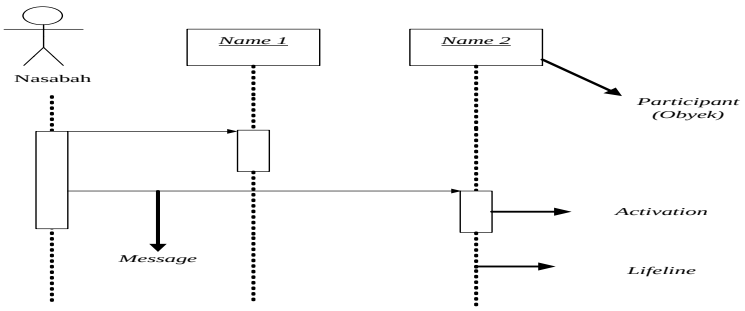
Diagram sekuence menggambarkan kelakuan / pelaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Karena itu untuk menggambarkan diagram sequence harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstasiasi menjadi objek itu. (Yuni Sugiarti; 2013: 69)

Diagram sekuence memiliki ciri yang berbeda dengan diagram interaksi pada diagram kolaborasi sebagai berikut :

1. Pada diagram sekuence terdapat garis hidup objek. Garis hidup objek adalah garis vertikal yang mencerminkan eksistensi sebuah objek sepanjang periode waktu. Sebagian besar objek-objek yang tercakup dalam diagram interaksi akan eksis sepanjang durasi tertentu dari interaksi, sehingga objek-objek itu diletakkan di bagian atas diagram dengan garis hidup tergambar dari atas hingga bagian bawah diagram. Suatu objek lain dapat saja diciptakan, dalam hal ini garis

hidup dimulai saat pesan *destroy*, jika kasus ini terjadi, maka garis hidupnya juga berakhir.

2. Terdapat fokus kendali, berupa empat persegi panjang ramping dan tinggi yang menampilkan aksi suatu objek secara langsung atau sepanjang sub ordinat. Puncak dari empat persegi panjang adalah permulaan aksi, bagian dasar adalah akhir dari suatu aksi. Pada diagram ini juga memperhatikan penyaringan (*nesting*) dan fokus kendali yang disebabkan oleh proses rekursif dengan menumpuk fokus kendali yang lain pada induknya. (Yuni Sugiarti; 2013: 70)



Gambar II.6 : Contoh Squence Diagram

(Sumber : Yuni Sugiarti ; 2013)

II.9. ERD *(Entity Relationship Diagram)*

Entity relationship diagram adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas – entitas dan menentukan hubungan antar entitas. Proses memungkinkan analis menghasilkan struktur basis data yang baik sehingga data dapat disimpan dan diambil secara efisien. Elemen-elemen diagram hubungan entitas yaitu :

1. Entitas (*Entity*)

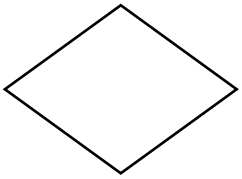
Entitas dalah sesuatu yang nyata atau abstrak di mana kita akan menyimpan data. Ada 4 kelas entitas, yaitu misalnya pegawai, pembayaran, kampus dan buku.

Gambar II.7 : Simbol Entitas

(Sumber : Janner Simarmata,dkk; 2013)

2. Relasi (*Relationship*)

Relasi adalah hubungan alamiah yang terjadi antara satu atau lebih entitas, misalny proses pembayaran pegawai. Kardinalitas menentukan kejadian suatu entitas untuk satu kejadian pada entitas yang berhubungan. Misalnya mahasiswa bisa mengambil banyak mata kuliah.



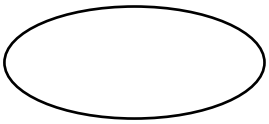
Gambar II.8 : Simbol Relasi

(Sumber : Janner Simarmata,dkk; 2013)

3. Atribut (*Attribute*)

Atribut adalah ciri umum semua semua atau sebagian besar instansi pada entitas tertentu. Sebutan lain atribut adalah *property*, elemen data, dan *field*. Misalnya, nama, alamat, nomor pegawai, dan gaji adalah atribut entitas pegawai. Sebuah atribut atau kombinasi atribut yang mengidentifikasikan

satu dan hanya satu instansi suatu entitas disebut kunci utama atau pengenal. Misalnya, nomor pegawai adalah kunci utama untuk pegawai.



Gambar II.9 : Simbol Atribut

(Sumber : Janner Simarmata,dkk; 2013)

II.10. Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basisdata relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilang data yang terduplikasi dari tabel relasional.

Pada waktu menormalisasi basisdata, ada empat tujuan yang harus dicapai, yaitu :

1. Mengatur data dalam kelompok-kelompok sehingga masing-masing kelompok hanya menangani bagian kecil sistem.
2. Meminimalkan jumlah data berulang dalma basisdata.
3. Membuat basisdata yang datanya diakses dan dimanipulasi secara cepat dan efisien tanpa melupakan integritas data.
4. Mengatur data sedemikian rupa sehingga ketika memodifikasi data, Anda hanya mengubah pada satu tempat. (Janner Simarmata & dkk; 2010 : 77)

II.11. Kamus Data

Kamus data (*data dictionary*) dipergunakan untuk memperjelas aliran data yang digambarkan pada DFD. Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukkan (*input)* dan keluaran (*output*) dapat dipahami secara umum (memiliki standar cara penulisan). Kamus data biasanya berisi:

1. Nama - nama dari data
2. Digunakan pada – merupakan proses-proses yang terkait data
3. Deskripsi – merupakan deskripsi data
4. Informasi tambahan – seperti tipe data, nilai data, batas nilai data, dan komponen yang membentuk data. (Rosa A.S & M Shalauddin; 2011 : 67)