

BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem adalah sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisir, saling berinteraksi, saling bergantung satu sama lain, dan terpadu.

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; 2005 : 2)

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 bekerja 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; 2005 :11-12).

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 pandangannya 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; 2005 : 13).

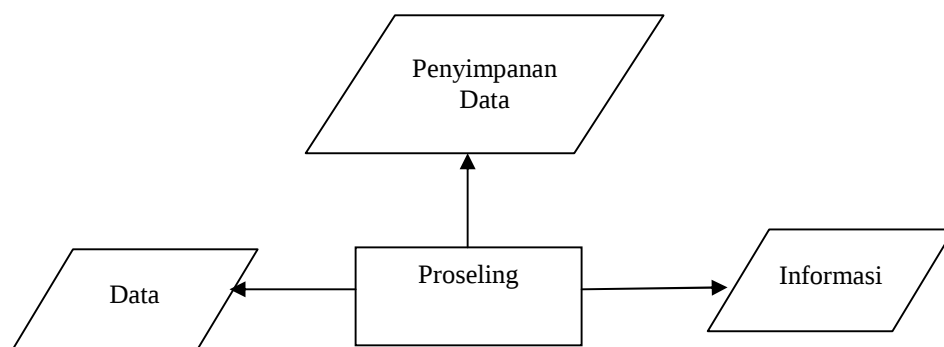
II.3. Informasi

Informasi adalah data yang telah diklasifikasikan diolah atau diinterpretasi untuk digunakan dalam proses pengambil keputusan. Sistem pengolahan informasi megolah 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; 2005 : 23).

II.3.1 Data

Mengenai pengertian data, lebih jelas apa yang didefenisikan oleh Drs. Jhon J. Longkutoy dalam bukunya “ Pengenalan Komputer” sebagai berikut : isitilah 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 menunjukan suatu ide, objek, kondisi, atau situasi dan lain-lain (Tata Sutabri, 2005 : 16).



Gambar II.1. Pemrosesan Data

Sumber : Tata Sutabri (2005 :16)

II.3.2. 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; 2005 :42)

II.3.2. 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 dimasukan, 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 efesienan, 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; 2005 :42-43).

II.4. **Produksi**

Produksi adalah suatu kegiatan yang mengubah input menjadi output. Kegiatan tersebut dalam ekonomi biasa di nyatakan dalam fungsi produk, Fungsi produk menunjukkan jumlah maksimum output yang dapat dihasilkan dari pemakaian sejumlah input dengan menggunakan teknologi tertentu. Produksi sering didefinisikan sebagai penciptaan guna, dimana guna berarti kemampuan barang atau jasa untuk memenuhi kebutuhan manusia. Menurut Sukirno (2006:6) pengertian faktor produksi adalah benda-benda yang disediakan oleh alam atau diciptakan oleh manusia yang dapat digunakan untuk memproduksi barang dan jasa. Produksi molto yang optimal adalah produksi yang mendatangkan produk yang menguntungkan ditinjau dari sudut ekonomi ini berarti biaya faktor-faktor input yang berpengaruh pada produksi jauh lebih kecil bila dibandingkan dengan hasil yang diperoleh sehingga molto dapat memperoleh keuntungan dari usaha produksinya. (*Nova Tumoka ; 2013 : 346*)

11.4.1 Laporan produksi

Produksi memiliki pengertian kegiatan yang dilakukan untuk menciptakan suatu barang atau menambah nilai suatu barang sehingga memiliki manfaat yang lebih untuk memenuhi kebutuhan hidup manusia. Bahkan tidak jarang, perusahaan yang mengembangkan atau memproduksi barang atau jasa lain untuk mempertahankan eksistensinya dalam dunia perdagangan di Indonesia. Untuk mempertanggung jawabkannya, setiap perusahaan diwajibkan untuk membuat laporan produksi.

a. Laporan Bahan Baku

Bahan baku adalah bahan dasar dari suatu produk. Bahan baku menjadi bahan utama dari suatu produk. Karena itu, kuantitas bahan baku yang dibutuhkan untuk menghasilkan produk selalu sebanding dengan kuantitas produk yang dihasilkan. Itu sebabnya, biaya bahan baku langsung memiliki sifat sebagai variabel, karena selalu berfluktuasi sejalan dengan perubahan volume produksi.

Dalam laporan harga pokok produksi harus tercakup tentang laporan biaya bahan baku, yaitu ;

- Persediaan awal bahan baku
- Persediaan bahan baku
- Persediaan total bahan baku
- Persediaan akhir bahan baku
- Biaya bahan baku
- Biaya tenaga kerja langsung

- Biaya overhead
- Biaya produksi

b. Laporan Biaya Produksi

Menurut Muhadi dan Joko Siswanto (2001) istilah laporan biaya produksi sering disebut pula laporan harga pokok produksi. Laporan biaya produksi memuat tiga bagian pokok.

1. Data produksi (skedul produksi), pada bagian ini dilaporkan mengenai perincian jumlah produk yang masuk proses, jumlah produk selesai, dan jumlah produk dalam proses awal dan akhir dari satu periode.
2. Biaya yang dibebankan, pada bagian ini dilaporkan mengenai perincian pembebanan biaya persataun produk yang meliputi biaya bahan, biaya tenaga kerja, dan biaya overhead pabrik.
3. Perhitungan harga pokok, pada bagian ini dilaporkan mengenai harga pokok produk selesai yang ditransfer ke departemen produksi terusannya atau ke gudang produk selesai.

c. Laporan Tenaga Kerja

Tenaga Kerja adalah seluruh jumlah penduduk yang dianggap dapat bekerja dan sanggup bekerja jika ada permintaan kerja, tenaga kerja dapat dilihat dari konsep produktivitasnya. Sumber: socialrewardsurvey

Tenaga kerja faktor produksi ini bukan saja berarti jumlah buruh yang terdapat dalam perekonomian. Pengertian tenaga kerja meliputi juga keahlian dan ketrampilan yang mereka miliki. Dari segi keahlian dan pendidikannya, tenaga kerja di bedakan kepada tiga golongan berikut:

1. Tenaga kerja kasar adalah tenaga kerja yang tidak berpendidikan atau rendah pendidikannya dan tidak memiliki keahlian dalam suatu bidang pekerjaan,
2. Tenaga kerja terampil adalah tenaga kerja yang memiliki keahlian dari pelatihan atau dari pengalaman kerja seperti montir mobil, tukang kayu dan ahli mereparasi TV dan radio.
3. Tenaga kerja terdidik adalah tenaga kerja yang memiliki pendidikan cukup tinggi dan ahli dalam bidang tertentu seperti dokter, akuntan, ahli ekonom dan insinyur. (Sudono Sukirno, 2003: hlm 7), Adapun factor – factor yang mempengaruhi harga pokok produksi, yaitu;

1. Full Costing

Full costing merupakan metode penentuan harga pokok produksi yang memperhitungkan semua unsur biaya produksi ke dalam harga pokok produksi yang terdiri dari biaya bahan baku, biaya tenaga kerja langsung dan biaya overhead pabrik baik yang berperilaku variabel maupun tetap. Menurut LM Samryn (2001), *full costing* adalah : “*Full costing* adalah metode penentuan harga pokok yang memperhitungkan semua biaya produksi yang terdiri dari biaya bahan baku, biaya tenaga kerja, dan *overhead* tanpa memperhatikan perilakunya.”

Pendekatan *full costing* yang biasa dikenal sebagai pendekatan tradisional menghasilkan laporan laba rugi dimana biaya-biaya diorganisir dan disajikan berdasarkan fungsi-fungsi produksi, administrasi dan penjualan. Laporan laba rugi yang dihasilkan dari pendekatan ini banyak digunakan untuk memenuhi pihak luar perusahaan, oleh karena itu sistematikanya harus disesuaikan dengan prinsip akuntansi yang berlaku umum untuk menjamin informasi yang tersaji dalam laporan tersebut.

2. Variabel Costing

Variabel costing merupakan metode penentuan harga pokok produksi yang hanya memperhitungkan biaya produksi yang berperilaku variabel ke dalam harga pokok produksi yang terdiri dari biaya bahan baku, biaya tenaga kerja langsung dan biaya *overhead* pabrik variabel. Dalam pendekatan ini biaya-biaya yang diperhitungkan sebagai harga pokok adalah biaya produksi variabel yang terdiri dari biaya bahan baku, biaya tenaga kerja langsung dan biaya *overhead* pabrik variabel. Biaya-biaya produksi tetap dikelompokkan sebagai biaya periodik bersama-sama dengan biaya tetap non produksi. Menurut Mas'ud Machfoed (1993), *variabel costing* adalah “ Suatu metode penentuan harga pokok dimana biaya produksi variabel saja yang dibebankan sebagai bagian dari harga pokok.”

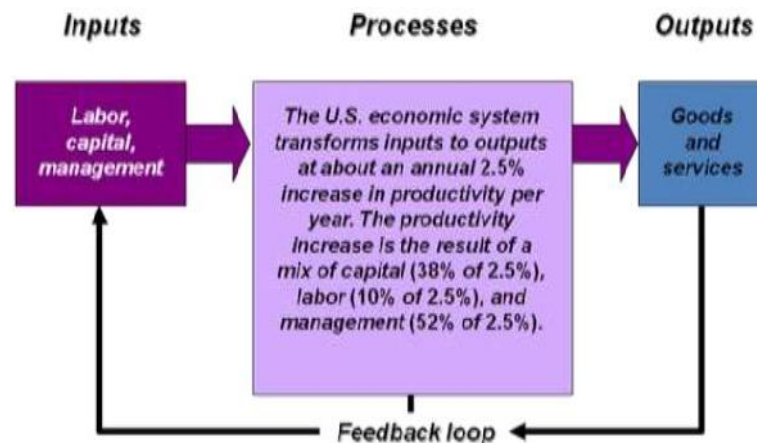
3. Overhead Pabrik

Biaya *overhead* pabrik disebut juga biaya produk tidak langsung, yaitu kumpulan dari semua biaya untuk membuat suatu produk selain biaya bahan baku langsung dan tidak langsung.

Overhead pabrik pada umumnya didefinisikan sebagai bahan tidak langsung, pekerja tidak langsung, dan bahan pabrik lainnya yang tidak secara mudah diidentifikasi atau dibebankan langsung ke pekerjaan produk atau tujuan akhir biaya.

Biaya *overhead* pabrik (FOH) terdiri dari biaya FOH tetap dan biaya FOH variabel. Biaya tetap adalah biaya yang jumlah totalnya tetap untuk tingkat volume kegiatan tertentu, biaya variabel adalah biaya yang jumlah totalnya berubah sebanding dengan perubahan volume kegiatan.

Ada juga yang dinamakan biaya semi variabel adalah biaya yang berubah tak sebanding dengan perubahan volume kegiatan.



Gambar II.1. Proses Transformasi
 Sumber : (F. Priyo Suprobo1 ; 2013 : 8)

II.5. 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.

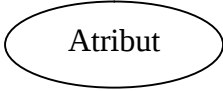
Entitas digambarkan dalam basis data dengan kumpulan atribut. Misalnya atribut nim, nama, alamat, dan kota bisa menggambarkan data mahasiswa tertentu dalam suatu universitas. Atribut-atribut membentuk entitas mahasiswa. Demikian pula, atribut kodeMK, namaMK, dan SKS mendeskripsikan mata kuliah.

Atribut NIM digunakan sebagai untuk mengidentifikasikan 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 pada dilihat pada tabel II.1.

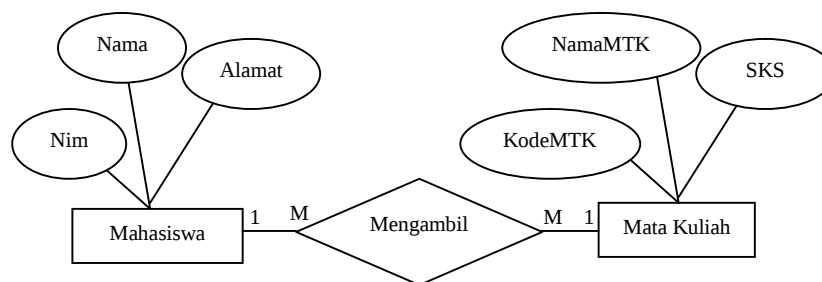
Tabel II.1. Komponen-Komponen Diagram ER

	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 (2006 :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.2. 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; 2006 : 59-60).



Gambar II.3. Diagram ER

Sumber : Janner Simarmata, dkk (2006 : 60)

II.5.1. Kamus Data

Kamus data (KD) atau data dictionary (DD) yang disebut juga dengan system dictionary data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan demikian KD, analisis sistem dapat mendefenisikan data yang mengalir di sistem dengan lengkap. KD dibuat pada pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis, KD dapat digunakan sebagai alat komunikasi antara analisis sistem dengan pemakai sistem tentang data yang mengalir di sistem. Pada tahap perancangan sistem KD digunakan untuk merancang input, merancang laporan-laporan dan database. KD dibuat berdasarkan arus data yang ada di DAD. Arus data di DAD sifatnya global, hanya

ditunjukkan nama arus datanya saja (Prof. Dr. Jogiyanto HM, MBA, Akt; 2005 : 725).

II.5.2. Isi Kamus Data

KD harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya. Untuk maksud keperluan ini maka KD harus memuat hal-hal berikut ini :

1. Nama Arus Data

Karena KD dibuat berdasarkan arus data yang mengalir di DAD, maka nama dari arus data juga harus dicatat di KD, sehingga mereka yang membaca DAD dan memerlukan penjelasan lebih lanjut tentang suatu arus data tertentu di DAD dapat langsung mencarinya dengan mudah di KD.

2. Alias

Alias atau nama lain dari data dapat dituliskan bila nama lain ini ada. Alias perlu ditulis karena data yang mempunyai nama yang berbeda untuk orang atau departemen satu dengan yang lainnya. Misalnya bagian pembuat faktur dan langganan menyebut bukti penjualan sebagai faktur, sedang bagian gudang menyebutnya sebagai tembusan permintaan persediaan barang. Baik faktur dan tembusan permintaan persediaan ini mempunyai struktur data yang sama tetapi mempunyai struktur yang berbeda.

3. Bentuk data

Telah diketahui bahwa arus data mengalir.

- a. Dari kesatuan luar ke suatu proses, data yang mengalir ini biasanya tercatat di suatu dokumen atau formulir.

- b. Hasil dari suatu proses ke kesatuan luar, data yang mengalir biasanya terdapat di media laporan atau *query* tampilan layar atau dokumen hasil cetakan komputer.
- c. Hasil dari suatu proses yang lain, data yang mengalir biasanya dalam bentuk variabel atau parameter yang dibutuhkan oleh proses penerimanya.
- d. Hasil dari suatu proses yang direkamkan ke simpanan data, data yang mengalir ini biasanya berbentuk suatu variabel.
- e. Dari simpanan data dibaca oleh suatu proses, data yang mengalir ini biasanya berupa suatu *field* (item data).

Dengan demikian bentuk dari data yang mengalir dapat berupa :

- 1. Dokumen dasar atau formulir
- 2. Dokumen hasil cetakan komputer
- 3. Laporan tercetak.
- 4. Tampilan dilayar monitor.
- 5. Variabel.
- 6. Parameter
- 7. *Field*

Bentuk dari data ini perlu dicatat di KD, karena dapat digunakan untuk mengelompokkan KD ke dalam kegunaanya, sewaktu perancangan sistem KD yang mencatat data yang mengalir dalam bentuk dokumen dasar atau formulir yang digunakan untuk merancang bentuk input sistem. KD yang mencatat data yang mengalir dalam bentuk laporan tercetak dan dokumen hasil cetakan komputer akan digunakan untuk merancang *output* yang akan dihasilkan oleh sistem. KD yang mencatat data yang mengalir dalam bentuk tampilan layar monitor akan digunakan juga untuk merancang tampilan layar yang akan

dihasilkan oleh sistem. KD yang mencatat data yang mengalir ke dalam bentuk parameter dan variabel akan digunakan untuk merancang proses dari program. KD yang mencatat data yang mengalir ke dalam bentuk dokumen, formulir, laporan, dokumen cetakan komputer, tampilan di layar monitor, variabel, dan *field* akan digunakan untuk merancang database.

II.5.3. Normalisasi

Normalisasi adalah suatu teknik yang menstrukturkan data dalam cara tertentu untuk membantu mengurangi atau mencegah timbulnya masalah yang berhubungan dengan pengolahan data dalam database. Proses normalisasi menghasilkan struktur *record* yang konsisten secara logis yang mudah dimengerti dan sederhana dalam pemeriksaannya. Beberapa level normalisasi dapat dijelaskan dan kriteria yang mendefenisikan level pada normalisasi adalah bentuk normal (*norm form*). Proses normalisasi merupakan proses pengkelompokan data elemen menjadi tabel yang menunjukkan *entity* dan relasinya. Pada proses normalisasi selalu diuji pada beberapa kondisi. Apakah ada kesulitan pada saat menambah/*insert*, menghapus/*delete*, mengubah/*update*, membaca.*retrive*, pada satu database. Bila ada kesulitan pada pengujian tersebut, maka relasi tersebut dipecahkan pada beberapa tabel lagi atau dengan kata lain perancangan belumlah mendapat database yang optimal. Pada proses normalisasi ini perlu dikenal dulu definisi dari tahap normalisasi yaitu :

a. Bentuk Tidak Normal (*Unnormalized Form*)

Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti suatu format tertentu, dapat saja data tidak lengkap

atau terduplikasi. Data dikumpulkan apa adanya sesuai dengan kedatangannya.

b. Bentuk Normal Kesatu (1 NF/ *First Normal Form*)

Bentuk normal kesatu mempunyai ciri; setiap data dibentuk dalam *flat file* (file datar/rata), data dibentuk dalam satu *record* demi *record* dan nilai dari *field* berupa "*atomic value*". Tidak ada set atribut yang berulang atau atribut nilai ganda (*multivalued*). Tiap *field* hanya satu pengertian, bukan merupakan kumpulan kata yang mempunyai arti ganda, hanya satu arti saja dan juga bukan pecahan kata sehingga artinya lain. Atom adalah zat terkecil yang masih memiliki sifat induknya, bila dipecah lagi, maka ia tidak memiliki sifat induknya.

c. Bentuk Normal Kedua (2 NF/ *Second Normal Form*)

Bentuk normal kedua mempunyai syarat; data telah memenuhi kriteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama / *primary key* sehingga untuk membentuk normal kedua sudah ditentukan kunci *field*. Kunci *field* haruslah unik dan dapat mewakili atribut lain yang menjadi anggotanya.

d. Bentuk Normal Ketiga (3 NF/ *Third Normal Form*)

Untuk menjadi normal ketiga, relasi harus dalam bentuk normal kedua, dan semua atribut bukan primer tidak punya hubungan yang transitif. Dengan kata lain, setiap atribut bukan kunci haruslah bergantung hanya pada *primary key* secara menyeluruh. Contoh pada bentuk kedua di atas

termasuk juga bentuk normal ketiga seluruh atribut yang ada di situ bergantung penuh pada kunci primernya.

e. Boycee- Cood Normal Form (BCNF)

Boycee-cood normal form mempunyai paksaan yang lebih kuat dari pada normal ketiga. Untuk menjadi BCNF, relasi harus dalam bentuk normal kesatu dan setiap atribut harus bergantung fungsi pada *atribut superkey* (Tata Sutabri; 2005 : 181-182).

Contoh :

1. Bentuk Normalisasi Pertama (1 NF)

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 definisi 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.1. Normalisasi Pertama Pemasok

P#	Status	Kota	B#	Qty
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 ; 2006 :80).

2. Normalisasi Bentuk Kedua (2 NF)

Pada contoh, kita memindahkan kolom p#, status, dan kota ke tabel baru yang disebut pemasok2. Kolom p# menjadi kunci utama tabel ini. Tabel II.2. menunjukkan hasilnya.

Tabel II.2. 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
			P1	B6	100
			P2	B1	300
			P2	B2	400
			P3	B2	200
			P4	B2	200
			P4	B4	300
			P4	B5	400

Sumber : (Janner Simarmata, dkk ; 2006 :82).

3. Normalisasi Bentuk Ketiga (3 NF)

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.3 menunjukkan hasilnya

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

PEMASOK_KOTA		KOTA_STATUS	
P#	Kota	Kota	Status
P1	Yogyakarta	Yogyakarta	20
P2	Medan	Medan	10
P3	Medan	Bandung	30
P4	Yogyakarta	Semarang	40
P5	Bandung		

Sumber : (Janner Simarmata, dkk ; 2006 :83).

II.5.4. Basis Data (Database)

Berikut ini pengertian database yang diberikan James Martin dalam bukunya ” *Database Organization*” sebagai berikut :

Database adalah suatu kumpulan data terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data (*cotrolled redudancy*) dengan cara tertentu sehingga mudah digunakan atau ditampilkan kembali; dapat digunakan oleh satu atau lebih program aplikasi secara optimal; data disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya; data disimpan sedemikian dan modifikasi dapat dilakukan dengan mudah dan terkontrol (Tata Sutabri; 2005 : 161)

II.6. Unified Modeling Language (UML)

UML singkatan dari *Unified Modelling Langguage* yang berarti bahasa pemodelan standart. (Chonoles; 2003 : 6) mengatakan sebagai bahasa, berarti *UML* memiliki sintaks dan *semantic*. Ketika kita membuat model menggunakan konsep *UML* ada aturan –aturan yang harus diikuti. Bagaimana elemen pada

model-model yang kita buat harus berhubungan satu dengan lainnya harus mengikuti standart yang ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi error yang terjadi? Bagaimana keamanan terhadap sistem yang ada kita buat? Dan sebagainya dapat dijawab dengan *UML*.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

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

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

Blok pembangunan utama *UML* adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan 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 Dan Herlawati; 2011 : 6-7).

II.6.1. 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*).

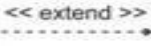
Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Pada *UML* dimungkinkan kita menggunakan diagram-diagram lainnya misalnya *Data Flow Diagram*, *Entity Relationship Diagram* dan sebagainya (Prabowo Pudjo Widodo, Dan Herlawati; 2011 : 10-12).

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 indentik dengan model karena model lebih luas dari diagram.

komponen pembentuk diagram *use case* adalah :

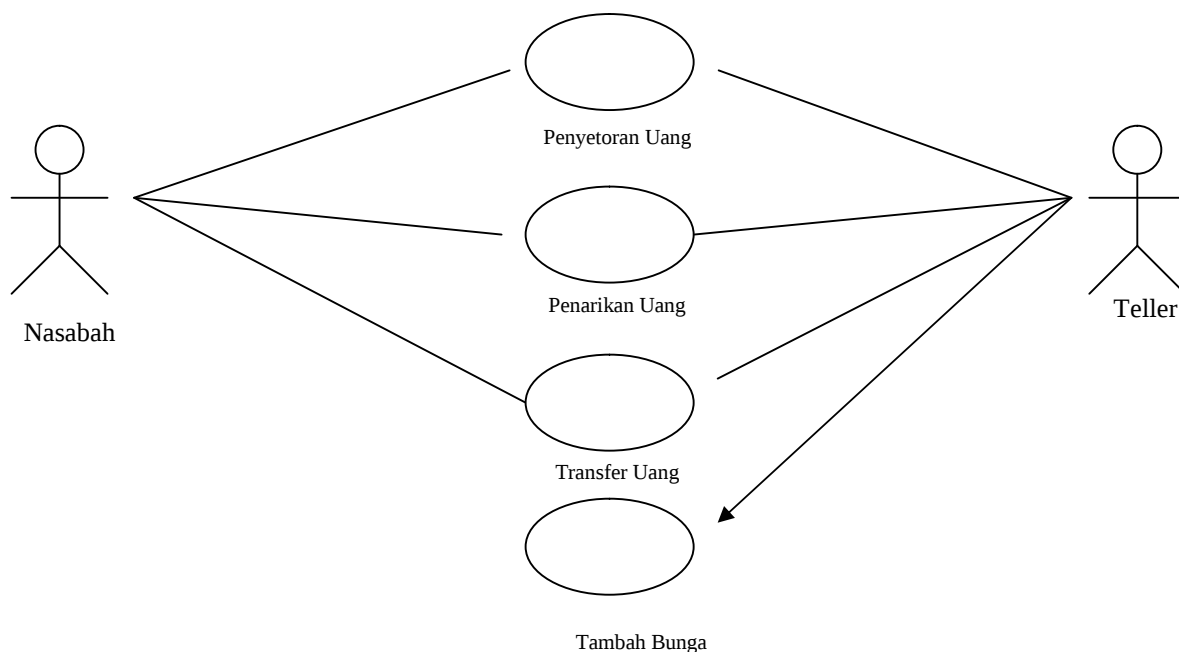
- a. Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam sistem.
- b. *Use Case*, aktivitas/ sarana yang disiapkan oleh bisnis/sistem.
- c. Hubungan (*Link*), aktor mana saja yang terlibat dalam *use case* ini.

	AKTOR : Sebuah entitas manusia / mesin yang berinteraksi dengan sistem
	USE CASE : Merepresentasikan fungsionalitas dari suatu sistem / pekerjaan tertentu pada sistem
	Asosiasi : Menjelaskan hubungan antar use case yang berupa pertukaran informasi.
	Dependency include : Menunjukkan suatu use case harus dipenuhi.
	Dependency extend : Menunjukkan suatu use case akan dilaksanakan bersifat optional.

Gambar II.4. Notasi Diagram Use Case

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:17)

Gambar di bawah ini merupakan salah satu contoh bentuk diagram use case (Prabowo Pudjo Widodo Dan Herlawati; 2011 : 16-17).



Gambar II.5. Diagram Use Case

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:17)

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*)

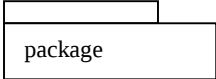
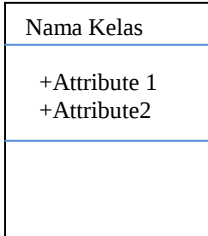
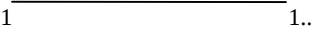
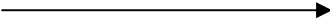
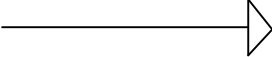
Kita biasanya membutuhkan *use case* yang membatalkan tujuan, misalnya pada *use case* pemesanan kamar, dibutuhkan pula *use case* pembatalan pesanan kamar.

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.

2. Diagram Kelas (*Class Diagram*)

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

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkus dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antar muka / interface 	Sama dengan konsep interface dalam pemrograman berorientasi objek
Asosiasi 	Relasi antar kelas dan makna umum , asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah /Directed Asosiasi 	Relasi antar kelas dengan makna kelas yang satu di gunakan oleh kelas yang lain asosiasi biasanya juga disertai dengan multiplicity
Generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum-khusus)
Ketergantungan/defedency 	Relasi antar kelas dengan makna ketergantungan antar kelas
Agregasi 	Relasi antar kelas dengan makna semua bagian (whole part)

Gambar II.6. Notasi Class Diagram

Sumber : (Yuni Sugiarti ; 2013 : 42)

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 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, Dan Herlawati; 2011 : 143-145).

Simbol	Keterangan
	Titik Awal
	Titik Akhir
	Activity
	Pilihan untuk pengambilan keputusan
	Fork; Untuk menunjukkan kegiatan yang dilakukan secara paralel
	Rake; menunjukkan adanya dekomposisi
	Tanda Waktu
	Tanda Penerimaan
	Aliran Akhir (Flow Final)

Gambar II.7. Activity Diagram

Sumber : (Yuni Sugiarti ; 2013 : 59)

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 konteks 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.

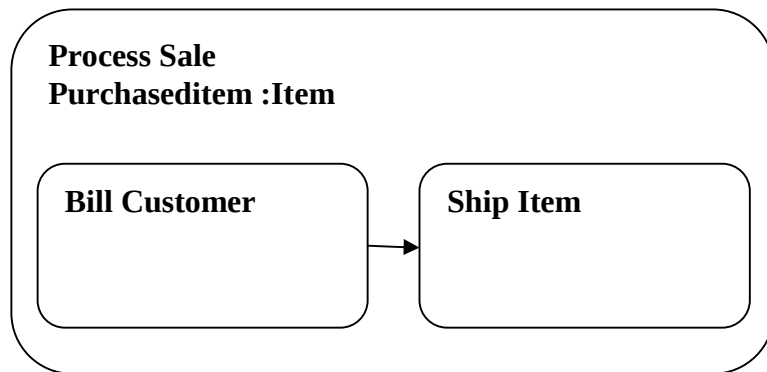
Aktivitas digambarkan dengan persegi panjang tumpul. Namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya.



Gambar II.8. Aktivitas sederhana tanpa rincian

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:145)

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



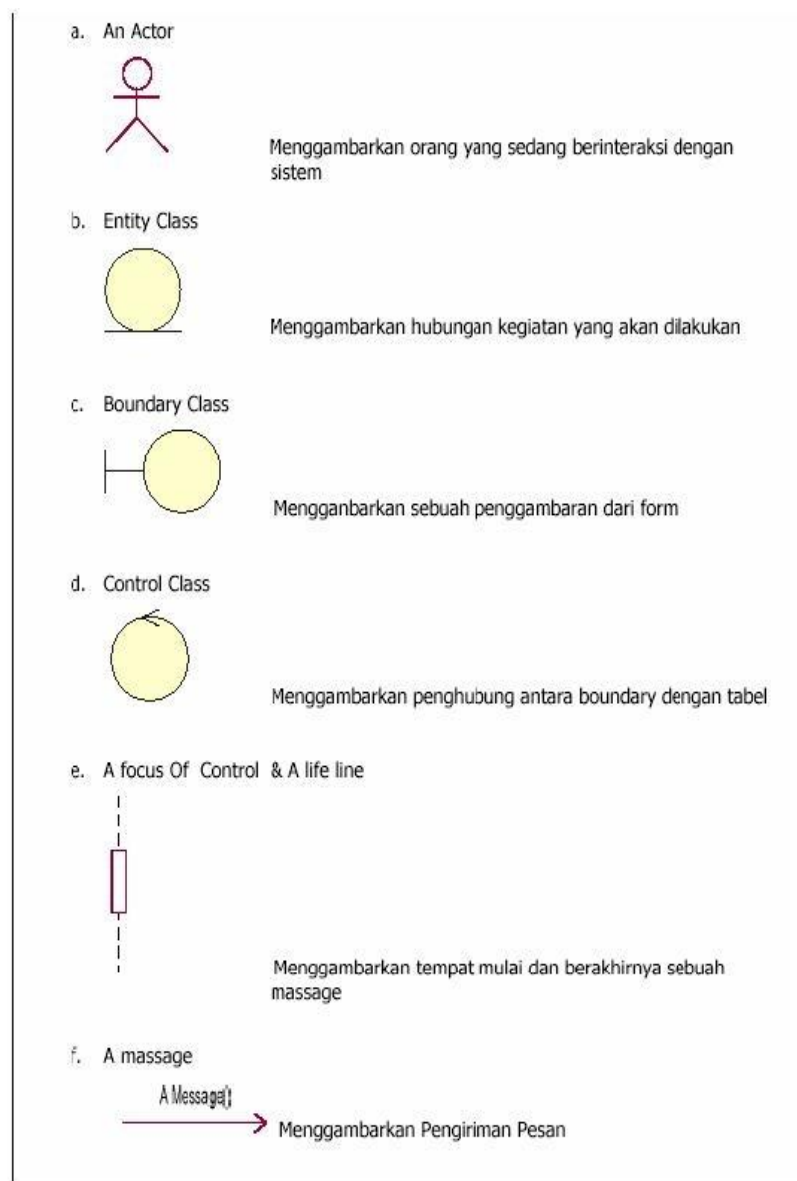
Gambar II.9. Aktivitas dengan detail rincian

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:145)

4. *Sequence Diagram*

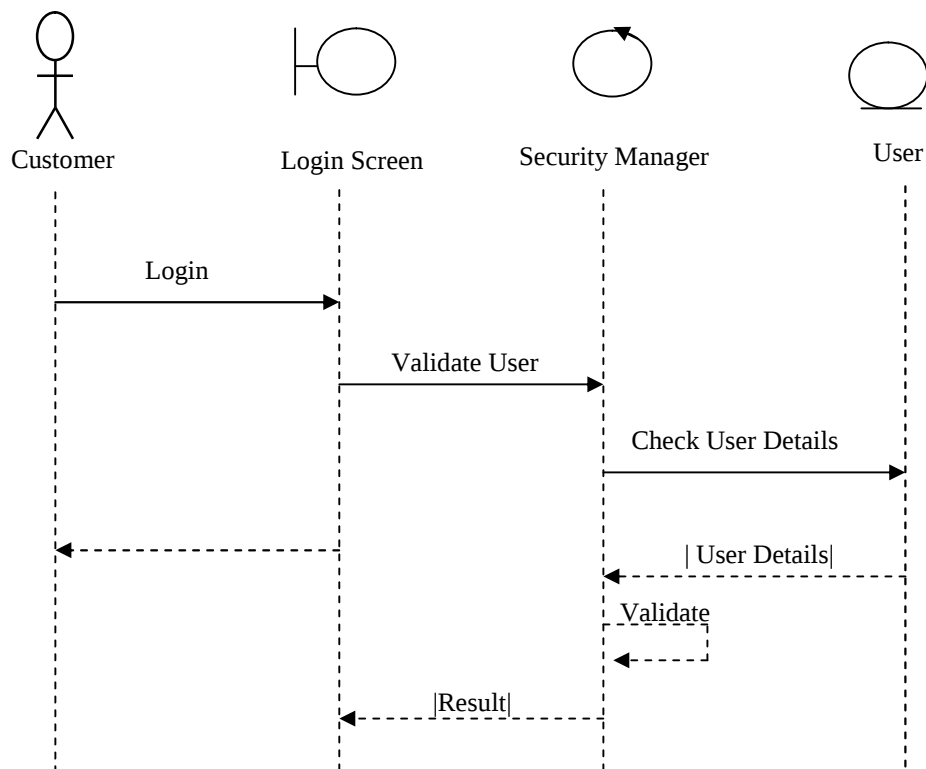
Menurut Douglas (2004 : 174) menyebutkan ada tiga diagram primer UML dalam memodelkan scenario interaksi, yaitu diagram urutan (*sequence diagram*), diagram waktu (*timing diagram*) dan diagram komunikasi (*communication diagram*).

Menurut Pilone (2005 : 174) menyatakan bahwa diagram yang paling banyak dipakai adalah diagram urutan. Gambar II.9. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nantinya (Prabowo Pudjo Widodo Dan Herlawati; 2011 : 174 – 175).



Gambar II.10. Notasi Diagram Urutan

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:175)



Gambar II.11. Diagram Urutan

Sumber : Prabowo Pudjo Widodo Dan Herlawati (2011:175)

II.7. Bahasa Pemrograman Microsoft Visual Studio 2010

Microsoft Visual Studio 2010 merupakan kelanjutan dari *Microsoft Visual Studio* sebelumnya, yaitu *Visual Studio. Net* 2003 yang diproduksi oleh Microsoft. Pada bulan Februari 2002 *Microsoft* memproduksi teknologi. *Net Framework* versi 1.0, teknologi. *Net* ini didasarkan atas susunan berupa *Net Framework*, sehingga setiap produk baru yang terkait dengan teknologi. *Net* akan selalu berkembang mengikuti perkembangan *Net Frameworknya*.

Pada perkembangannya nantinya mungkin untuk membuat program dengan teknologi. *Net* memungkinkan para pengembang perangkat lunak akan dapat menggunakan lintas sistem operasi, yaitu dapat dikembangkan di sistem operasi

windows juga dapat dijalankan pada sistem operasi lain, misalkan pada sistem operasi *Linux*, seperti yang telah dilakukan pada pemograman *Java* oleh *Sun Microsystem*.

Pada saat ini perusahaan-perusahaan sudah banyak mengupdate aplikasi lama yang dibuat *Microsoft Visual Basic 6.0* ke teknologi *.Net* karena kelebihan-kelebihan yang ditawarkan, terutama memungkinkan pengembang perangkat lunak secara cepat mampu membuat program *robust*, serta berbasiskan integrasi ke internet yang dikenal dengan *XML Web Service* (Ketut Darmayuda ; 2009 : 1)

II.8. *MYSQL*

Mysql adalah salah satu software sistem manajemen *database* (DBMS) *structured Query Language (SQL)* yang bersifat *open source*. *SQL* adalah bahasa standart untuk mengakses *database* dan didefenisikan dengan standart *ANSI/ISO SQL*. *MYSQL* dikembangkan, disebarluaskan, dan didukung oleh *MYSQL AB*. *MYSQL AB* adalah perusahaan komersial yang didirikan oleh pengembang *MYSQL*. *MYSQL* merupakan aplikasi *Relational Database Management System* (RDBMS) yang digunakan untuk aplikasi *client server* atau sistem *embedded*.

Mysql mempunyai beberapa sifat yang menjadikannya sebagai salah satu software database yang banyak digunakan oleh pemakai diseluruh dunia. Sifat-sifat yang dimiliki oleh *MYSQL* antara lain :

- a. *Mysql* merupakan DBMS (*Database Management System*)
- b. *Database* adalah kumpulan data yang terstruktur. Data dapat berupa daftar belanja, kumpulan gambar, atau yang lebih luas yaitu informasi jaringan perusahaan. Agar dapat menambah, mengakses, dan memproses data

tersimpan pada sebuah komputer database, kita membutuhkan sistem manajemen *database* (DBMS) seperti *MYSQL Server*. Sejak komputer sangat baik menangani sejumlah besar data, *sistem manajemen database* (DBMS) memainkan peran utama dalam perhitungan baik sebagai peralatan yang berdiri sendiri maupun bagian aplikasi.

- c. *Mysql* merupakan RDBMS (*Relational Database Management System*).
- d. *Database relational* menyimpan data pada table-tabel yang terpisah, bukan menyimpan data dalam ruang penyimpanan yang besar. Hal ini menambah kecepatan *fleksibilitas*.
- e. *Mysql* merupakan *software open source*.
- f. *Open source* berarti setiap orang dapat menggunakan dan mengubah software yang bersangkutan. Setiap orang dapat mendownload *software MYSQL* dari internet dan menggunakan tanpa membayar. Bahkan jika mengkehendakinya anda bisa mempelajari kode sumber dan mengubah sesuai yang anda butuhkan (Wahana Komputer; 2010 : 26-27).

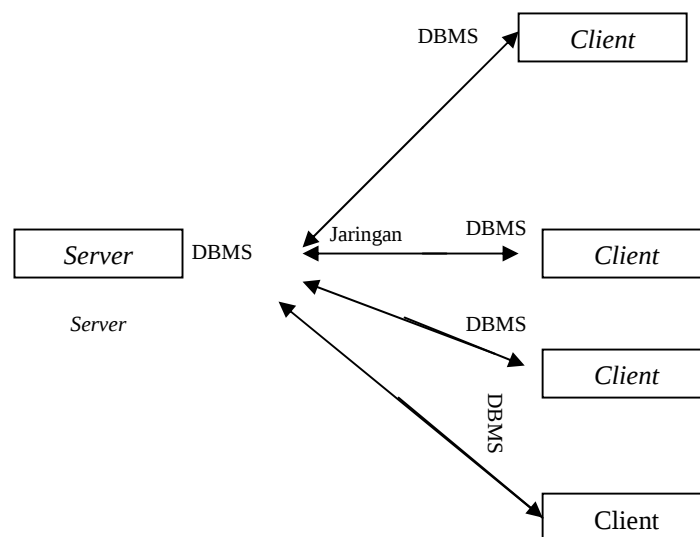
II.8.1. Client Server

Client server adalah satu model komunikasi 2 komputer atau lebih yang berfungsi melakukan pembagian tugas. *Client* bertugas untuk melakukan input, update, dan menampilkan data sebuah *database*. Sementara *server* bertugas untuk menyediakan pelayanan untuk melakukan manajemen yaitu : menyimpan dan mengolah *database* (Wahana Komputer; 2010 : 5).

II.8.2. Arsitekur *Client Server*

1. Arsitektur StandAlone (1-Tier)

Model pertama aplikasi pemrograman *database client server* adalah standalone atau 1 tier (1 -tingkat) adalah sebuah komputer yang mengakses sebuah *database* dari komponen sendiri. Dengan kata lain, aplikasi antarmuka *user* dan aplikasi DBMS terdapat pada komputer yang sama.



Gambar II.12. Arsitektur StandAlone (1-Tier)

Sumber : Wahana Komputer (2010 : 6)

Adapun karakteristik arsitektur 1 tier sebagai berikut :

- Beban jaringan menjadi tinggi karena yang diminta adalah file database secara keseluruhan pada komputer *server client* melalui jaringan.
- Setiap komputer pada jaringan harus mempunyai DBMS tersendiri untuk menyimpan hasil salinan dari *server* sehingga mengurangi sumber daya yang dimiliki oleh komputer *client* terutama *memory*.

- c. Komputer *client* harus mempunyai kemampuan proses yang tinggi untuk mendapatkan waktu respon yang baik saat komputer *server* mengirimkan *file* yang diminta.
- d. Arsitektur 1-tier cocok untuk bisnis kecil yang hanya membutuhkan data sebuah komputer untuk memproses dan menyimpan data sekaligus, tetapi kurang tepat diterapkan pada model jaringan (Wahana Komputer; 2010 : 7).

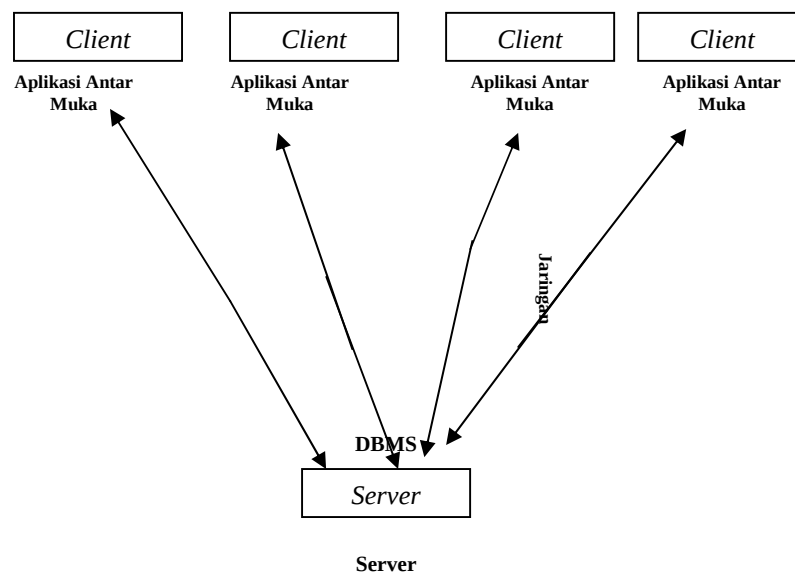
2. Arsitektur 2-Tier

Model kedua sebuah pemrograman *database* adalah model 2-tier. Arsitektur pada model demikian membagi tugas antara komputer *client* server. Komputer *client* bertugas menyediakan antar muka untuk *user*, permintaan (*request data*) ke DBMS Server, serta pemrosesan data (mencakup logika penyajian data, logika pemrosesan data, dan logika atau bisnis) komputer *client* hanya mengirimkan sebuah *statement* untuk menambah (*insert*) data, mengubah (*update*), menghapus (*delete*), dan yang terakhir meminta (*select*) data untuk ditampilkan melalui antarmuka yang dibuat oleh *programmer*. Sedangkan *server* bertanggung jawab terhadap penyimpanan, pengelolaan, melayani permintaan akses data, dan pemrosesan *client*.

Karakteristik arsitektur 2-tier adalah :

- a. 2-tier terjadi pada jaringan dan melakukan pemodelan programan *database* dalam 2 tingkat. Tingkat pertama adalah *client* dan tingkat kedua adalah *server*.

- b. Tingkat pertama komputer *client* sebagai penyedia aplikasi antarmuka untuk mengolah *database*, baik menampilkan data kedalam *user interface*, menambah, menghapus data, maupun logika bisnis (*bussines logic*)
- c. Tingkat kedua adalah *server* yang menyediakan aplikasi untuk mengelola *database* serta menyediakan pula *query stored procedure*, dan *triggers*, yang dapat dipanggil *client* untuk mengolah data.
- d. Komputer *client* hanya mengirimkan sebuah *statement sql* untuk meminta data ke *server*.
- e. *Server* hanya memberikan data yang diminta melalui *statement* bersangkutan.
- f. Komputer *server* dituntut untuk memiliki kemampuan pemrosesan yang tinggi karena harus melayani permintaan banyak komputer *client* yang mengakses satu atau lebih DBMS.
- g. Beban jaringan menjadi ringan karena data yang berjalan pada jaringan hanya data yang diminta oleh *client*.
- h. Otentifikasi pemakai, pemeriksaan integritas, dan pemeliharaan kamus data dilakukan pada sisi *server*.
- i. Sederhana dan mudah untuk diterapkan, khususnya pada bisnis kecil yang hanya terdapat pada satu gedung (Wahana Komputer; 2010 : 7-8).



Gambar II.13. Arsitekur 2-Tier

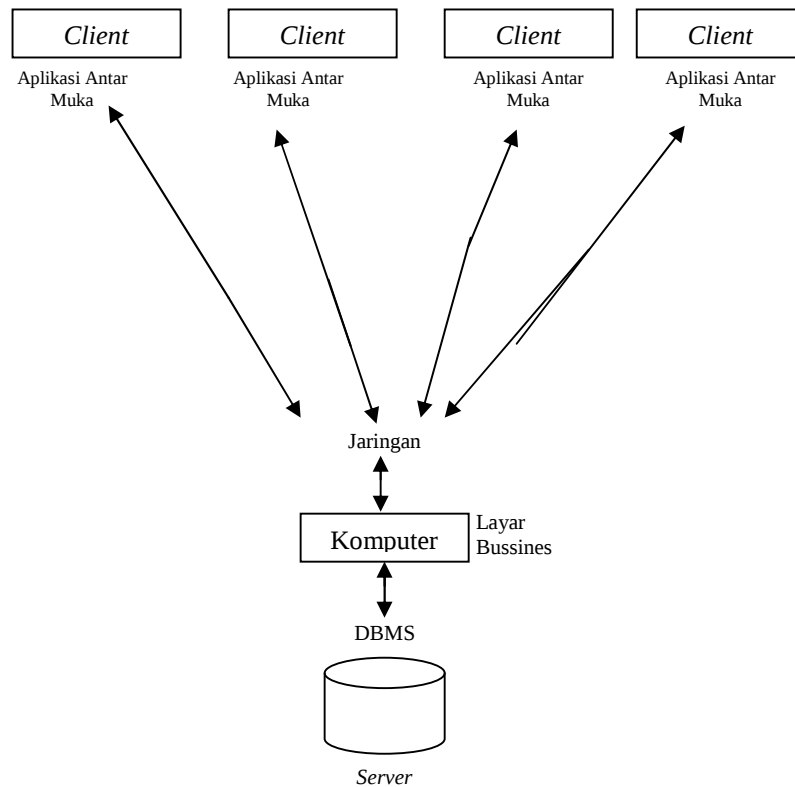
Sumber : Wahana Komputer (2010 : 8)

3. Arsitekur N-Tier

Arsitektur *n-tier* berarti membagi komponen menjadi *n* entitas yaitu 1 tier *client* dan *n-1 tier server*. Seperti pada model sebelumnya *client* bertugas menyediakan antarmuka aplikasi, sedangkan bertugas menyediakan data. Pada model *n-tier* (sebagai contoh adalah 3-tier), server dibagi 2 menjadi, yaitu satu server yang dipakai sebagai *bussines object (middle tier)* dan satu server yang hanya menyimpan *database (server tier)*.

Secara nyata model 3-tier adalah pada jaringan internet yang hanya memanfaatkan *database*. Internet lapisan pertama adalah komputer *client* yang menampilkan halaman *web*, tempat konten atau data halaman *web* berasal dari *database*. Lapisan kedua adalah *web* atau *HTTP server* yang menterjemahkan *script server side (PHP, JSP, ASP, dan lainnya)* dari komputer *client* untuk meminta data pada *database*. Kemudian lapisan

ketiga adalah komputer *database server* yang menyediakan *database* yang diminta oleh *web* atau HTTP server (Wahana Komputer; 2010: 6-9).



Gambar II.14. Arsitekur N-Tier

Sumber : Wahana Komputer (2010 : 9)