# BAB II

# TINJAUAN PUSTAKA

## II.1. Pengertian Sistem

Untuk mengawali pembahasan tentang analisis dan perancangan sistem informasi, pemahaman akan sistem terlebih dahulu harus ditekankan. Definisi sistem berkembang sesuai dengan konteks dimana pengertian sistem itu digunakan. Berikut akan diberikan beberapa definisi sistem secara umum :

Kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama. Contoh :

1. Sistem Tatasurya
2. Sistem Pencernaan
3. Sistem Transportasi Umum
4. Sistem Otomatif
5. Sistem Komputer
6. Sistem Informasi

Sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan.

Dengan demikian, secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variable-variabel yang saling terogranisasi, saling berinteraksi, dan saling bergantung sama lain. *Murdick* dan *Ross* mendefinisikan sistem sebagai seperangkat elemen yang digabungkan satu

dengan lainnya untuk satu tujuan bersama. Sementara definisi sistem dalam kamus *Webster's Unbriged* adalah elemen-elemen yang saling berhubungan dan membentuk satu kesatuan atau organisasi (Hanif Al Fatta ; 2007 : 3).

## II.2. Pengertian Informasi

Informasi adalah data yang berguna yang telah diolah sehingga dapat dijadikan dasar untuk mengambil keputusan yang tepat. Informasi sangat penting bagi organisasi. Pada dasarnya informasi adalah penting seperti sumber daya yang lain, misalnya peralatan, bahan, tenaga, dsb.

Informasi yang berkualitas dapat mendukung keunggulan kompetitif suatu organisasi. Dalam sistem informasi, kualitas dari informasi yang disediakan merupakan hal penting dalam kesuksesan sistem. Secara konseptual seluruh sistem organisasional mencapai tujuannya melalui proses alokasi sumber daya, yang diwujudkan melalui proses pengambilan keputusan manajerial. Informasi memiliki nilai ekonomik pada saat ia mendukung keputusan alokasi sumber daya, sehingga dengan demikian mendukung sistem untuk mencapai tujuan (Agustinus ; 2012 : 1).

## II.3. Pengertian Sistem Informasi

Sistem informasi bukan merupakan hal yang baru, yang baru adalah komputerisasinya. Sebelum ada komputer, teknik penyaluran informasi yang memungkinkan manajer merencanakan serta mengendalikan operasi telah ada. Komputer menambahkan satu atau dua dimensi, seperti kecepatan, ketelitian dan

penyediaan data dengan volume yang lebih besar yang memberikan bahan pertimbangan yang lebih banyak untuk mengambil keputusan.

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi untuk dapat menyediakan laporan-laporan yang diperlukan oleh pihak luar tertentu (Tata Sutabri ; 2012 : 38).

## II.4. Sistem Informasi Geografis

Sistem informasi geografis merupakan bagian dari geografi teknik (*technical geography*) berbasis komputer yang digunakan untuk menyimpan dan memanipulasi data-data keruangan (spasial) untuk kebutuhan atau kepentingan tertentu. Seiring dengan kemajuan dan perkembangan komputer, SIG dewasa ini telah mengalami kemajuan dan perkembangan yang sangat pesat sehingga merupakan suatu keharusan dalam perencanaan, analisis, dan pengambilan keputusan atau kebijakan. Kemajuan dan perkembangan SIG ini didorong oleh kemajuan dan perkembangan komputer, serta teknologi penginderaan jauh melalui pesawat udara dan satelit yang telah dimiliki oleh hampir sebagian besar negara maju di dunia. Kesimpulan yang didapat dari beberapa pengertian bahwa *Geography Information System* (GIS) atau Sistem Informasi Geografis (SIG) adalah suatu sistem informasi yang digunakan untuk memasukkan, menyimpan, memanggil kembali, mengolah, menganilisis, dan menghasilkan data bereferensi geografis atau data geospasial (Hartono ; 40 : 2007).

### II.4.1. Konsep Dasar Sistem Informasi Geografis

Sistem informasi geografis (SIG) pertama pada tahun 1960 yang bertujuan untuk menyelesaikan permasalahan geografis. 40 tahun kemudian perkembangan GIS berkembang tidak hanya bertujuan untuk menyelesaikan permasalahan geografi saja tetapi sudah merambah ke berbagai bidang seperti:

1. Analisis penyakit epidemik (demam berdarah)
2. Analisis kejahatan (kerusuhan)
3. Navigasi dan *vehicle routing* (lintasan terpendek)
4. Analisis bisnis (sistem stock dan distribusi)
5. *Urban* (tata kota) dan regional *planning* (tata ruang wilayah)
6. Peneliti: spatial data *Exploration*
7. *Utility* (listrik, PAM, telpon) *inventory and Management*
8. Pertahanan (military simulation), dll (Rahmad Husein ; 2006 : 1).

### II.4.2. karakteristik Sistem Informasi Geografis

Terdapat beberapa karakteristik dalam pembangunanan sistem informasi geografis seperti berikut :

1. Merupakan suatu sistem hasil pengembangan perangkat keras dan perangkat lunak untuk tujuan pemetaan, sehingga fakta wilayah dapat disajikan dalam satu sistem berbasis komputer.
2. Melibatkan ahli geografi, informatika dan komputer, serta aplikasi terkait.
3. Masalah dalam pengembangan meliputi: cakupan, kualitas dan standar data, struktur, model dan visualisasi data, koordinasi kelembagaan dan etika, pendidikan, *expert system* dan *decision support system* serta penerapannya.

4. Perbedaannya dengan Sistem Informasi lainnya: data dikaitkan dengan letak geografis, dan terdiri dari data tekstual maupun grafik.
5. Bukan hanya sekedar merupakan pengubahan peta konvensional (tradisional) ke bentuk peta digital untuk kemudian disajikan (dicetak / diperbanyak) kembali.
6. Mampu mengumpulkan, menyimpan, mentransformasikan, menampilkan, memanipulasi, memadukan dan menganalisis data spasial dari fenomena geografis suatu wilayah.
7. Mampu menyimpan data dasar yang dibutuhkan untuk penyelesaian suatu masalah. Contoh : penyelesaian masalah perubahan iklim memerlukan informasi dasar seperti curah hujan, suhu, angin, kondisi awan. Data dasar biasanya dikumpulkan secara berkala dalam jangka yang cukup panjang (Rahmad Husein ; 2006 : 4).

## II.5. Pengertian Quantum GIS

Quantum gis adalah gratis *dekstop* aplikasi GUS yang menyediakan data kepemirsaan, *editing* dan kemampuan analisis. QGIS berjalan pada *Linux*, Mac OSX, dan *Windows*. Quantum GIS ditulis dalam C + +, dan GUI yang menggunakan *librabry* Qt. Quantum GIS memungkinkan integrasi plugs-in yang dikembangkan baik menggunakan C + + atau *Python*. Pustaka Qt menyediakan *cross-platform* kerangka pengembangan aplikasi. didukung oleh perangkat lunak lain, QGIS menyediakan integrasi dengan paket GIS *open source* lainnya, termasuk PostGIS, GRASS dan *MapServer* untuk memberikan pengguna fungsi luas. QGIS terus dipelihara oleh kelompok aktif pengembang relawan yang secara

teratur merilis *update* dan perbaikan *bug*. Sebuah perangkat lunak GIS-komponen adalah sebuah blok bangunan yang, ketika ditambahkan ke perangkat lunak GIS, membentuk ditingkatkan, lingkungan pribadi bagi pengguna. komponen fungsi spesifik melakukan tugas khusus yang menambah alat GIS-lingkungan. Komponen tersebut termasuk *konverter* format data, analisis aliran data, dan perangkat lunak pengolah gambar. pengembangan perangkat lunak pengguna, di sisi lain, adalah pengembangan *toolkit* yang memungkinkan pengguna untuk komponen program untuk melakukan fungsi tertentu. Misalnya, seseorang mungkin perlu untuk menanamkan peta ke dalam program *non*-GIS, dan tidak ada komponen pra-dikembangkan untuk raster-satunya GIS. pengembangan perangkat lunak pengguna adalah satu-satunya jawaban dalam kasus ini (Yupo Chan ; 2011 : 432).

## II.6. Sekilah Mengenai PHP

PHP merupakan bahasa skrip yang digunakan untuk membuat halaman web yang dinamis. PHP bersifat *open product*. Pengguna dapat mengubah *source code* dan mendistribusikannya secara bebas serta diedarkan secara gratis. PHP bersifat *server scripting* yang dapat ditambahkan kedalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip PHP akan dilakukan di sebuah *web server*, kemudian hasilnya akan dikirimkan ke *browser*. Salah satu *web server* yang paling umum digunakan untuk PHP adalah Apache. PHP dapat dijalankan pada sistem operasi *Unix*, *Windows*, dan Mac OS X (Arief Ramadhan ; 2005 : 1).

## 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 populer di antara *database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multi user*. 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 Januari 2001.
- Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (*unions*) (Wahana ; 2010 : 5).

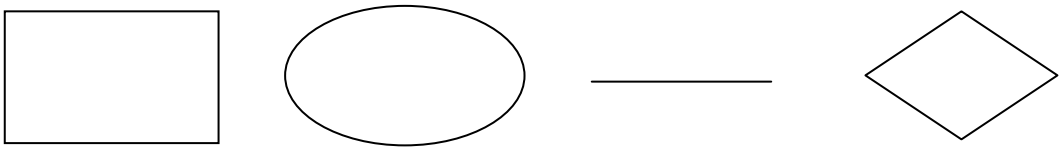
## II.8. Pengertian Database

*Database* adalah sekumpulan *file* data yang saling berhubungan dan diorganisasi sedemikian rupa sehingga memudahkan untuk mendapat dan memproses data. Lingkungan sistem *database* menekankan data yang tidak tergantung (*idenpendent data*) pada aplikasi yang akan menggunakan data. Data adalah kumpulan fakta dasar (mentah) yang terpisah.

Sebuah *database* harus dibuat dengan rapi agar data yang dimasukkan sesuai dengan tempatnya. Sebagai contoh, di sebuah perpustakaan, penyimpanan buku dikelompokkan berdasar jenis atau kategori-kategori tertentu, misalnya kategori buku komputer, buku pertanian, dan lain-lain. Kemudian dikelompokkan lagi berdasarkan abjad judul buku, ini dilakukan agar setiap pengunjung dapat dengan mudah mencari dan mendapatkan buku yang dimaksud (Wahana Komputer ; 2006 ; 1).

## II.9. 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 di cermati secara seksama, tool ini mencapai 2NF (Ir. Yuniar Supardi ; 2010 : 448).



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

**Gambar II.1. Bentuk Simbol ERD**
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

## II.10. Kamus Data

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

## II.11. 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 "*insertion anomalies"*, "*deletion anomalies",* dan "*update anomalies".* Tipe-tipe kesalahan tersebut sangat mungkin terjadi pada *database* yang tidak normal.

### II.11.1.Tahap-tahap Normalisasi

**a. Tahap 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. Tahap 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. Tahap normal tahap kedua (2nd normal form)**

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

**d. Tahap normal tahap ketiga (3rd normal form)**

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

- X haruslah *superkey* pada tabel tersebut.
- Atau Y merupakan bagian dari *primary key* pada tabel tersebut.

**e. Boyce Code Normal Form (BCNF)**

- Memenuhi 1st NF
- Relasi harus bergantung fungsi pada atribut *superkey* .

**f. Tahap 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 (*multivalued dependency*) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional. Adapun bentuk normal tahap kelima merupakan nama lain dari *Project Join Normal Form* (PJNF) (Kusrini ; 2007 : 39-43).

## II.12. *Unified Modeling Language* (UML)

UML singkatan dari *Unified Modelling Langguage* yang berarti bahasa pemodelan standart. 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.12.1. *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. 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 adalah 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 ketergantungan 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 (*run time)*. 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.

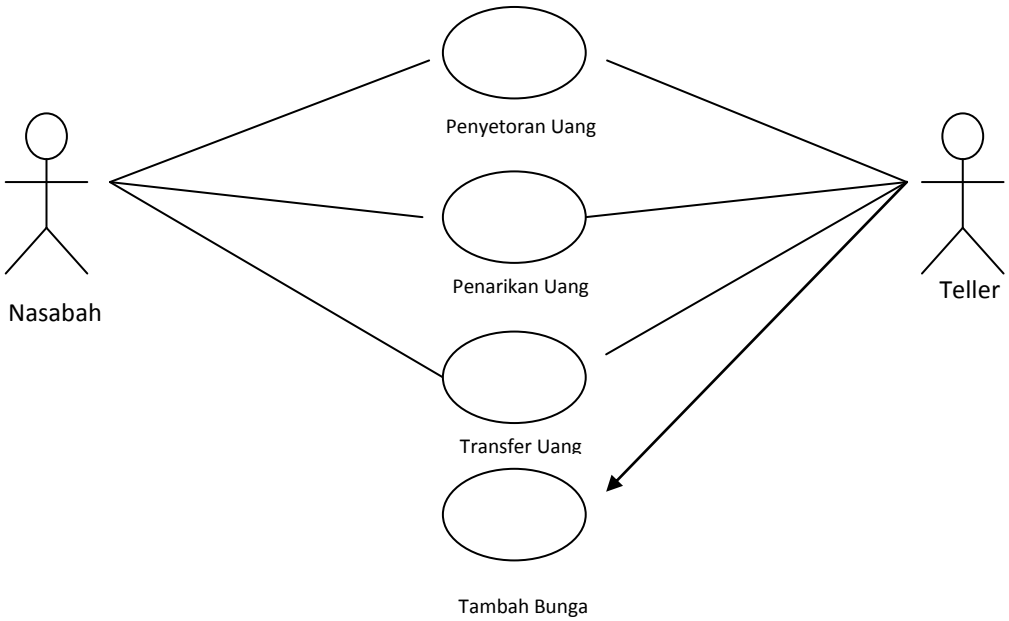
***Diagram Use Case (use case diagram)***

*Use Case* menggambarkan *external view* dari sistem yang akan kita buat modelnya. Menurut Pooley (2005:15) mengatakan bahwa model *use case* dapat dijabarkan dalam diagram, tetapi yang perlu diingat.

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 *use case* ini.

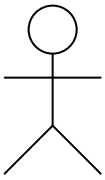
Gambar di bawah ini merupakan salah satu contoh bentuk diagram *use case.*




**Gambar II.2. Diagram *Use Case***
Sumber : Probowo Pudjo Widodo (2011:17)

**1. Aktor**

Menurut Chonoles (2003 :17) menyarankan sebelum mebuat use case dan menentukan aktornya, agar mengidentifikasi siapa saja pihak yang terlibat dalam sistem kita. Pihak yang terlibat biasanya dinamakan *stakeholder*.

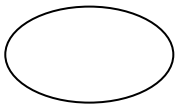


**Gambar II.3. Aktor**
Sumber : Probowo Pudjo Widodo (2011:17)

**2. *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 *use case* sebagai urutan langkah-langkah

yang secara tindakan saling terkait (skenario) baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk *ellips*/oval



**Gambar II.4. Simbol *Use Case***
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 *use case* kecuali anda mengetahui tujuannya. Sebagai contoh memilih tempat tidur (*King Size, Queen Size,* atau dobel) saat tamu memesan tidak dapat dijadikan *use case* 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, *use case* 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)***

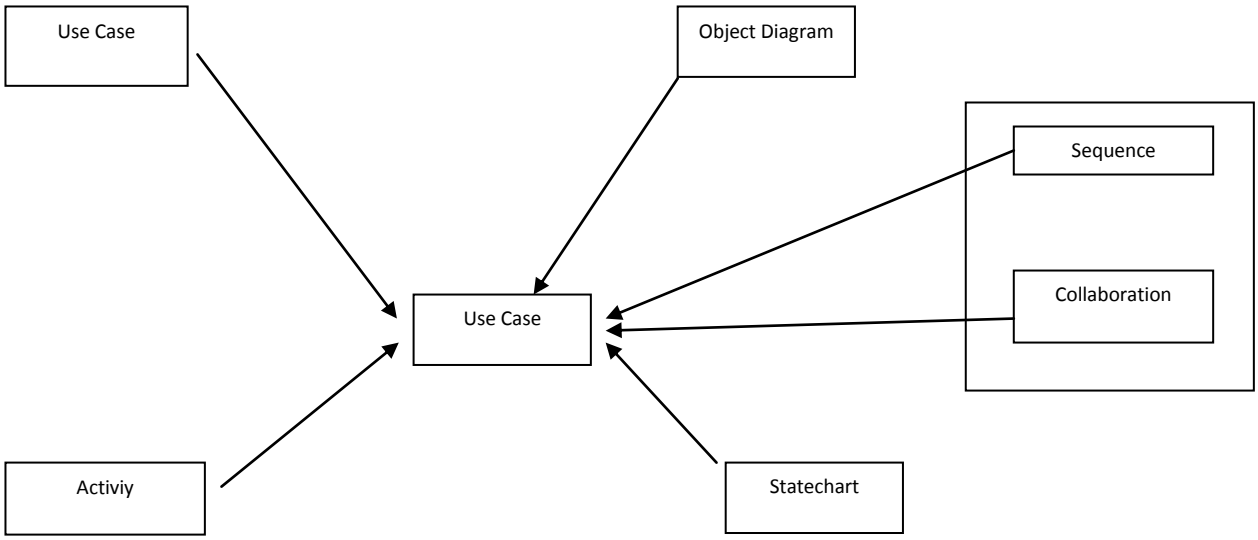
Kita biasanya membutukan *use case* yang membatalkan tujuan, misalnya pada u*se case* pemesanan kamar, dibutuhkan pula *use case* pembatalan pesanan kamar.

**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 check in* dan *check out* dalam satu *use case* menghasilkan ketidakfokusan, karena memiliki dua perilaku yang berbeda.

**3. 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).




**Gambar II.5. Hubungan Diagram Kelas Dengan Diagram *UML* lainya**
Sumber : Probowo Pudjo Widodo (2011 : 38)

**4. 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 (Probowo Pudji Widodo ; 2011 : 143-145).

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

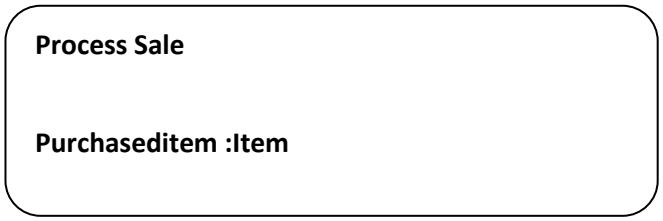
a. Fungsi Matematika

b. Pemanggilan Perilaku

c. Pemrosesan Data

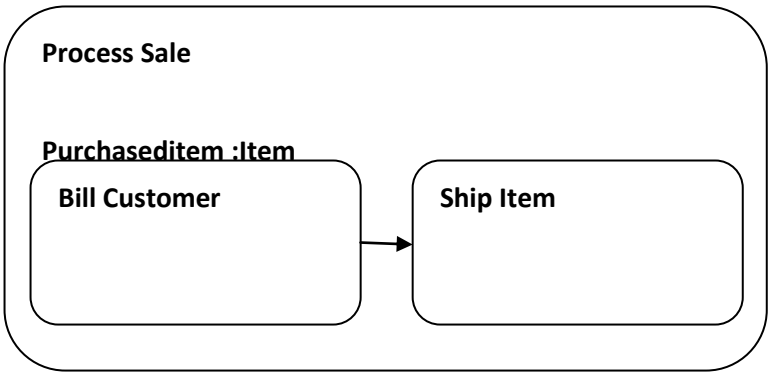
Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classfier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classfier*, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan dengan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




**Gambar II.6. Aktivitas serderhana tanpa rincian**
Sumber : Probowo Pudjo Widodo (2011:145)

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

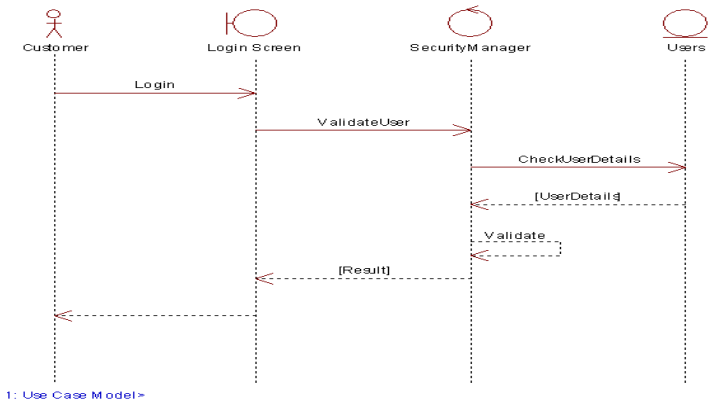



**Gambar II.7. Aktivitas dengan detail rincian**
Sumber : Probowo Pudjo Widodo (2011:145)

**5. *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. Gambar II.14. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nantinya.




**Gambar II.8. Diagram Urutan**
Sumber : Probowo Pudjo Widodo (2011:175)

## II.13. Daftar Pustaka.


Chan, Yupo, 2011. ***Location Theory and Decision Analysis: Analytics of Spatial InformationTechnology***. Springer. USA.

Fatta, Hanif Al, 2007. ***Analisis & Perancangan Sistem Informasi Untuk Keunggulan Bersaing Perusahaan & Organisasi Modern***. Andi, Yogyakarta.

Hartono, 2007. ***Geografi : Jelajah Bumi dan Alam Semesta***. Citra Praya, Bandung.

Husein, Rahmad, 2006. ***Jurnal : Konsep Dasar Sistem Informasi Geografis***. Ilmu Komputer.

Komputer, Wahana, 2010. ***Panduan Belajar MySQL Database Server***. TransMedia, Jakarta.

Komputer, Wahana, 2006. ***Seri Panduang Aplikatif Membuat Aplikasi Database Dengan Java 2***. Andi, Yogyakarta.

Kusrini, 2007. ***Strategi Perancangan dan Pengolaan Basis Data***. Andi, Yogyakarta.

McLeod, Raymond, 2008. ***Sistem Informasi Manajemen***. Salemba Empat, Jakarta.

Mujilan, Agustinus, 2012. ***Sistem Informasi Akuntansi Teori dan Wawasan di Dunia Elektronis***. WIMA Pers, Madiun.

Prabowo, 2011. ***Menggunakan UML***. Informatika, Bandung.

Ramadhan, Arief, 2005. ***Buku Latihan PHP 5 dan MySQL***. Elex Media, Jakarta.

Supardi, Yuniar, 2010. ***Semua Bisa Menjadi Programmer Java Basic Programming***. Elex Media, Jakarta.

Sutabri, Tata, 2012. ***Analisis Sistem Informasi***. Andi, Yogyakarta.