

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **II.1. Sistem**

Secara sederhana suatu sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari suatu unsur, komponen, atau variabel yang terorganisir, saling tergantung satu sama lain dan terpadu. Teori sistem secara umum yang pertama kali diuraikan oleh Kennet Boulding, terutama menekankan pentingnya perhatian terhadap setiap bagian yang membentuk sebuah sistem. Kecenderungan manusia yang mendapat tugas memimpin suatu organisasi adalah terlalu memusatkan perhatian pada salah satu komponen saja dari sistem organisasi.

Teori sistem melahirkan konsep-konsep futuristik, antara lain yang terkenal adalah konsep sibernetika (*cybernetics*). Konsep atau dibidang kajian ilmiah ini berkaitan dengan upaya menerapkan berbagai ilmu yaitu ilmu perilaku, fisika, biologi, dan teknik. Oleh karena itu sibernetika biasanya berkaitan dengan usaha-usaha otomasi tugas-tugas yang dilakukan manusia, sehingga melahirkan studi-studi tentang robotika, kecerdasan buatan (*artificial intelegence*). Unsur-unsur yang mewakili suatu sistem secara umum adalah masukan (*input*), pengolahan (*processing*), dan keluaran (*output*).

selain itu, suatu sistem tidak bisa lepas dari lingkungan maka umpan balik (*feed back*) dapat berasal dari lingkungan sistem yang dimaksud. Organisasi dipandang sebagai suatu sistem yang tentunya akan memiliki semua unsur ini (Tata Sutabri; 2012 : 10

### II.1.1. Karakteristik Sistem

Model umum sebuah sistem adalah input, proses, dan output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana sebab sebuah sistem dapat mempunyai beberapa masukan dan keluaran. Selain itu sebuah sistem memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut:

#### 1. Komponen Sistem (*Component*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling berkerja sama membentuk satu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat dari sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem yang lebih besar, yang disebut “supra sistem.

#### 2. Batas Sistem (*Boundary*).

Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lain atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisah-pisahkan.

#### 3. Lingkungan Luar Sistem (*Environment*).

Bentuk apapun yang ada di luar lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut operasi lingkungan luar sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan dan dapat juga

bersifat merugikan sistem tersebut. Lingkungan yang menguntungkan merupakan bagi sistem tersebut. Dengan demikian, lingkungan luar tersebut harus dijaga dan dipelihara. Lingkungan luar yang merugikan harus dikendalikan. Kalau tidak maka akan mengganggu kelangsungan hidup sistem tersebut.

#### 4. Penghubung Sistem (*Interface*)

Media yang menghubungkan sistem dengan subsistem lainnya disebut penghubung sistem atau *interface*. Penghubung ini memungkinkan sumber-sumber daya mengalir dari subsistem lain. Bentuk keluaran dari satu subsistem akan menjadi masukan untuk subsistem lain melalui penghubung tersebut. Dengan demikian dapat terjadi suatu integrasi sistem untuk membentuk satu kesatuan.

#### 5. Masukan Sistem (*Input*)

Energi yang dimasukkan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan (*maintenance input*) dan sinyal (*signal input*). Contoh di dalam suatu sistem unit komputer. “program” adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan “data” adalah *signal input* untuk diolah menjadi informasi.

#### 6. Keluaran Sistem (*Output*)

yaitu hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain. Contoh, sistem informasi. Keluaran yang dihasilkan adalah informasi. Informasi ini

dapat digunakan sebagai masukan untuk pengambil keputusan atau hal-hal lain yang menjadi *input* bagi subsistem lain

#### 7. Pengolah Sistem (*Proses*).

Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran. Contoh, sistem akuntansi. Sistem ini akan mengolah data transaksi menjadi laporan-laporan yang dibutuhkan oleh pihak manajemen.

#### 8. Sasaran Sistem (*Objective*)

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat *deterministik*. Kalau suatu sistem tidak memiliki sasaran, maka operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan (Tata Sutabri; 2012 :20-21).

### II.1.2. Klasifikasi Sistem

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lain karena sistem memiliki sasaran yang berbeda setiap kasus yang terjadi yang ada di dalam sistem tersebut. Oleh karena itu sistem dapat diklasifikasikan dari beberapa sudut pandangnya antara lain.

#### 1. Sistem Abstrak Dan Sistem Fisik

Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik, misalnya sistem teologia, yaitu sistem yang berupa pemikiran hubungan antara manusia dengan tuhan, sedangkan sistem fisik merupakan sistem yang ada secara fisik, misalnya sistem komputer, sistem

produksi, sistem penjualan, sistem administrasi personalia, dan lain sebagainya.

2. Sistem Alamiah Dan Sistem Buatan Manusia.

Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia, misalnya sistem perputaran bumi, terjadinya siang malam, pergantian musim. Sedangkan sistem buatan manusia dengan mesin, merupakan melibatkan interaksi manusia dengan mesin, yang disebut "*human machine system*". Sistem informasi berbasis komputer merupakan contoh *human machine system* karena menyangkut penggunaan komputer yang berinteraksi dengan manusia.

3. Sistem Deterministik Dan Sistem Probabilistik.

Sistem yang beroperasi dengan tingkah laku yang dapat diprediksi disebut sistem deterministic. Sistem komputer adalah contoh dari sistem yang tingkah lakunya dapat dipastikan berdasarkan program-program komputer yang dijalankan. Sedangkan sistem bersifat probabilistik adalah sistem yang mana kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilistic.

4. Sistem Terbuka Dan Sistem Tertutup.

Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh oleh lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa campur tangan pihak luar. Sedangkan sistem terbuka adalah sistem yang berhubungan dan dipengaruhi oleh lingkungan luarnya. Sistem ini

menerima masukan dan menghasilkan keluaran untuk subsistem lainnya (Tata Sutabri; 2012 : 22-26).

### **II.1.3. Daur Hidup Sistem**

Siklus hidup sistem (*system life cycle*) adalah merupakan proses evolusioner yang diikuti dalam menerapkan sistem atau subsistem informasi komputer. Siklus hidup sistem terdiri dari serangkaian tugas yang erat mengikuti langkah-langkah pendekatan sistem sistem karena tugas-tugas tersebut mengikuti pola yang teratur dan dilakukan secara *top down*. Siklus hidup sistem sering disebut sebagai pendekatan air terjun (*waterfall approach*) bagi pembangunan dan pengembangan sistem.

Pembangunan sistem hanyalah salah satu dari rangkaian daur hidup suatu sistem. Meskipun demikian, proses ini merupakan aspek yang sangat penting. beberapa fase/tahapan dari daur hidup suatu sistem. (Tata Sutabri; 2012 :27-29

#### **1. Mengenali Adanya Kebutuhan.**

Sebelum segala sesuatunya terjadi, timbul suatu kebutuhan atau problema yang harus dapat dikenali sebagaimana adanya. Kebutuhan dapat terjadi sebagai hasil perkembangan dari organisasi dan volume yang meningkat melebihi kapasitas dari sistem yang ada. Semua kebutuhan ini harus dapat didefinisikan dengan jelas. Tanpa adanya kejelasan dari kebutuhan yang ada, pembangunan sistem akan kehilangan arah dan efektifitasnya.

## 2. Pembangunan Sistem

Suatu proses atau seperangkat prosedur yang harus diikuti untuk menganalisis kebutuhan yang timbul dan membangun suatu sistem untuk dapat memenuhi kebutuhan tersebut.

## 3. Pemasangan Sistem

Setelah tahap pembangunan sistem selesai. Sistem kemudian akan dioperasikan. Pemasangan sistem merupakan tahap yang penting pula dalam daur hidup sistem. Peralihan dari tahap pembangunan menuju tahap operasional terjadi pemasangan sistem yang sebenarnya, yang akan merupakan langkah akhir dari suatu pembangunan.

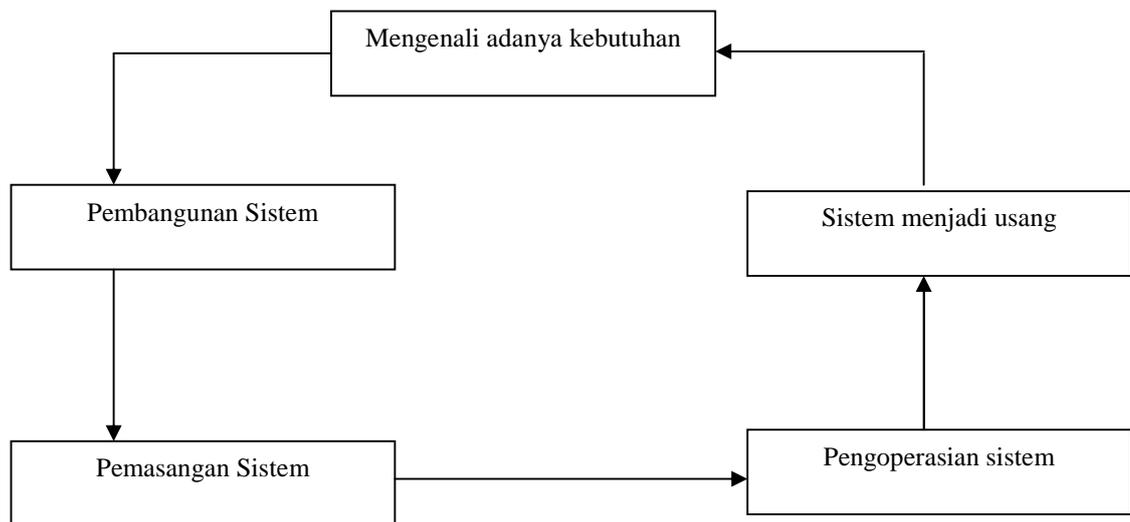
## 4. Pengoperasian Sistem

Program-program komputer dan prosedur-prosedur pengoperasian yang membentuk suatu sistem informasi semuanya bersifat statis. Sedangkan organisasi ditunjang oleh sistem informasi tadi. Ia selalu mengalami perubahan-perubahan itu karena pertumbuhan kegiatan bisnis, perubahan pengaturan, dan kebijaksanaan ataupun kemajuan teknologi. Untuk mengatasi perubahan-perubahan tersebut, sistem harus diperbaiki atau diperbaharui.

## 5. Sistem Menjadi Usang

Kadang perubahan yang terjadi begitu drastis, sehingga tidak dapat diatasi hanya dengan melakukan perbaikan-perbaikan pada sistem yang berjalan. Tibalah saatnya secara ekonomis dan teknis sistem yang ada sudah tidak layak lagi untuk dioperasikan dan sistem yang baru perlu dibangun untuk menggantikannya.

Sistem informasi kemudian akan melanjutkan daur hidupnya. Sistem dibangun untuk memenuhi kebutuhan yang muncul. Sistem beradaptasi terhadap perubahan-perubahan lingkungannya dinamis. Sampailah pada kondisi dimana sistem tersebut tidak dapat lagi beradaptasi dengan perubahan-perubahan yang ada atau secara ekonomis tidak layak lagi untuk dioperasikan. Sistem yang baru kemudian dibangun untuk menggantikannya. Untuk dapat menggambarkan daur hidup sistem dapat dilihat pada gambar II.1. sebagai berikut.



**Gambar II.1. Daur Hidup Sistem**

*Sumber : Tata Sutabri (2012 : 29)*

## **II.2. Informasi**

Informasi adalah data yang telah diklasifikasikan diolah atau diinterpretasi untuk digunakan dalam proses pengambil keputusan. Sistem pengolahan

informasi mengolah data menjadi informasi atau tepatnya mengolah dari bentuk tak berguna menjadi berguna bagi penerimanya.

Informasi adalah data yang telah diklasifikasikan atau diinterpretasi untuk digunakan dalam pengambil keputusan (Tata Sutabri; 2012 : 29).

### **II.2.1. Sistem Informasi**

Sistem informasi adalah berupa suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan data transaksi harian yang mendukung operasi yang bersifat manajerial dengan kegiatan strategi suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang diperlukan (Tata Sutabri; 2012 :46)

### **II.2.2. Komponen Dan Jenis Sistem Informasi.**

Sistem informasi terdiri dari komponen-komponen yang disebut blok bangunan (*building block*), yang terdiri dari blok masukan, blok model, blok keluaran, blok teknologi, blok basis data dan blok kendali. Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lain membentuk satu kesatuan untuk mencapai sasaran.

#### **1. Blok Masukan (*Input Block*).**

Input mewakili data yang masuk ke dalam sistem informasi. *Input* di sini termasuk metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

2. Blok Model (*Model Block*)

Blok ini terdiri dari kombinasi prosedur, logika, dan model matematik yang akan memanipulasi data *input* dan data yang tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

3. Blok Keluaran (*Output Block*)

Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta pemakai sistem.

4. Blok Teknologi (*Technology Block*)

Teknologi merupakan “*tool box*” dalam sistem informasi. Teknologi digunakan untuk menerima *input*, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem keseluruhan. Teknologi sistem terdiri dari 3 (tiga) bagian yaitu teknisi teknologi (*brainware*), perangkat lunak (*software*), dan perangkat keras (*hardware*).

5. Blok Basis Data (*Database Block*)

Basis data (*database*) merupakan kumpulan data yang saling berkaitan dan berhubungan satu dengan lainnya, tersimpan di perangkat keras komputer dan menggunakan perangkat lunak untuk memanipulasinya. Data perlu disimpan di basis data untuk keperluan penyediaan informasi lebih lanjut. Data di dalam basis data perlu diorganisasikan sedemikian rupa supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpanannya. Basis data diakses atau

dimanipulasi menggunakan perangkat lunak paket yang disebut DBMS (*database management system*).

#### 6. Blok Kendali (*Control Block*)

Banyak hal yang dapat merusak sistem informasi, seperti bencana alam, api, temperature, air, debu, kecurangan-kecurangan, kegagalan-kegagalan sistem itu sendiri, ketidak efisienan, sabotase, dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah atau bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi (Tata Sutabri; 2012 :46-47).

### **II.3. Pelaporan SPT**

Menurut Mardiasmo (2011 : 29) adalah : “Surat yang oleh Wajib Pajak digunakan untuk melaporkan penghitungan dan atau pembayaran pajak, objek pajak dan atau bukan objek pajak dan atau harta dan kewajiban yang terhutang menurut ketentuan peraturan perundang-undangan perpajakan.” Dalam Surat Edaran Direktur Jenderal Pajak Nomor SE- 103/PJ/2011 tentang petunjuk teknis tata cara penerimaan dan pengolahan Surat Pemberitahuan Tahunan yang selanjutnya disebut dengan SPT Tahunan adalah : “Surat Pemberitahuan untuk suatu tahun pajak atau bagian tahun pajak yang meliputi SPT Tahunan Pajak Penghasilan Wajib Pajak Orang Pribadi (SPT 1770, SPT 1770 S, SPT 1770 SS), SPT Tahunan Pajak Penghasilan Wajib Pajak Badan (SPT 1771 dan SPT 1771/S) termasuk SPT Tahunan Pembetulan.” Dari kedua pengertian di atas dapat ditarik kesimpulan bahwa SPT Tahunan adalah surat yang oleh Wajib Pajak digunakan

untuk melaporkan perhitungan dan atau pembayaran pajak yang terutang, objek pajak dan atau bukan objek pajak dan atau harta dan kewajiban menurut peraturan perundang-undangan. (Soemitro dalam Mardiasmo;20011:1)

Link <http://eprints.uny.ac.id/7889/3/BAB%202-09409134015.pdf>

#### **II.4. SPPT ( Surat Pemberitahuan Pajak Tahunan )**

SPT Tahunan PPh adalah formulir yang diisi Wajib Pajak untuk melaporkan identitas diri, harta, kewajiban/utang, penghasilan, dan penghitungan pajaknya setiap tahun (Kementerian Keuangan RI Direktorat Jenderal Pajak:2014:1)

Pengisian SPT Tahunan PPh pada dasarnya tidaklah sulit, yang terpenting sebelum mengisi, buku petunjuknya harus dibaca serta data yang akan dilaporkan harus dipersiapkan.

Khusus untuk Orang Pengisian SPT Tahunan PPh pada dasarnya tidaklah Pribadi, langkah-langkah yang harus dilakukan untuk mengisi SPT Tahunan PPh adalah:

##### **1. Formulir dipersiapkan.**

Formulir SPT Tahunan PPh dapat diperoleh dari Kantor Pelayanan Pajak atau Kantor Pelayanan Penyuluhan dan Konsultasi Perpajakan (KP2KP) terdekat atau dapat mengunduh (men-download) di website [www.pajak.go.id](http://www.pajak.go.id).

##### **2. Data yang akan dilaporkan dipersiapkan.**

Bagi Orang Pribadi yang memiliki penghasilan dari usaha dan/atau pekerjaan bebas, yang dipersiapkan:

- a. Neraca dan laporan laba rugi (bagi Orang

- b. Pribadi yang menyelenggarakan pembukuan) atau rekapitulasi bulanan peredaran bruto (bagi Orang Pribadi yang menyelenggarakan pencatatan);
- c. Bukti pemotongan PPh atas penghasilan yang berasal dari dalam negeri maupun luar negeri (apabila ada penghasilan yang dipotong oleh pemberi penghasilan)
- d. Rincian penghasilan selain yang berasal dari usaha dan/atau pekerjaan bebas;

Bukti pembayaran, seperti pembayaran Zakat yang dibayar ke Badan Amil Zakat atau lembaga Amil Zakat atau pembayaran Sumbangan Keagamaan yang sifatnya wajib ke lembaga yang disahkan oleh Kementerian Agama, atau pembayaran Fiskal Luar Negeri;

- a. Rincian harta dan kewajiban (hutang), misalnya untuk rumah dan tanah cukup melihat SPPT PBB-nya, untuk kendaraan lihat BPKB-nya, dan dokumen.
  - b. Data lainnya, seperti Daftar Susunan Keluarga.
3. Mengisi SPT Tahunan PPh dengan memperhatikan hal-hal sebagai berikut:
- a. Yang diisi terlebih dahulu adalah formulir
  - b. Lampiran, bukan induknya;
  - c. Di setiap lembar jangan lupa mengisi identitas seperti nama, NPWP dan tahun pajaknya;
  - d. Jangan lupa membubuhkan tanda tangan, karena jika tidak SPT yang anda laporkan dianggap tidak sah.

- e. sebelum SPT dikirim/disampaikan ke Kantor Pelayanan Pajak (KPP) atau melalui tempat lain yang ditunjuk, jika SPT menunjukkan kurang bayar, kekurangan tersebut harus dibayar paling lambat sebelum SPT disampaikan ke KPP dan bukti pembayaran tersebut dilampirkan pada SPT tersebut;
- f. pembayaran dapat dilakukan di kantor pos.

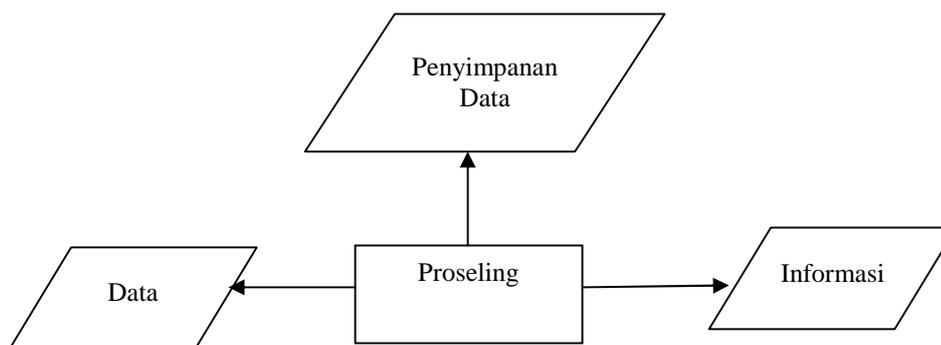
## **II.5. Wajib Pajak**

Menurut Undang-Undang Perpajakan, No.28 Tahun 2007 Tentang Ketentuan Umum dan Tata Cara Perpajakan Pasal 1, pajak adalah kontribusi wajib kepada negara yang terutang oleh orang pribadi atau badan yang bersifat memaksa berdasarkan Undang-Undang, dengan tidak mendapatkan imbalan secara langsung dan digunakan untuk keperluan negara bagi sebesar- besarnya kemakmuran rakyat. Pajak merupakan pungutan yang bersifat wajib yang dibebankan kepada masyarakat (individu dan badan) sesuai dengan kemampuan ekonomis yang dimiliki dan pajak juga merupakan peralihan kekayaan dari sektor rakyat ke sektor pemerintah untuk mem- biayai pengeluaran Negara.

## **II.6. Data**

Mengenai pengertian data, lebih jelas apa yang didefinisikan oleh Drs. Jhon J. Longkutoy dalam bukunya “ Pengenalan Komputer” sebagai berikut :  
istilah data adalah suatu istilah majemuk yang berarti fakta atau bagian dari fakta yang mengandung arti yang dihubungkan dengan kenyataan simbol-simbol,

gambar-gambar, angka-angka, huruf-huruf, atau simbol-simbol yang menunjukkan suatu ide, objek, kondisi, atau situasi dan lain-lain (Tata Sutabri, 2012 : 2).



**Gambar II.2. Pemrosesan Data**

*Sumber : Tata Sutabri, S. Kom, MM (2012 :2)*

## **II.7. Basis Data (Database)**

Basis data menurut Stephen dan Plew adalah (2000) adalah mekanisme yang digunakan untuk menyimpan informasi atau data.

Kemudian Silberchatz, dkk (2002) mendefenisikan basis data sebagai kumpulan data berisi informasi yang sesuai untuk perusahaan. Sistem manajemen basis data (DBMS) adalah kumpulan data yang saling berhubungan dan kumpulan program untuk mengakses data. Tujuan utama sistem manajemen basis data adalah menyediakan cara menyimpan dan mengambil informasi basis data secara mudah dan efisien.

Ramakrishnan dan Gehkre (2003) menyatakan basis data sebagai kumpulan data, yang umumnya mendeskripsikan aktivitas satu organisasi atau lebih yang berhubungan (Janner Simarmata, dkk; 2010 : 1).

1. Keuntungan DBMS (*Database Management System*)

DBMS memungkinkan perusahaan maupun pengguna individu untuk :

a. Mengurangi perulangan data

Apabila dibandingkan dengan *file-file* komputer yang disimpan terpisah di setiap lokasi komputer. DBMS mengurangi jumlah total *file* dengan menghapus data yang terduplikasi di berbagai *file*. Data terduplikasi selebihnya dapat ditempatkan dalam satu *file*.

b. Mencapai independensi data

Spesifikasi data disimpan dalam skema pada tiap program aplikasi. Perubahan dapat dibuat pada struktur data tanpa mempengaruhi program yang mengakses data.

c. Mengintegrasikan data beberapa *file*

Saat *file* dibentuk sehingga menyediakan kegiatan logis, maka organisasi fisik bukan merupakan kendala. Organisasi logis, pandangan pengguna, dan program aplikasi tidak harus tercemin pada media.

d. Mengambil data dan informasi dengan cepat.

Hubungan-hubungan logis, bahasa manipulasi data, serta bahasa *query* memungkinkan pengguna mengambil data dalam hitungan detik atau menit.

e. Meningkatkan keamanan

DBMS *mainframe* maupun komputer mikro dapat menyertakan beberapa lapis keamanan seperti kata sandi (*password*), direktori pemakai, dan bahasa sandi (*encryption*) sehingga data yang dikelola akan lebih aman.

2. Kerugian DBMS

Keputusan menggunakan DBMS mengikat perusahaan atau pengguna untuk :

a. Memperoleh perangkat lunak

DBMS *mainframe* masih sangat mahal. Walaupun harga DBMS berbasis komputer mikro lebih murah, tetapi tetap merupakan pengeluaran besar bagi suatu organisasi kecil.

b. Memperoleh konfigurasi perangkat keras yang besar

DBMS sering memerlukan kapasitas penyimpanan dan memori lebih besar dari pada program aplikasi lain.

c. Mempekerjakan dan mempertahankan staf DBA

DBMS memerlukan pengetahuan khusus agar dapat memanfaatkan kemampuannya secara penuh. Pengetahuan khusus ini disediakan paling baik oleh para pengguna basis data (DBA)

Baik basis data terkomputerisasi maupun DBMS bukanlah prasyarat untuk memecahkan masalah. Namun, keduanya memberikan dasar-dasar menggunakan komputer sebagai suatu sistem informasi bagi para spesialis informasi dan pengguna (Janner Simarmata, dkk; 2010 : 8-9).

### II.7.1. *Entity Relationship Diagram (ERD)*

Entity Relationship (ER) data model didasarkan pada persepsi terhadap dunia nyata yang tersusun atas kumpulan objek-objek dasar yang disebut entitas dan hubungan antar objek. Entitas adalah sesuatu objek dalam dunia nyata yang dapat dibedakan dari objek lain. Sebagai contoh, masing-masing mahasiswa adalah entitas dan mata kuliah dapat dianggap sebagai entitas.

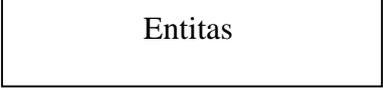
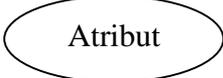
Entitas digambarkan dalam basis data dengan kumpulan atribut. Misalnya atribut nim, nama, alamat, dan kota bisa menggambarkan data mahasiswa tertentu dalam suatu universitas. Atribut-atribut membentuk entitas mahasiswa. Demikian pula, atribut kodeMK, namaMK, dan SKS mendeskripsikan mata kuliah.

Atribut NIM digunakan sebagai untuk mengidentifikasi mahasiswa secara unik karena dimungkinkan terdapat dua mahasiswa dengan nama, alamat, dan kota yang sama. Pengenal unik harus diberikan pada masing-masing mahasiswa.

Relasi adalah hubungan antara beberapa entitas. Sebagai contoh relasi menghubungkan mahasiswa dengan mata kuliah yang diambilnya. Kumpulan semua entitas bertipe sama disebut kumpulan entitas (*entitas sel*), sedangkan kumpulan semua relasi bertipe sama disebut dengan kumpulan relasi (*relationship sel*).

Struktur logis (skema database) dapat ditunjukkan secara grafis dengan diagram ER yang dibentuk dari komponen-komponen berikut dilihat pada tabel II.1.

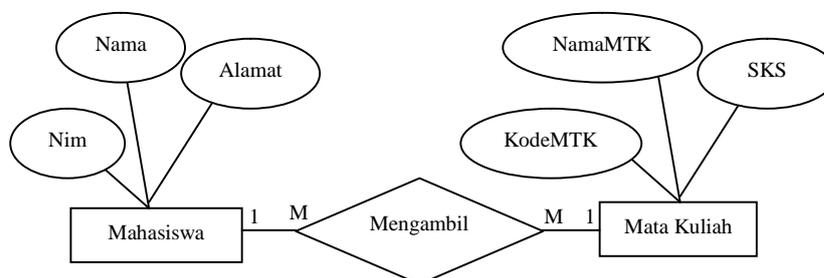
**Tabel II.1. Komponen-Komponen Diagram ER**

	Persegi Panjang mewakili kumpulan entitas
	Elips Mewakili Atribut
	Belah Ketupat Mewakili Relasi
	Garis Menghubungkan atribut dengan kumpulan entitas dan kumpulan entitas dengan relasi

*Sumber : Janner Simarmata, dkk (2010 :60)*

Masing-masing komponen diberi nama entitas atau relasi yang diwakilinya.

Sebagai ilustrasinya bayangkan anda mengambil bagian sistem basis data universitas yang terdiri dari mahasiswa dan mata kuliah. gambar II.3. menunjukkan diagram ER dari contoh. Diagram menunjukkan bahwa ada dua kumpulan entitas yaitu mahasiswa dan mata kuliah dan bahwa relasi mengambil contoh mahasiswa dan mata kuliah (Janner Simarmata; 2010 : 59-60)



**Gambar II.3. Diagram ER**

*Sumber : Janner Simarmata, dkk (2010 : 60)*

## II.7.2. Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel relasional (*www. utexas. edu*).

### 1. Bentuk Nornal Pertama (1 NF)

Contoh yang kita gunakan di sini adalah sebuah perancangan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri. Masing-masing pemasok bisa menyediakan banyak barang. Tabel relasionalnya dapat dituliskan sebagai berikut :

PEMASOK (P#, Status, Kota, b#, qty) di mana

p# : kode pemasok (kunci utama)

status : kode status kota

Kota : nama kota

b# : barang yang dipasok

qty : jumlah barang yang dipasok.

Sebuah tabel relasional secara defenisi selalu berada dalam bentuk normal pertama. Semua nilai pada kolom-kolomnya adalah atomi. Ini berarti kolom-kolom tidak mempunyai nilai berulang. Tabel II.2. menunjukkan tabel pemasok dalam 1 NF

**Tabel II.2. Normalisasi Pertama Pemasok**

<b>P#</b>	<b>Status</b>	<b>Kota</b>	<b>B#</b>	<b>Qty</b>
P1	20	Yogyakarta	B1	300
P1	20	Yogyakarta	B2	200
P1	20	Yogyakarta	B3	400
P1	20	Yogyakarta	B4	200
P1	20	Yogyakarta	B5	100
P1	20	Yogyakarta	B6	100
P2	10	Medan	B1	300
P2	10	Medan	B2	400
P3	10	Medan	B2	200
P4	20	Yogyakarta	B2	200
P4	20	Yogyakarta	B4	300
P4	20	Yogyakarta	B5	400

*Sumber : (Janner Simarmata, dkk ; 2010 :80)*

## 2. Bentuk Normal Kedua (2 NF).

Defenisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1 NF, tetapi tidak pada 2 NF, sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada 1 NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada kunci utama. Ini berarti bahwa setiap kolom bukan kunci harus tergantung pada seluruh kolom yang membentuk kunci utama. Tabel pemasok berada pada 1 NF, tetapi tidak pada 2 NF karena status dan kota tergantung secara fungsional hanya pada kolom p# dari kunci gabungan (p#, b#). Ini dapat digambarkan dengan membuat daftar ketergantungan fungsional.

P#  $\longrightarrow$  Kota, Status

Kota  $\longrightarrow$  Status

(P#, B#)  $\longrightarrow$  qty

Proses mengubah tabel 1 NF ke 2 NF adalah :

- a. tentukan sembarang kolom penentu selain kunci gabungan dan kolom-kolom yang ditentukannya.
- b. Buat dan beri nama tabel baru untuk masing-masing penentu dan kolom-kolom yang ditentukan.
- c. Pindahkan kolom-kolom yang ditentukan dari tabel asal ke tabel baru penentu akan menjadi kunci utama pada tabel baru.
- d. Hapus kolom yang baru dipindahkan dari tabel asal, kecuali penentu yang akan berfungsi sebagai kunci tamu.

- e. Tabel asal bisa diberi nama baru.

Pada contoh, kita memindahkan kolom p#, status, dan kota ke tabel baru yang disebut pemasok2. Kolom p# menjadi kunci utama tabel ini. Tabel II.3. menunjukkan hasilnya.

**Tabel II.3. Tabel Bentuk Normal Kedua (2NF).**

Pemasok2			Barang		
P#	Status	Kota	P#	B#	Qty
P1	20	Yogyakarta	P1	B1	300
P2	10	Medan	P1	B2	200
P3	10	Medan	P1	B3	400
P4	20	Yogyakarta	P1	B4	200
P5	30	Bandung	P1	B5	100
P2	10	Medan	P1	B6	100
P3	10	Medan	P2	B1	300
P4	20	Yogyakarta	P2	B2	400
P5	30	Bandung	P3	B2	200
			P4	B2	200
			P4	B4	300
			P4	B5	400

*Sumber : (Janner Simarmata, dkk ; 2010 :82).*

### 3. Bentuk Normal Ketiga (3 NF).

bentuk normal ketiga mengharuskan semua kolom pada tabel relasional hanya pada kunci utama. Secara defenisi, sebuah tabel berada pada bentuk normal ketiga (3 NF) jika tabel sudah berada pada 2 NF dan setiap kolom yang bukan kunci tidak tergantung secara transistif pada kunci utamanya. Dengan kata lain, semua atribut bukan kunci tergantung secara fungsional hanya pada kunci utama. Tabel barang sudah dalam bentuk normal ketiga. Kolom bukan kunci, qty, tergantung sepenuhnya pada kunci utama (p#,

b#). Pemasok masih berada pada 2 NF, tetapi belum berada pada 3 NF karena dia mengandung ketergantungan transitif. Ketergantungan transitif terjadi ketika sebuah kolom bukan kunci, yang ditentukan oleh kunci utama, menentukan kolom lainnya. Konsep ketergantungan transistif dapat digambarkan dengan menunjukkan ketergantungan fungsional pada pemasok2, yaitu :

Pemasok2. p#  $\longrightarrow$  Pemasok2, status

Pemasok2. p#  $\longrightarrow$  Pemasok2, kota

Pemasok2. kota  $\longrightarrow$  Pemasok2, status

Perlu dicatat bahwa pemasok2, status ditentukan, baik oleh kunci utama p#, maupun kolom bukan kunci, kota

Proses mengubah tabel menjadi 3 NF adalah :

- a. Tentukan semua penentu selain kunci utama dan kolom yang ditentukannya.
- b. Buat dan beri nama tabel baru untuk masing-masing penentu dan kolom yang ditentukannya.
- c. Pindahkan kolom yang ditentukan dari tabel asal ke tabel baru. Penentu menjadi kunci utama tabel baru.
- d. Hapus kolom yang baru saja dipindahkan dari tabel asal, kecuali penentu yang akan berfungsi sebagai kunci tamu.
- e. Tabel asal bisa diberi nama baru.

Untuk mengubah PEMASOK2 menjadi 3 NF, kita membuat tabel baru yang disebut KOTA\_STATUS dan memindahkan kolom kota dan

status ke tabel baru. Status dihapus dari tabel diberi nama baru PEMASOK\_KOTA. Tabel II.4 menunjukkan hasilnya

**Tabel II.4. Tabel Bentuk Normal Ketiga (3 NF)**

PEMASOK\_KOTA

P#	Kota
P1	Yogyakarta
P2	Medan
P3	Medan
P4	Yogyakarta
P5	Bandung

KOTA_STATUS	
Kota	Status
Yogyakarta	20
Medan	10
Bandung	30
Semarang	40

*Sumber : (Janner Simarmata, dkk ; 2010 :83).*

#### 4. Bentuk Normal Boyce Code (BCNF)

Setelah 3 NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3 NF sudah cukup karena sangat jarang entitas yang berada pada 3 NF bukan merupakan 4 NF dan 5 NF. Lebih lanjut, mereka berpendapat bahwa keuntungan yang didapat mengubah entitas ke 4 NF dan 5 NF sangat kecil sehingga tidak perlu dikerjakan. Bentuk Normal Boyce- Code (BCNF) adalah versi 3 NF lebih teliti dan berhubungan dengan tabel relasional yang mempunyai (a) banyak kunci kandidat (b) kunci kandidat gabungan, dan (c) kunci kandidat yang saling tumpang tindih.

BCNF didasarkan pada konsep penentu. Sebuah kolom penentu adalah kolom di mana kolom-kolom lain sepenuhnya tergantung secara fungsional. Sebuah tabel relasional berada pada BCNF jika dan hanya jika setiap penentu adalah kunci kandidat.

#### 5. Bentuk Normal Keempat (4 NF)

Sebuah tabel relasional berada pada bentuk normal keempat (4 NF) jika dia dalam BCNF dan semua ketergantungan multivalued merupakan ketergantungan fungsional.

Bentuk normal keempat (4 NF) didasarkan pada konsep ketergantungan multivalued (MVD). Sebuah ketergantungan multivalued terjadi ketika dalam sebuah tabel relasional yang mengandung setidaknya tiga kolom, satu kolom mempunyai banyak baris bernilai sama, tetapi kolom lain bernilai berbeda.

Defenisi secara formal diberikan oleh CJ. Date, yaitu :

Misalnya, ada sebuah tabel relasional R dengan kolom A, B dan C, Maka  $R.A \twoheadrightarrow R.B$  (kolom A menentukan kolom B).

Adalah benar jika dan hanya jika himpunan nilai B yang cocok dengan pasangan nilai A dan nilai C pada R hanya tergantung pada nilai A dan tidak tergantung pada nilai C.

MVD selalu terjadi dalam pasangan, yaitu  $R.A \twoheadrightarrow R.B$  dipenuhi jika dan hanya jika  $R.A \twoheadrightarrow R.C$  dipenuhi pula.

6. Bentuk Normal Kelima (5 NF).

Sebuah tabel berada pada bentuk normal kelima jika dia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil.

Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*). Ketergantungan gabungan berarti sebuah tabel, setelah deskomposisi menjadi tiga atau lebih tabel yang lebih kecil, harus dapat digabungkan kembali untuk membentuk tabel asal. Dengan kata lain 5 NF menunjukkan ketika sebuah tabel tidak dapat dideskomposisi lagi (Janner Simarmata; 2012 : 77 - 86)

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

UML singkatan dari *Unified Modelling Language* 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 lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasikan 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 mensupport 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.8.1. 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 dikelompokkan 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 umum dijumpai pada pemodelan sistem berorientasi 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 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 (*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.

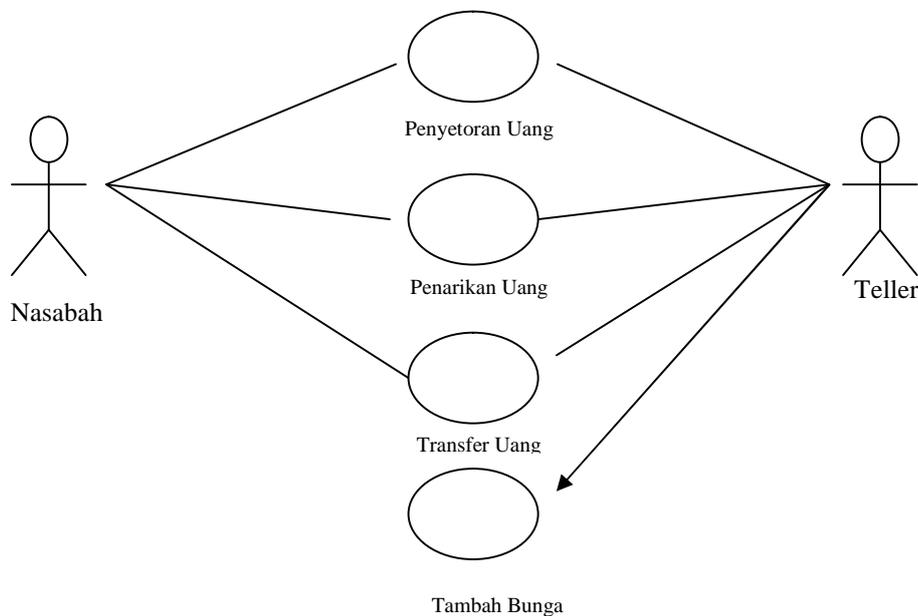
### 1. *Diagram Use Case (use case diagram)*

*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, diagram tidak identik dengan model karena model lebih luas dari diagram.

Komponen pembentuk diagram *use case* adalah :

- Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam sistem.
- Use Case*, aktivitas/ sarana yang disediakan oleh bisnis/sistem.
- Hubungan (*Link*), aktor mana saja yang terlibat dalam *use case* ini.

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

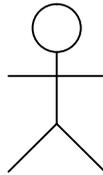


**Gambar II.4. Diagram Use Case**

*Sumber : Probowo Pudjo Widodo (2011:17)*

## 2. Aktor

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

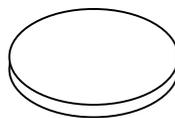


**Gambar II.5. Aktor**

*Sumber : Probowo Pudjo Widodo (2011:17)*

## 3. 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*

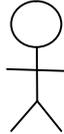
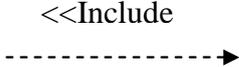


**Gambar II.6. Simbol Use Case**

*Sumber : Probowo Pudjo Widodo (2011:22)*

Berikut Simbol-simbol dalam *Use Case*, yaitu :

**Tabel II.5 Tabel Simbol-simbol Use case**

NAMA	KETERANGAN	SIMBOL
<i>Actor</i>	Seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang kita kembangkan	
<i>Use Case</i>	Peringkat Tertinggi dari fungsional yang dimiliki sistem.	
<i>Relasi Asosiasi</i>	Relasi yang terjadi antara aktor dengan use case biasanya berupa asosiasi.	
<i>Include Relationship</i>	Relasi cakupan memungkinkan suatu use case untuk menggunakan fungsionalitas yang disediakan oleh use case lainnya.	

<p><b><i>Extends Relationship</i></b></p>	<p>Memungkinkan suatu use case memiliki kemungkinan untuk memperluas fungsional yang disediakan use case yang lainnya.</p>	<p>&lt;&lt;Extends -----&gt;</p>
---	--	--------------------------------------

*Sumber : Adi Nugroho( 2005:49)*

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

**4. Diagram Kelas (*Class Diagram*)**

Diagram kelas mempunyai dua jenis yaitu *domain class diagram* dan *design class diagram*. Fokus *domain class diagram* adalah pada sesuatu dalam lingkungan kerja pengguna, bukan pada *class* perangkat lunak yang nantinya

akan anda rancang. Sedangkan *design class diagram* tujuannya adalah untuk mendokumentasikan dan menggambarkan kelas-kelas dalam pemrograman yang nantinya akan dibangun.



**Gambar II.7. Notasi Domain Diagram Class**

*Sumber : E. Triandini dan G. Suardika (2012 : 49 -50)*



**Gambar II.8. Notasi Design Diagram Class**

*Sumber : E. Triandini dan G. Suardika (2012 : 49 -50)*

## 5. Diagram Aktivitas (*Activity Diagram*)

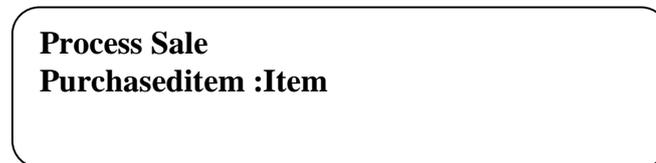
Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan bisnis juga. Diagram aktivitas menunjukkan aktivitas sistem dalam kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software*, 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 melakukan langkah 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 *classifier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classifier*, 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

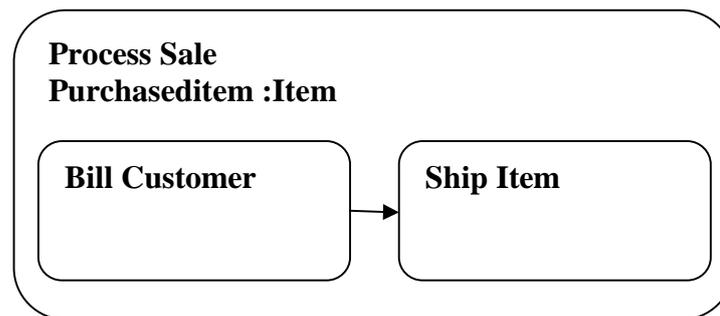
diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya *specific* dan digunakan hanya untuk aktivitas tertentu.



**Gambar II.9. Aktivitas sederhana tanpa rincian**

*Sumber : Probowo Pudjo Widodo (2011:145)*

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



**Gambar II.10. Aktivitas dengan detail rincian**

*Sumber : Probowo Pudjo Widodo (2011:145)*

Berikut komponen-komponen dalam *activity diagram*, yaitu :

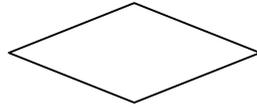
- a. *Action State*, adalah langkah-langkah dalam sebuah *activity*. *Action* bisa terjadi saat memasuki *activity*, meninggalkan *activity*, atau pada *event* yang spesifik.



**Gambar II.11. Action State**

*Sumber : Probowo Pudjo Widodo (2011:145)*

- b. *Decision*, menunjukkan dimana sebuah keputusan perlu dibuat dalam aliran kerja.



**Gambar II.12. *Decision***

*Sumber : Probowo Pudjo Widodo (2011:145)*

- c. *Transition*, menunjukkan bagaimana aliran kerja itu berjalan dari satu aktivitas ke aktivitas lainnya.



**Gambar II.13. *Transition***

*Sumber : Probowo Pudjo Widodo (2011:145)*

- d. *Synchronization*, menunjukkan dua atau lebih langkah dalam aliran kerja berjalan secara serentak a *Fork and Join*.



**Gambar II.14. *Synchronization***

*Sumber : Probowo Pudjo Widodo (2011:145)*

- e. *Swimlane*, menunjukkan siapa yang bertanggung jawab melakukan aktivitas dalam suatu diagram.



**Gambar II.15. Swimlane**

*Sumber : Probowo Pudjo Widodo (2011:145)*

- f. *Start State*, memperlihatkan dimana aliran kerja berawal.



**Gambar II.16. Start State**

*Sumber : Adi Nugroho( 2005:66)*

- g. *End State*, memperlihatkan dimana aliran kerja berakhir.



**Gambar II.17. End State**

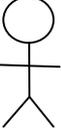
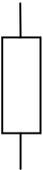
*Sumber : Adi Nugroho( 2005:66)*

## **6. Sequence Diagram**

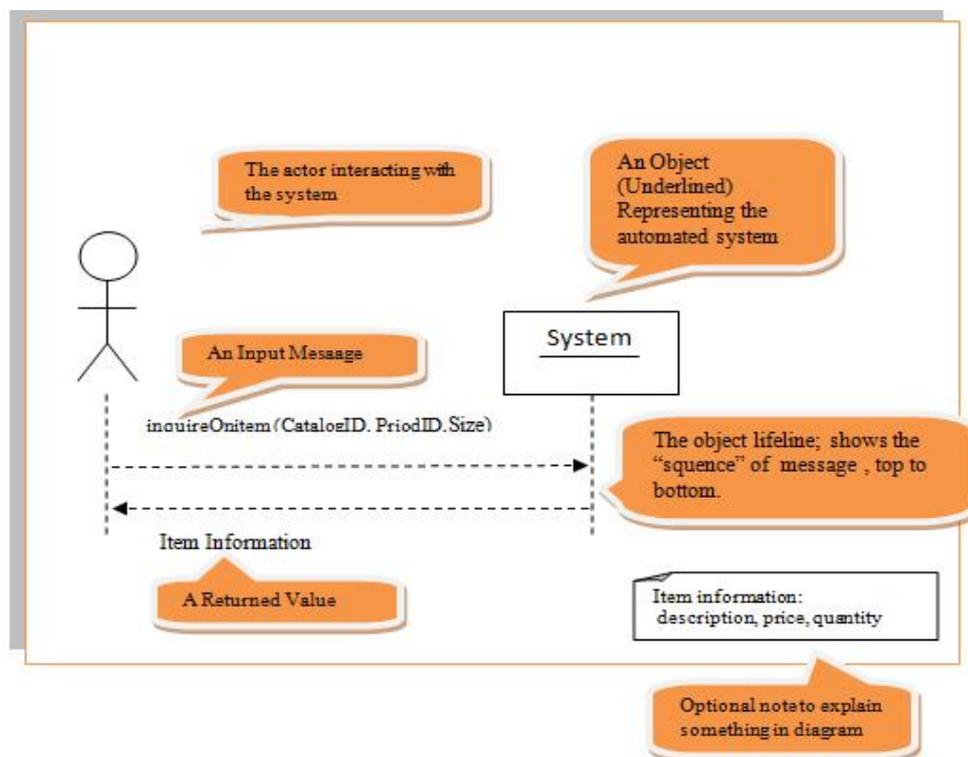
Menurut John Satzinger, 2010, dalam buku *System Analysis and Design in a Changing World*, “System Sequence Diagram (SSD) adalah diagram yang digunakan untuk mendefinisikan input dan output serta urutan interaksi antara pengguna dan sistem untuk sebuah use case (E. Triandini dan G. Suardika ; 2012 : 71).

Berikut adalah notasi-notasinya :

**Tabel II.6. Notasi Sequence Diagram**

<b>Object</b>	<i>Object</i> merupakan instance dari sebuah <i>class</i> dan dituliskan tersusun secara horizontal. Digambarkan sebagai sebuah <i>class</i> (kotak) dengan nama obyek di dalamnya yang diawali dengan sebuah titik koma.	
<b>Actor</b>	<i>Actor</i> juga dapat berkomunikasi dengan objek, maka <i>actor</i> juga dapat diurutkan sebagai kolom. Simbol <i>Actor</i> sama dengan simbol pada <i>Actor Use Case Diagram</i> .	
<b>Lifeline</b>	<i>Lifeline</i> mengindikasikan keberadaan sebuah objek dalam basis waktu. Notasi untuk <i>Lifeline</i> adalah garis putus-putus vertikal yang ditarik dari sebuah obyek.	
<b>Activation</b>	<i>Activation</i> dinotasikan sebagai kotak segi empat yang digambar pada sebuah <i>Lifeline</i> . <i>Activation</i> mengindikasikan sebuah obyek yang akan melakukan sebuah aksi.	
<b>Message</b>	<i>Message</i> digambarkan dengan anak panah horizontal antara <i>activation</i> . <i>Message</i> mengindikasikan komunikasi antara <i>object-object</i> .	

*Sumber : Adi Nugroho (2005 : 92)*



**Gambar II.18. Notasi Sequence Diagram**

*Sumber : Evi Triandini dan Gede Suardika (2012 : 71)*

## II.9. PHP

Menurut kamus komputer, PHP adalah bahasa pemrograman untuk dijalankan melalui halaman web, umumnya digunakan untuk mengolah informasi di internet. Sedangkan dalam pengertian lain, PHP adalah singkatan dari PHP *Hypertext Preprocessor* yaitu bahasa pemrograman web *server-side* yang bersifat *open source* atau gratis. PHP merupakan *script* yang menyatu dengan HTML dan berada pada server (*server side HTML embedded scripting*) (Rulianto Kurniawan; 2010: 2)

## **II.10. HTML**

HTML (*Hyper Text Markup Language*) adalah bahasa yang digunakan pada dokumen web sebagai bahasa untuk pertukaran dokumen web. HTML adalah dokumen *text* yang bisa dibaca untuk dipublikasikan di World Wide Web (WWW), dan semua nama file dokumen HTML mempunyai ekstensi html atau htm (Sibero; 2012:9).

## **II.11. Macromedia Dreamweaver 8**

Pengertian Macromedia Dreamweaver adalah sebuah software HTML editor profesional yang digunakan untuk mendesain secara visual dan mengelola situs web maupun halaman web. Bilamana kita menyukai untuk berurusan dengan kode-kode HTML secara manual atau lebih menyukai bekerja dengan lingkungan secara visual dalam melakukan editing, Dreamweaver membuatnya menjadi lebih mudah dengan menyediakan tools yang sangat berguna dalam peningkatan kemampuan dan pengalaman kita dalam mendesain web. Dreamweaver 8 dalam hal ini digunakan untuk web desain. Dreamweaver 8 mengikutsertakan banyak tools untuk kode-kode dalam halaman web beserta fasilitas-fasilitasnya, antara lain : Referensi HTML, CSS dan Javascript, Javascript debugger, dan editor kode ( tampilan kode dan Code inspector) yang mengizinkan kita mengedit kode Javascript, XML, dan dokumen teks lain secara langsung dalam Dreamweaver.

Teknologi Dreamweaver Roundtrip HTML mampu mengimpor dokumen HTML tanpa perlu memformat ulang kode tersebut dan kita dapat menggunakan Dreamweaver pula untuk membersihkan dan memformat ulang HTML bila kita

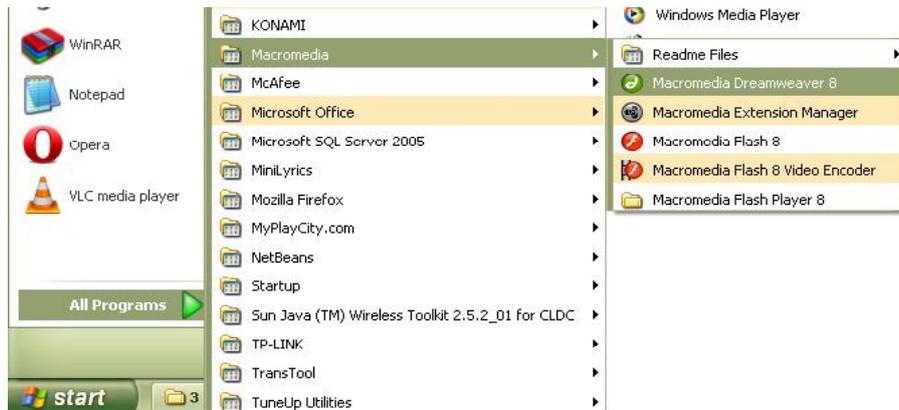
menginginkannya. Selain itu Dreamweaver juga dilengkapi kemampuan manajemen situs, yang memudahkan kita mengelola keseluruhan elemen yang ada dalam situs. Kita juga dapat melakukan evaluasi situs dengan melakukan pengecekan broken link, kompatibilitas browser, maupun perkiraan waktu download halaman web. Macromedia Dreamweaver 8, Evolusi Macromedia DreamWeaver diantaranya sebagai berikut:

DreamWeaver 3 : merupakan versi lamanya yaitu web authoring terbaik pada tahun 2000. DreamWeaver 4 : merupakan software yang lebih baik lagi dan lebih canggih pada tahun 2002. Macromedia DreamWeaver MX: pertama kali di tampilkan pada tahun 2004, selain software untuk mendesain web, juga bisa untuk menyunting kode dan untuk membuat aplikasi web dengan menggunakan bahasa pemrograman JPS, PHP, ASP atau Coldfusion. Di lengkapi dengan fasilitas yang cukup lengkap untuk manajemen situs. Yang terbaru ada Macromedia DreamWeaver 8 MX, adobe DreamWeaver cs3 dan cs4.

### **II.11.1. Menjalankan Macromedia Dreamweaver 8**

Untuk memulai pembuatan program dengan menggunakan Macromedia Dreamweaver 8 lakukan langkah-langkah sebagai berikut :

1. Klik tombol start pada taskbar.
2. Pilih program > Macromedia, kemudian klik pada Macromedia Dreamweaver 8.



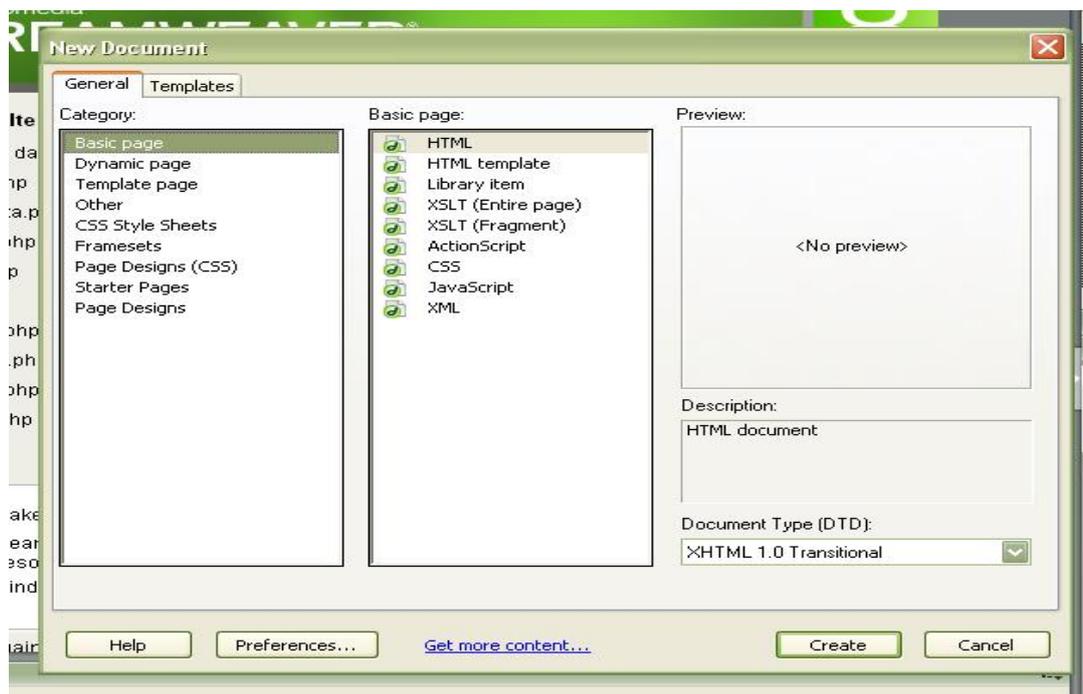
**Gambar II.19. Macromedia Dreamweaver 8 pada start menu program**

3. Beberapa saat kemudian muncul tampilan seperti gambar di bawah in,



**Gambar II.20. Tampilan awal Macromedia Dreamweaver 8**

4. Kemudian Klik Menu File > New, Maka akan muncul tampilan di bawah ini :



**Gambar II.21 Lingkungan Macromedia Dreamweaver 8**

*Macromedia Dreamweaver 8* secara default akan menampilkan *File* baru dengan Basic Page *Html*. Bagi pemula disarankan menggugurkan *Html*, klik *Create* untuk membuat area kerja baru.

## II.12. My SQL

MySQL adalah salah satu jenis database server yang sangat terkenal. MySQL termasuk jenis RDBMS (*Relational Database Management System*). MySQL ini mendukung bahasa pemrograman PHP. MySQL juga mempunyai *query* atau bahasa SQL (*Structured Query Language*) yang simpel dan menggunakan *escape character* yang sama dengan PHP (Rulianto Kurniawan; 2010: 16).