BAB II

GAMBARAN UMUM PERUSAHAAN

BAB II

TINJAUAN PUSTAKA

II.1. Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) atau *Geographic Information System* (GIS) adalah sebuah system yang di desain untuk menangkap, menyimpan, memanipulasi, menganalisa, mengatur dan menampilkan seluruh jenis data geografis.

Akronim GIS *(Geographic Information System)* terkadang dipakai sebagai istilah untuk *geographical information science* atau *geospasial information studies* yang merupakan ilmu studi atau pekerjaan yang berhubungan dengan *Geographic Information System*. Dalam artian sederhana system informasi geografis dapat kita simpulkan sebagai gabungan kartografi, analisis dan teknologi system basis data (databases). (Edy Irwansyah ; 2013 : 1)

Pengertian system informasi geografis menurut beberapa ahli :

1. Burrough, 1986

   Kumpulan alat yang *powerful* untuk mengumpulakan, menyimpan, menampilkan dan mentransformasikan data spasial dari dunia nyata (*real world*)

2. Aronoff, 1989

   Segala jenis prosedur manual; maupun berbasis *computer* untuk menyimpan data dan memanipulasi data bereferensi geografis.

3. ESRI, 2004

Sebuah sistem untuk mengatur, menganalisa dan menapilkan informasi geografis. Sehingga dapat drangkum konsep sebuah sistem informasi geografis adalah sebagai berikut.

1. Informasi geografis adalah informasi mengenai tempat di pemukiman bumi
2. Teknologi informasi geografis meliputi *Global Positioning System* (GPS), *remote sensing* dan Sistem Informasi Geografis.
3. Sistem Informasi Geografis adalah system computer dan piranti lunak (*software*).
4. Sistem Informasi Geografis digunakan untuk berbagai macam aplikasi variasi aplikasi.
5. Sains Informasi Geografis merupakan ilmu sains yang melatarbelakangi teknologi Sistem informasi Geografis.

   SIG tidak lepas dari data spasial, yang merupakan sebuah data yang mengacu pada posisi,objek dan hubungan di antaranya dalam ruang bumi. Data spasial merupakan salah satu item dari informasi dimana di dalamnya terdapat informasi.

II.2. Intisari Jurnal Sistem Informasi Geografis

Menurut Edy Harseno, Vickey Igor R Tampubolon dalam Jurnal Aplikasi SistemInformasi Geografis Dalam Pemetaan batas Administrasi Tanah, Geologi, Penggunaan Lahan, Lereng , Daerah Istimewa Yogy7akarta dan Daerah Aliran

Sungai Di Jawa Tengah Menggunakan Software ArcView GIS, yang di maksud dengan Definisi Sistem Informasi Geografis (SIG) selalu berkembang, bertambah, dan bervariasi. SIG juga merupakan suatu bidang kajian ilmu dan teknologi yang relatif baru, digunakan oleh berbagai bidang disiplin ilmu, dan berkembang dengan cepat. SIG adalah sistem komputer yang digunakan untuk memasukkan (*capturing*), menyimpan, memeriksa, mengintegrasikan, memanipulasi, menganalisa, dan menampilkan datadata yang berhubungan dengan posisi-posisi di permukaan bumi. SIG dapat didefinisikan sebagai kombinasi perangkat keras dan perangkat lunak komputer yang memungkinkan untuk mengelola (*manage*), menganalisa, memetakan informasi spasial berikut data atributnya (data deskriptif) dengan akurasi kartografi (Basic, 2000 dalam Eddy Prahasta, 2002). Dari definisi ini dapat diuraikan menjadi beberapa subsistem yaitu data input, dasa otput, data manajemen, dan data manipulasi dan analisis.

Arsitektur GIS yaitu :

1. Data Raster

Dalam model data raster setiap lokasi direpresentasikan sebagai suatu posisi sel. Sel ini diorganisasikan dalam bentuk kolom dan baris sel-sel dan biasa disebut sebagai grid. Dengan kata lain, model data raster menampilkan, menempatkan, dan menyimpan data spasial dengan menggunakan struktur matriks atau piksel-piksel yang membentuk grid. Setiap piksel atau sel ini memiliki atribut tersendiri, termasuk koordinatnya yang unik. Setiap baris matrik berisikan sejumlah sel yang memiliki nilai tertentu yang merepresentasikan suatu fenomena

geografik. Nilai yang dikandung oleh suatu sel adalah angka yang menunjukan data nominal.

2. Atribut

Data atribut adalah data yang mendeskripsikan atau penjelasan dari suatu objek. data atribut dapat berupa informasi numerik, foto, narasi, dan lain sebagainya.

3.Peta

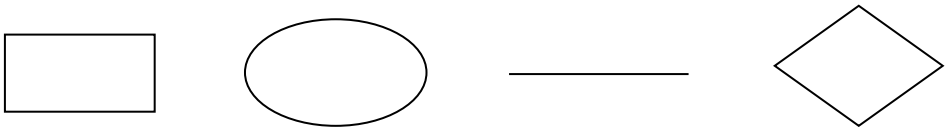
Peta dapat didefenisikan sebagai suatu alat penyajian secara grafis tentang penyebaran kenampakan-kenampakan geografis atau fenomena yang ada pada permukaan atau yang ada di dalam bumi.

Sumber : http://4archiculture.net

II.3. Basis Data *(Database)*

Secara sederhana database (basis data/ pangkalan data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan computer yang memungkinkan data dapat diakses dengan mudah dan cepat (kadir, 2004). Pengertian akses dapat mencakup pemerolehan data maupun pemanipulasian data

seperti menambah serta menghapus data. Dengan memanfaatkan computer, data dapat disimpan dalam media pengingat yang disebut *hard disk.* Dengan menggunakan media ini keperluan kertas untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih cepat untuk diakses terutama jika dikemas dalam bentuk database.



Entitas | Atribut/Field | Link (hubungan) | Himpunan Relasi/Interface

Gambar. II.2. Bentuk Simbol ERD
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

II.4. Sistem Basis Data

Sistem basis data merupakan perpaduan antara basis data dan sistem manajemen basis data (SMBD). Komponen –komponen sistem basis data meliputi:

a. Perangkat keras (Hardware)

   Perangkat keras computer adalah semua bagian fisik computer, contoh dari perangkat keras computer yaitu : Mouse, Keyboard, Monitor, CPU, Memori, dan lain-lain.

b. Sistem Operasi (Operating System)

   Sistem operasi merupakan suatu software sistem betugas untuk melakukan kontrol dan manajemen hardware serta operasi-operasi dasar sistem,

termasuk menjalankan software aplikasi seperti program program pengolah kata dan browser web. Secara umum, Sistem Operasi adalah sostware pada lapisan pertama yang ditaruh pada memori computer pada saat komputer dinyalakan.

II.5. Pengertian PHP

PHP adalah singkatan dari *Hypertext Prepprocessor* yaitu bahasa pemograman web server-side yang bersifat open source. PHP merupakan script atau kode yang digunakan untuk membuat halaman website yang dinamis. Dinamis berarti halaman yang ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru/up to date. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan. (Anhar st ; 2012 : 2).

PHP merupakan suatu bahasa pemrograman sisi server yang dapat anda gunakan untuk membuat halaman Web dinamis. Contoh bahasa yang lain adalah *Microsoft Active Server Page (*ASP) dan *Java Server Page*(JSP). Dalam suatu halaman HTML anda dapat menanamkan kode PHP yang akan dieksekusi setiap kali halaman tersebut dikunjungi. Karena kekayaannya akan fitur yang mempermudah perancangan dan pemrograman Web, PHP memiliki pepularitasyang tinggi. Anda dapat mengecek *survey*popularitas yang dilakukan *netcraft*di URL www.php.net/usage.php PHP adalah kependekan dari *HyperText Prepocessor* (suatu akronim rekursif) yang dibangun oleh Rasmus Lerdorf pada tahun 1994. Dahulu, pada awal pengembangnnya PHP disebut sebagai

kependekan dari *Personal Home Page.* PHP merupakan produk *OpenSource* sehingga anda dapat mengakses *sourcecode,* menggunakan dan mengubahnya tanpa harus membayar sepeser pun atau gratis..

II.6. Pengertian ArcView

ArcView adalah *software* yang dikeluarkan oleh ESRI (*Environmental System Research Institute*) Perangkat lunak ini meberikan fasilitas teknis yang berkaitan dengan penglolaan data spasial. Kemampuan grafis yang baik dan kemampuan teknis dalam pengolahan data spasial tersebut memberikan secara nyata pada *ArcView* untuk melakukan analisis spasial. Kekuatan analisis inilah yang pada akhirnya menjadikan *ArcView* banyak diterapkan dalam berbagai pekerjaan, seperti analisis pemasaran, perencanaan wilayah dan tata ruang, sistem informasi persil, pengendalian dampak lingkungan, bahkan keperluan militer. Sistem Informasi Kemampuan perangkat SIG *ArcView* secara umum dapat dijabarkan sebagai berikut (Eko Budianto ; 2010 : 177) .

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) Menjawab *query* spasial maupun atribut.
5) Melakukan fungsi-fungsi dasar SIG.
6) Membuat peta tematik.
7) Meng-*customize* aplikasi dengan menggunakan bahasa skrip.

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

II.7. Pengertian MySQL

Mysql pertama kali dirintis oleh seorang programmer *database* bernama Michael Widenius, yang dapat anda hubungi di emailnya monty@analytikerna.

Mysql database server adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. meskipun begitu, tidak menuntut *resource* yang besar. Mysql adalah *database* yang paling popular diantara *database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multiuser*. MySQL memiliki dua bentuk lisensi, yaitu *free software* dan *shareware*. penulis sendiri dalam menjelaskan buku ini menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*general public license*), yang dapat anda *download* pada alamat resminya http://www.mysql.com. MySQL sudah cukup lama dikembangkan, beberapa fase penting dalam pengembangan MySQL adalah sebagai berikut :

- MySQL dirilis pertama kali secara internal pada 23 Mei 1995
- Versi windows dirilis pada 8 Januari 1998 untuk windows 95 dan windows NT.
- Versi 3.23 : beta dari Juni 2000, dan dirilis pada January 2001.

– Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (unions) (Wahana Komputer ; 2010 : 5).

Gambar II.1. Tampilan Awal MySQL
(Sumber : Wahana Komputer ; 2010 : 1)

II.8. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD merupakan salah satu alat (tool) berbentuk grafis yang populer untuk *desain database.* Tool ini relatif lebih mudah dibandingkan dengan Normalisasi. Kebanyakan sistem analis memakai alat ini, tetapi yang jadi masalah, kalau kita cermati secara seksama, tool ini mencapai 2NF (Ir. Yuniar Supardi ; 2010 : 448).

II.9. Kamus Data

Karena DBMS menyimpan kumpulan beberapa item data yang terpisah yang dapat digunakan pemakai pada beberapa aplikasi secara bersama-sama, adalah penting bahwa beberapa mekanisme digunakan untuk menyediakan

informasi mengenai beberapaiitem data bersangkutan. Itu adalah fungsi dari kamus data.

Kamus data adalah suatu file yang terpisah yang menyimpan informasi seperti :

a. Nama setiap item/jenis/kolom data.
b. Struktur data untuk tiap item.
c. Program yang menggunakan tiap item
d. Tingkat keamanan untuk setiap item.

Pemakai yang perlu memperoleh informasi dari database dapat menuju ke kamus data untuk mendapatkan nama dari item data yang digunakan pada penelusuran (search). Dan yang mendesain aplikasi dapat menggunakan kamus untuk menentukan apakah item data sudah disimpan di komputer, dan kalau sudah dengan nama apa item data tersebut dapat dipanggil dan aplikasi apa yang digunakan.

Kamus data berguna khusus bagi perlindungan timbulnya kelebihan data. Tanpa kamus data, pemakai dari lain bagian mungkin menyimpan versi identik dari item data yang sama pada beberapa lokasi, dimana masing-masing item data mempunyai nama yang berbeda.

Operasional komputer dalam organisasi dewasa ini banyak yang sudah menggunakan model kerja jaringan (network). Dengan menggunakan data dasar yang sama untuk keperluan informasi masing-masing unit dan manajemen organisasi secara keseluruhan (Zulkifli Amsyah ; 2005 : 382).

II.10. Teknik Normalisasi

Salah satu topik yang cukup kompleks dalam dunia manajemen *database* adalah proses untuk menormalisasi tabel-tabel dalam *database relasional.* Dengan normalisasi kita ingin mendesain *database relasional* yang terdiri dari tabel-tabel berikut :

1. Berisi data yang diperlukan.
2. Memiliki sesedikit mungkin redundansi.
3. Mengakomodasi banyak nilai untuk tipe data yang diperlukan.
4. Mengefisienkan update.
5. Menghindari kemungkinan kehilangan data secara tidak disengaja/tidak diketahui.

Alasan utama dari normalisasi *database* minimal sampai dengan bentuk normal ketiga adalah menghilangkan kemungkinan adanya "*insert ionanomalies"*, "*deletion anomalies",* dan "*update anomalies".* Tipe-tipe kesalahan tersebut sangat mungkin terjadi pada *database* yang tidak normal.

II.10.1. Bentuk-bentuk Normalisasi

a. Bentuk tidak normal

Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja tidak lengkap dan terduplikasi. Data dikumpulkan apa adanya sesuai keadaanya.

b. Bentuk normal tahap pertama (1" Normal Form)

Definisi :

Sebuah table disebut 1NF jika :

- Tidak ada baris yang duplikat dalam tabel tersebut.
- Masing-masing cell bernilai tunggal

Catatan: Permintaan yang menyatakan tidak ada baris yang duplikat dalam sebuah tabel berarti tabel tersebut memiliki sebuah kunci, meskipun kunci tersebut dibuat dari kombinasi lebih dari satu kolom atau bahkan kunci tersebut merupakan kombinasi dari semua kolom.

c. Bentuk normal tahap kedua (2nd normal form)

Bentuk normal kedua (2NF) terpenuhi jika pada sebuah tabel semua atribut yang tidak termasuk dalam primarykey memiliki ketergantungan fungsional pada primarykey secara utuh.

d. Bentuk normal tahap ketiga (3rd normal form)

Sebuah tabel dikatakan memenuhi bentuk normal ketiga (3NF), jika untuk setiap ketergantungan fungsional dengan notasi X -> A, dimana A mewakili semua atribut tunggal di dalam tabel yang tidak ada di dalam X, maka :

- X haruslah superkey pada tabel tersebut.
- Atau A merupakan bagian dari primarykey pada tabel tersebut.

e. Bentuk Normal Tahap Keempat dan Kelima

Penerapan aturan normalisasi sampai bentuk normal ketiga sudah memadai untuk menghasilkan tabel berkualitas baik. Namun demikian,

terdapat pula bentuk normal keempat (4NF) dan kelima (5NF). Bentuk Normal keempat berkaitan dengan sifat ketergantungan banyak nilai (*multivalueddependency*) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional. Adapun bentuk normal tahap kelima merupakan nama lain dari *Project Join Normal Form*(PJNF).

f. BoyceCode Normal Form (BCNF)

- Memenuhi 1st NF
- Relasi harus bergantung fungsi pada atribut superkey (Kusrini ; 2007 : 39-43).

II.11. UML (*Unified Modeling Language*)

UML singkatan dari *Unified Modelling Langguage* yang berarti bahasa pemodelan standart. (Chonoles; 2003 : 6) mengatakan sebagai bahasa, berarti UML memiliki sintaks dan *semantic*. Ketika kita membuat model menggunakan konsep UML ada aturan–aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat harus berhubungan satu dengan lainnya harus mengikuti standart yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi error yang terjadi? Bagaimana keamanan terhadap sistem yang ada kita buat? Dan sebagainya dapat dijawab dengan UML.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

UML telah diaplikasikan dalam investasi perbankan,lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, sales, dan supplier.

Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorentasikan objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota team untuk bekerja sama dalam mengaplikasikan beragam sistem. Intinya, UML merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini. Sebagai perancang sistem mau tidak mau pasti menjumpai UML, baik kita sendiri yang membuat sekedar membaca diagram UML buatan orang lain (Prabowo Pudjo Widodo Herlawati ; 2011 ; 6).

II.11.1. *Diagram-Diagram* UML

Beberapa literatur menyebutkan bahwa UML menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa yang digabung,

misalnya diagram komunikasi, diagram urutan, dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

1. Diagram Kelas (*Class Diagram*). Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi, serta relasi-relasi diagram. Diagram ini umu dijumpai pada pemodelan sistem berorentasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas.
2. Diagram paket (*Package Diagram*) bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram *Use Case* bersifat statis. Diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.
4. Diagram interaksi dan *Sequence* (urutan). Bersifat dinamis. Diagram urutan adal;ah diagram interaksi yang menekankan pada pengiriman pesan dalam waktu tertentu.
5. Diagram komunikasi (*Communication Diagram*) bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML yang menekankan organisasi *structural* dari objek-objek yang menerima serta mengirim pesan.

6. Diagram *Statechart* (*Statechart Diagram)* bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*State)*, transisi kejadian serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi dan terutam penting pada pemodelan sistem-sistem yang reaktif.
7. Diagram aktivitas (*Activity Diagram*) bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan member tekanan pada aliran kendali antar objek.
8. Diagram komponen (*Component Diagram*) bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan diagram kelas dimana komponen dipetakan kedalam satu atau lebih kelas-kelas. Antarmuka-antarmuka serta kolaborasi-kolaborasi.
9. Diagram *Deployment* (*Deployment Diagram)* bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*runtime)*. Memuat simpul-simpul berserta komponen-komponen yang ada di dalamnya. Diagram *Deployment*berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing).*

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan.

1. *Diagram UseCase (usecase diagram)*

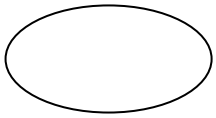
Use Case menggambarkan *externalview* dari sistem yang akan kita buat modelnya. Menurut Pooley (2005:15) mengatakan bahwa model *usecase*dapat dijabarkan dalam diagram, tetapi yang perlu diingat, diagram tidak identik dengan model karena model lebih luas dari diagram.

komponen pembentuk diagram *use case* adalah :

a. Aktor (*actor),* menggambarkan pihak-pihak yang berperan dalam sistem.
b. *Use Case,* aktivitas/ sarana yang disiapkan oleh bisnis/sistem.
c. Hubungan (*Link),* aktor mana saja yang terlibat dalam *usecase*ini.

1. *Use Case*

Menurut Pilone (2005 : 21) *use case* menggambarkan fungsi tertentu dalam suatu sistem berupa komponen kejadian atau kelas. Sedangkan menurut Whitten (2004 : 258) mengartikan *usecase* sebagai urutan langkah-langkah yang secara tindakan saling terkait (skenario) baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Usecase* digambarkan dalam bentuk *ellips*/oval



Gambar II.3. Simbol *UseCase*
Sumber : Probowo Pudjo Widodo (2011:22)

Use case sangat menentukan karakteristik sistem yang kita buat, oleh karena itu Chonoles (2003:22-23) menawarkan cara untuk menghasilkan *use case* yang baik yakni :

a. Pilihlah nama yang baik

Use case adalah sebuah *behaviour* (prilaku), jadi seharusnya dalam frase kata kerja. Untuk membuat namanya lebih detil tambahkan kata benda mengindikasikan dampak aksinya terhadap suatu kelas objek. Oleh karena itu diagram *use case* seharusnya berhubungan dengan diagram kelas.

b. Ilustrasikan perilaku dengan lengkap.

Use case dimulai dari inisiasi oleh aktor primer dan berakhir pada aktor dan menghasilkan tujuan. Jangan membuat *usecase* kecuali anda mengetahui tujuannya. Sebagai contoh memilih tempat tidur (*King Size*, *Queen Size*, atau dobel) saat tamu memesan tidak dapat dijadikan *usecase* karena merupakan bagian dari *use case* pemesanan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).

c. Identifikasi perilaku dengan lengkap.

Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, *usecase* harus lengkap. Ketika memberi nama pada *use case*, pilihlah frasa kata kerja yang implikasinya hingga selesai. Misalnya gunakan frasa *reserve a room*(pemesanan kamar) dan jangan *reserving a room*(memesan kamar) karena memesan menggambarkan perilaku yang belum selesai.

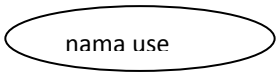
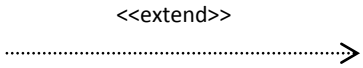
d. Menyediakan use case lawan (*inverse)*

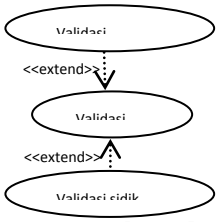
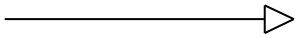
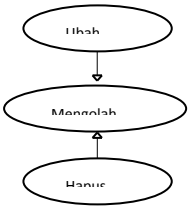
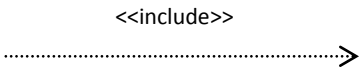
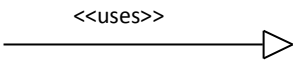
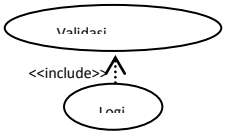
Kita biasanya membutukan *use case* yang membatalkan tujuan, misalnya pada use *case* pemesanan kamar, dibutuhkan pula *use case* pembatalan pesanan kamar.

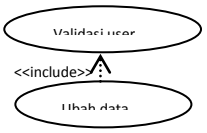
e. Batasi use case hingga satu perilaku saja.

Kadang kita cenderung membuat *use case* yang lebih dari satu tujuan aktivitas. Guna menghindari kerancuan, jagalah use case kita hanya fokus pada satu hal. Misalnya, penggunaan *use case checkin* dan *checkout* dalam satu *use case* menghasilkan ketidak fokusan, karena memiliki dua perilaku yang berbeda.

Tabel II.1. Simbol Dari Use Case Diagram

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor / *actor*  nama aktor	Orang, proses, atau sistem yang lain berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan di buat itu sendiri
Asosiasi / *association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case*, atau *usecase* memiliki interasi dengan aktor
Ekstensi / *extend* 	Relasi usecase tambahan ke subuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanppa *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 misal

	 arah panah mengarah pada *use case* yang ditambahkan
Generalisasi / *generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainya misalnya :  Arah panah mengarah pada use case yang menjadi generalisasinya (umum)
Menggunakan / *include* / *uses*  	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini Ada 2 sudut pandang yang cukup besar mengenai include di usecase 1. include berarti use case yang ditambahkan akan selalu dipanggil saat use case dijalankan misal pada kasus berikut :  2. include berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang di tambahkan telah di jalankan sebelum use case tambahan di jalankan, misal pada kasus berikut :

	 Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

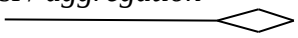
Sumber : Probowo Pudjo Widodo (2011:175)

2. Diagram Kelas (*Class Diagram*)

Diagram kelas adalah inti dari proses pemodelan objek. Baik *forward engineering* maupun *reverse engineering* memanfaatkan diagram ini *forward engineering* adalah proses perubahan model menjadi kode program sedangkan *reverse engineering* sebaliknya merubah kode program menjadi model (Probowo Pudji Widodo; 2011 : 37)

Tabel II.2. Simbol Diagram Kelas

Simbol	Deskripsi
Kelas Nama_kelas +atribut +operasi()	Kelas pada struktur sistem
Antarmuka / *interface* Nama_*interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* Asosiasi berarah/ *directed association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity* Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
Kebergantungan	Relasi antar kelas dengan makna kebergantungan antar kelas

Agregasi / *aggregation* 	Semua bagian (*whole part*)

(Sumber : Probowo Pudji Widodo; 2011 : 37)

3. Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan bisnis juga. Diagram aktivitas menunjukan aktivitas sistem dalam kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software*, diagram aktivitas merepresentasikan pemanggilan suatu fungsi tertentu misalnya *call.* Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian diluar seperti pemesanan atau kejadian-kejadian internal misalnya penggajian tiap jumat sore (ProbowoPudji Widodo ;2011 : 143-145).

Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi melakukan langka sekali saja tidak boleh dipecah menjadi beberapa langkah-langkah lagi. Contoh aksinya yaitu :

a. Fungsi Matematika
b. Pemanggilan Perilaku
c. Pemrosesan Data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classfier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classfier*, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan dengan model

proses bisnis, informasi itu biasanya disebut *process-relevant data*. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya *specific* dan digunakan hanya untuk aktivitas tertentu.

Aktivitas digambarkan dengan persegi panjang tumpul. Namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya.

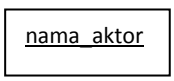
Detail aktivitas dapat dimasukan di dalam kotak. Aksi diperlihatkan dengan symbol yang sama dengan aktivitas dan namanya diletakkan didalam persegi panjang.

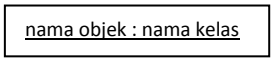
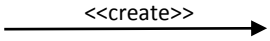
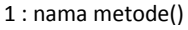
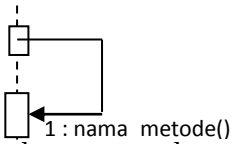
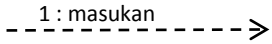
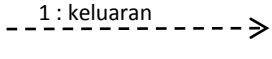
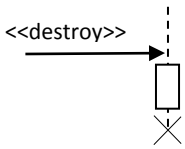
4. *Squence Diagram*

Menurut Douglas (2004 : 174) menyebutkan ada tiga diagram primer UML dalam memodelkan scenario interaksi, yaitu diagram urutan (*sequence diagram)*, diagram waktu (*timing diagram)* dan diagram komunikasi (*communication diagram).*

Menurut Pilone (2005 : 174) menyatakan bahwa diagram yang paling banyak dipakai adalah diagram urutan. Tabel II.3. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nantinya.

Tabel II.3. Simbol Dari Sequence Diagram

Simbol	Deskripsi
Aktor  nama aktor atau  tampa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya di nyatakan menggunakan kata benda di awali frase nama aktor

Garis hidup / lifeline	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berintaraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create 	Objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tope call 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri  Arah panah mengarah pada objek yang memiliki operasi / metode, karena ini memanggil operasi / metode maka operasi / metode yang di panggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainya, arah panah mengarah pada objek yang dikirimi
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasiatau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Sumber : Probowo Pudjo Widodo (2011:175)