BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem

Sistem adalah sebuah tatanan yang terdiri atas sejumlah komponen fungsional (dengan tugas/fungsi khusus) yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses/pekerjaan tertentu. Sebagai contoh, sistem kendaraan terdiri dari komponen stater, komponen pengapian, komponen penggerak, komponen pengerem, komponen kelistrikan – *speedometer*, lampu dan lain-lain. Komponen-komponen tersebut diatas memiliki tujuan yang sama, yaitu untuk membuat kendaraan tersebut bias dikendarai dengan nyaman dan aman (Kusrini ; 2007 : 11).

II.2. Pengertian Sistem Informasi

Sistem Informasi dapat didefinisikan sebagai kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan serta mendistribusikan informasi. Dengan kata lain, SI merupakan kesatuan elemen-elemen yang saling berinteraksi secara sistematis dan teratur untuk menciptakan dan membentuk aliran informasi yang akan mendukung pembuatan keputusan dan melakukan kontrol terhadap jalannya perusahaan.

Sistem Informasi juga mampu mendukung para pengelola dan staf perusahaan untuk menganalisa permasalahan, memvisualisasikan ikhtisar analisa

melalui grafik-grafik dan tabel-tabel, serta memungkinkan terciptanya produk serta layanan yang baru. Sistem Informasi yang baik tentu memiliki sistematika yang jelas, ringkas, dan sederhana. Mulai dari tahap pemasukan data, pengolahan dengan prosedur yang ditentukan, penyajian informasi yang akurat, interprestasi yang tepat dan distribusinya (Budi Sutedjo Dharma Oetomo ; 2006 : 11).

II.3. Sistem Informasi Geografis

Sistem Informasi Geografis (*geographicinformationsystem-GIS)* adalah kategori khusus dari DSS yang menggunakan teknologi visualisasi data untuk menganalisis dan menampilkan data untuk perencanaan dan pengambilan keputusan dalam bentuk peta digital. Peranti lunak tersebut merakit, menyimpan, memanipulasi, dan menampilkan secara geografis informasi referensi, menghubungkan data dengan titik, garis, dan bidang pada sebuah peta. GIS mempunyai kemampuan membuat model, memungkinkan manajer untuk mengubah data dan secara otomatis memperbarui scenario bisnis untuk mencari solusi yang lebih baik.

GIS membantu pengambilan keputusan yang membutuhkan pengetahuan tenatang distribusi penduduk atau sumber daya lain secara geografis. Sebagai contoh, GIS mungkin digunakan untuuk membantu pemerintah Negara dan pemerintah lokal menghitung waktu respon bahaya untuk bencana alam, untuk membantu perusahaan eceran mengidentifikasi lokasi pertokoan baru, atau membantu bank mengidentifikasi tempat terbaik untuk membangun cabang atau memasang terminal ATM baru.

Sesi interaktif manajemen menjelaskan aplikasi GIS untuk mengendalikan tindak kejahatan. CompStat diciptakan oleh Departemen Kepolisian *NewYork* untuk mengambil data tentang insiden tindak kejahatan dan aktivitas penegakan hukum di setiap sudut kota. CompStat menggunakan peranti lunak GIS untuk menampilkan data mengenai di mana kejahatan berlangsung dan diklaim berhasil mengurangi jumlah rata-rata tindak kejahatan di *New York* dan kota-kota lain (Kenneth C. Laudan ; 2008 : 167).

II.4. Intisari Jurnal Sistem Informasi Geografis

Menurut Lusi Melian dalam Jurnal Perancangan Dan Pembangunan Sistem Informasi Geografis Pariwisata Di Kota Bandung Berbasis Website, yang dimaksud dengan Sistem Informasi Georafis (SIG) atau *GeoraphicInformation Sistem* (*GIS*) merupakan suatu sistem informasi yang berbasis komputer, dirancang untuk bekerja dengan menggunakan data yang memiliki informasi spasial (bereferensi keruangan). Sistem ini menangkap, mengecek, mengintegrasikan, memanipulasi, menganalisa, dan menampilkan data yang secara spasial mereferensikan kepada kondisi bumi. Teknologi SIG mengintegrasikan operasi-operasi umum *database*, seperti *query* dan analisa statistik, dengan kemampuan visualisasi dan analisa yang unik yang dimiliki oleh pemetaan. Kemampuan inilah yang membedakan SIG dengan Sistem Informasi lainya yang membuatnya menjadi berguna berbagai kalangan untuk menjelaskan kejadian, merencanakan strategi, dan memprediksi apa yang terjadi.

Analisa kebutuhan sistem dalam merancang sistem informasi geografis menurut jurnal Hamidi yang berjudul Aplikasi Sistem Informasi Geografis Berbasis Web Penyebaran Dana BantuanOperasional Sekolah, menentukan bagaimana orang, data, proses dan teknologi informasi dapat saling terhubung. Dengan analisis, suatu sistem diharapkan dapat diuraikan secara utuh menjadi komponen-komponen dasar dengan tujuan mengidentifikasi serta mengevaluasi permasalahan dan kebutuhan yang diharapkan.

Mapserver merupakan salah satu *software*web GIS *opensoure*. Konsep kerja map sever yaitu output program dari ArcView berupa *file*. shp, shx dan dbf dideklarasikan dalam suatu *file* map sebagai layanan konfigurasi dasar dalam *file* ArcView semua atribut peta berupa garis, titik dan area. Untuk menggunakan Map Server diperlukan dukungan keberadaan beberapa perangkat lunak seperti: Sistem Operasi komputer, *WebServer*, program aplikasi CGI MapServer itu sendiri, *texteditor* dan program aplikasi *browser internet.*

II.5. Pengertian Quantum GIS

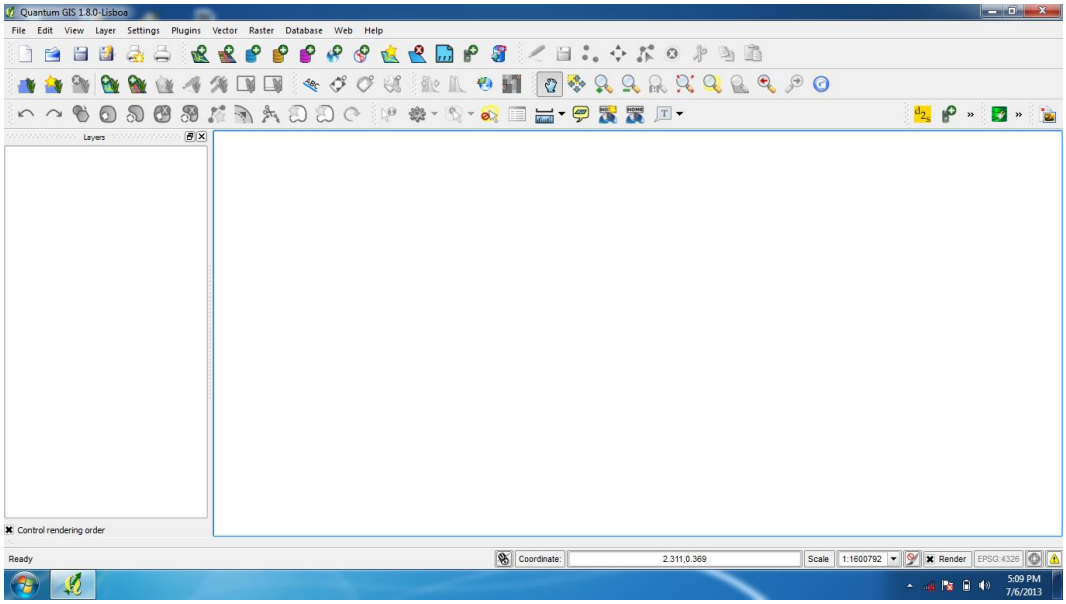
Quantum GIS atau yang sering disingkat menjadi QGIS adalah sebuah apliasi sistem informasi geografis berbasis *desktop* yang menyediakan fitur untuk menampilkan data, pengubahan datam dan kemampuan dalam menganalisis data spasial. QGIS dapat berjalan pada sistem operasi Linux, UNix, Mac OS, dan Windows.

Quantum GIS dapat dibuat dengan bahasa pemrograman C++ dan untuk tampilan grafisnya menggunakan pustaka kode QT-Library. Quantum GIS

memungkinkan untuk membentuk integrasi pada Plug-In yang dikembangkan dengan C++ maupun Python.

QT-Library menyediakan tampilan grafis yang dapat berjalan secara *Cross-Platform* dalam *Framework* pengembangan aplikasi yang didukung oleh perangkat lunak lainnya. Quantum GIS memungkinkan untuk dihubungkan atau di integrasikan dengan berbagai paket perangkat lunak GIS yang bersifat Open-Source lainnya, seperti Post GIS, GRASS, dan MapServer untuk memberikan fugsionalitas yang ekstensif kepada penggunanya. Quantum GIS secara berkesinambungan terus diperbaiki dan dikembangkan oleh grup pengembang yang aktif dan pengembang sukarela yang secara teratur merilis pembaharuan dan perbaikan pada beberapa kesalahan sistem.

Komponen perangkat lunak GIS dibangun berdasarkan blok-blok sehingga dapat ditambahkan perangkat lunak GIS dan dibentuk dengan baik serta lingkungan pengembangan yang dapat disesuaikan untuk pengguna. Fungsi komponen yang spesifik memberikan dedikasi tugas yang ditambahkan pada lingkungan alat pengembangan GIS, seperti komponen yang memungkinkan untuk memasukkan format data tertentu agar dapat dikonversi, penganalisis data teratur, dan perangkat pemrosesan citra, perangkat pengembangan pengguna di sisi lainnya sebagai fungsi yang spesifik(Yupo Chan ; 2011 : 432).



Gambar II.1. Tampilan Quantum GIS
(Sumber :Yupo Chan ; 2011 : 432)

II.6. Pengertian PHP

PHP adalah kode atau skrip yang akan dieksekusi pada *serverside*. Skrip PHP akan membuat suatu aplikasi daat diintegrasikan ke dalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip dilakukan di *server* baru kemudian hasilnya dikirimkan ke *browser* (Deni Sutaji; 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 popularitas

yang tinggi. Anda dapat mengecek *survey* popularitas yang dilakukan *netcraft*di URL www.php.net/usage.php.

PHP adalah kependekan dari *HyperTextPrepocessor* (suatu akronim rekursif) yang dibangun oleh RasmusLerdorf 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. Gratis (Antonius Nugraha Widhi Pratama ; 2010 : 9).

II.7. Pengertian Database

Database merupakan komponen terpenting dalam pembangunan SI, karena menjadi tempat untuk menampung dan mengorganisasikan seluruh data yang ada dalam sistem, sehingga dapat dieksplorasi untuk menyusun-menyusun informasi-informasi dalam berbagai bentuk. *Database* merupakan himpunan kelompok data yang saling berkaitan. Data tersebut diorganisasikan sedemikian rupa agar tidak terjadi duplikasi yang tidak perlu, sehingga dapat diolah atau di eksplorasi secara cepat dan mudah untuk menghasilkan informasi (Budi Sutedjo Dharma Oetomo ; 2006 : 99).

D*atabase* 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.8. Pengertian MySQL

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

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

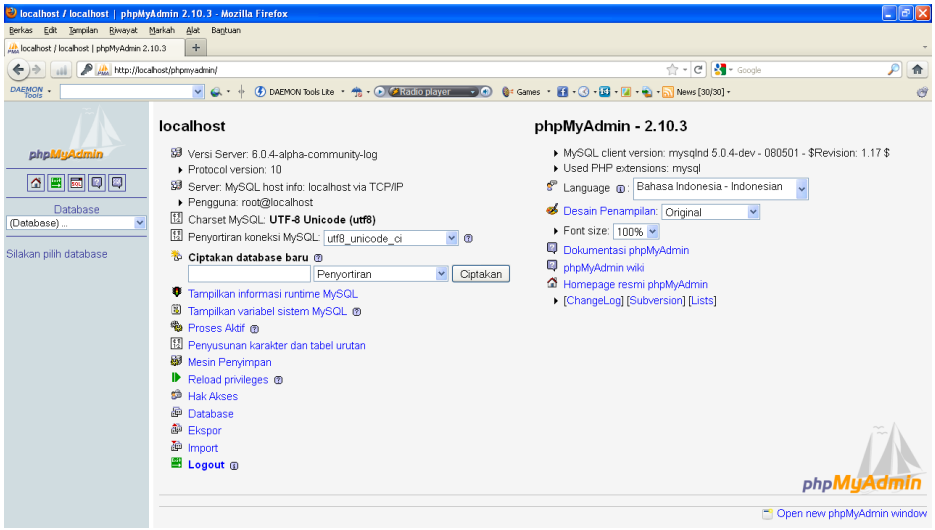
MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multiuser*. MySQL memiliki dua bentuk lisensi, yaitu *freesoftware* dan *shareware*. penulis sendiri dalam menjelaskan buku ini menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*generalpubliclicense*), 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.

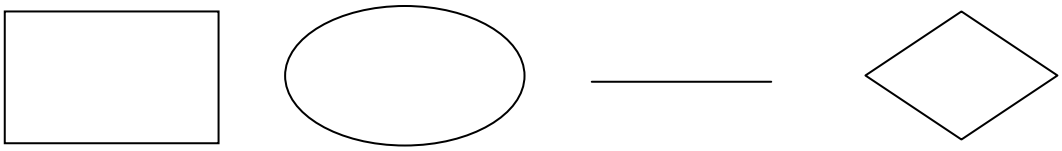
– Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (unions) (Wahana Komputer ; 2010 : 5).



Gambar II.2. Tampilan Awal MySQL
(Sumber : Wahana Komputer ; 2010 : 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 kita cermati secara seksama, tool ini mencapai 2NF (Ir. Yuniar Supardi ; 2010 : 448).



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

Gambar. II.3 Bentuk Simbol ERD
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

II.10. 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 beberapa item data bersangkutan. Itu adalah fungsi dari kamus data.

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

1. Nama setiap item/jenis/kolom data.
2. Struktur data untuk tiap item.
3. Program yang menggunakan tiap item
4. 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.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 "*insertionanomalies"*,

"*deletionanomalies",* dan "*updateanomalies".* Tipe-tipe kesalahan tersebut sangat mungkin terjadi pada *database* yang tidak normal.

Bentuk-bentuk Normalisasi antara lain :

1. 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.

2. 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.

3. Bentuk normal tahap kedua (2^{nd} 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.

4. 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.

5. 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).

6. BoyceCode Normal Form (BCNF)

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

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

UML singkatan dari *UnifiedModellingLangguage* 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 (PrabowoPudjo Widodo Herlawati ; 2011 ; 6).

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 (*PackageDiagram*) bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram *UseCase* 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 *Statechar t*(*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 (*distributedcomputing).*

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

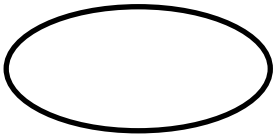
II.12.1. *Diagram UseCase (usecase diagram)*

UseCase 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 *usecase* adalah :

1. Aktor (*actor),* menggambarkan pihak-pihak yang berperan dalam sistem.
2. *UseCase,* aktivitas/ sarana yang disiapkan oleh bisnis/sistem.
3. Hubungan (*Link),* aktor mana saja yang terlibat dalam *usecase* ini.

Menurut Pilone (2005 : 21) *usecase* 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.4. Simbol *UseCase*
(Sumber : Probowo Pudjo Widodo ; 2011:22)

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

1. Pilihlah nama yang baik

Usecase 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 *usecase* seharusnya berhubungan dengan diagram kelas.

2. Ilustrasikan perilaku dengan lengkap.

Usecase 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 *usecase* pemesanan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).

3. Identifikasi perilaku dengan lengkap.

Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, *usecase* harus lengkap. Ketika memberi nama pada *usecase*, 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.

4. Menyediakan usecase lawan (*inverse)*

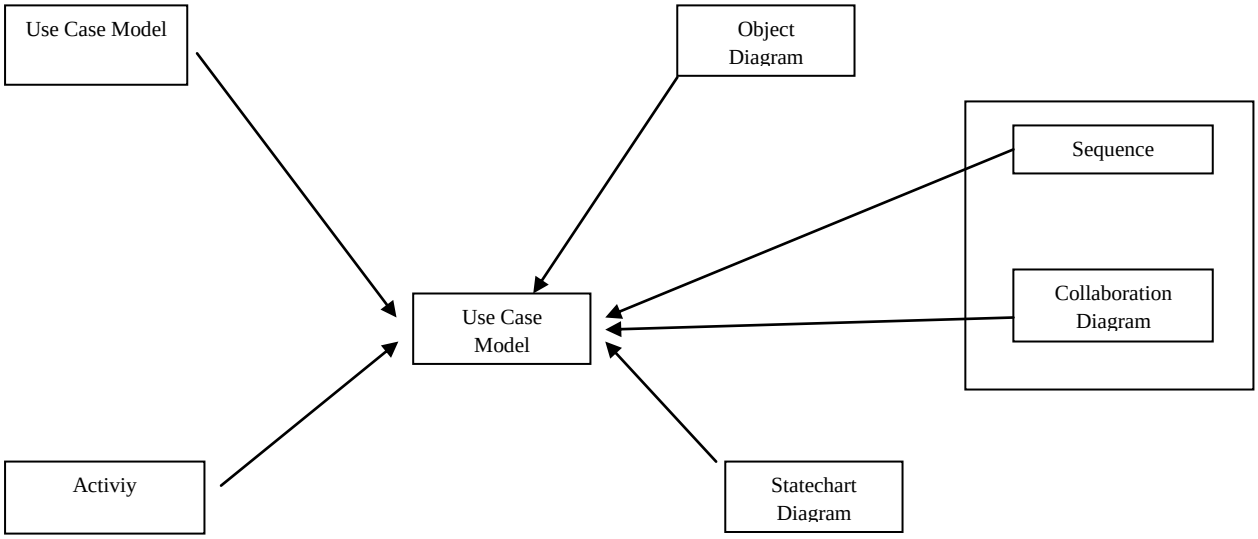
Kita biasanya membutukan *usecase* yang membatalkan tujuan, misalnya pada use*case* pemesanan kamar, dibutuhkan pula *usecase* pembatalan pesanan kamar.

5. Batasi usecase hingga satu perilaku saja.

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

II.12.2. Diagram Kelas (*Class Diagram*)

Diagram kelas adalah inti dari proses pemodelan objek. Baik *forwardengineering* maupun *reverseengineering* memanfaatkan diagram ini *forwardengineering* adalah proses perubahan model menjadi kode program sedangkan *reverseengineering* sebaliknya merubah kode program menjadi model (ProbowoPudji Widodo; 2011 : 37).



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

II.12.3. 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 (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 :

1. Fungsi Matematika
2. Pemanggilan Perilaku
3. 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.

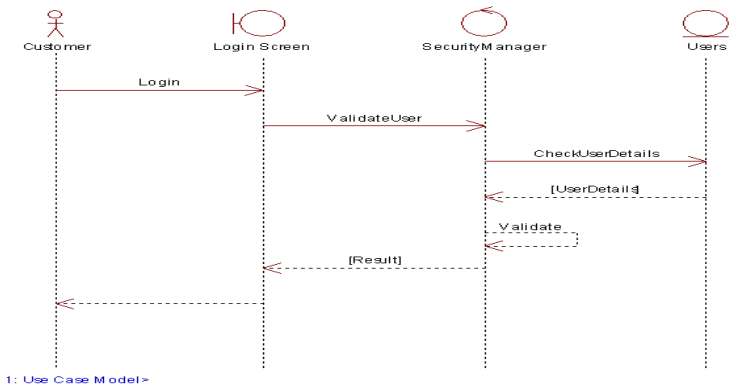
Aktivitas digambarkan dengan persegi panjang tumpul. Namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya.

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

II.12.4. *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. Gambar II.7. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nantinya.



Gambar II.6. Diagram Urutan
(Sumber : Probowo Pudjo Widodo ; 2011:175)