BAB II

TINJAUAN PUSTAKA

II.1. Sistem Informasi

II.1.1. Sistem

Suatu sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu (Tata Sutabri ; 2005 : 8).

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu:

1. Komponen Sistem (*Components*)

   Suatu sistem terdiri dari sejumlah komponen-komponen yang saling berinteraksi, artinya saling bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu bentuk subsistem.

2. Batas Sistem (*Boudary*)

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

3. Lingkungan Luar Sistem (*Environtment*)

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

4. Penghubung Sistem (*Interface*)

   Media yang menghubungkan sistem dengan subsistem lain disebut penghubung sistem atau interface. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lain.

5. Masukan Sistem (*Input*)

   Energi yang dimasukkan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan (*maintenance input*) dan sinyal (*signal input).*

6. Keluaran Sistem (*Output*)

   Hasil energi yang diolah dan diklasifikasikan menjadi pengeluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain.

7. Pengolahan Sistem (*Proses*)

   Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran.

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).

II.1.2. Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan (Tata Sutabri ; 2005 : 23). Kualitas informasi tergantung dari 3 hal yaitu :

a. Akurat (*Accurate*)

Informasi harus bebas dari kesalahan-kesalahan dan tidak menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai penerima informasi kemungkinan banyak terjadi gangguan *(noise)* yang dapat mengubah atau merusak informasi tersebut.

b. Tepat waktu (*Timeline*)

Informasi yang datang pada si penerima tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi karena informasi merupakan landasan dalam pengambilan keputusan.

c. Relevan (*Relevance*)

Informasi tersebut mempunyai manfaat untuk pemakainya. Relevansi informasi untuk orang satu dengan yang lain berbeda, misalnya informasi sebab kerusakan mesin produksi kepada akuntan perusahaan adalah kurang relevan dan akan lebih relevan bila ditujukan kepada ahli teknik perusahaan (Tata Sutabri ; 2005: 35).

II.1.3. Sistem Informasi

Sistem Informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar dengan laporan-laporan yang diperlukan (Tata Sutabri ; 2005: 42).

II.2. Sistem Informasi Akutansi

II.2.1. Akutansi

Akutansi adalah sebuah sistem informasi yang menghasilkan informasi keuangan kepada phak-pihak yang berkepentingan mengenai aktivitas ekonomi dan kondisi suatu perusahaan (Rudianto ; 2009 : 4).

II.2.2. Sistem Informasi Akutansi

Sistem informasi akutansi adalah sistem yang bertujuan untuk mengumpulkan dan memproses data serta melaporkan informasi yang berkaitan dengan transaksi keuangan (Anastasia Diana dan Lilis Setiawati ; 2011 : 4).

II.3. Aktiva Tetap

II.3.1. Pengertian Aktiva Tetap

Aktiva tetap adalah barang berwujud milik perusahaan yang sifatnya relatif permanen dan digunakan dalam kegiatan normal perusahaan, bukan untuk di perjualbelikan (Rudianto ; 2009 : 272).

II.3.2. Penilaian dan Pencatatan Aktiva Tetap

Untuk memperoleh aktiva tetap, perusahaan harus mengeluarkan sejumlah uang yang tidak hanya dipakai untuk membayar barang itu sendiri sesuai dengan nilai yang tercantum di dalam faktur, tetapi juga untuk beban pengiriman, pemasangan, perantara, balik nama dan sebagainya. Dan keseluruhan uang yang

dikeluarkan untuk memperoleh aktiva tersebut disebut dengan harga perolehan. Sedangkan dineraca aktiva tetap dicatat sebagai nilai bukunya.

Sedangkan nilai buku adalah nilai bersih dari suatu aktiva seperti yang tercantum didalam neraca, yaitu harga perolehan aktiva terebut setelah dikurangi dengan depresiasi dari aktiva tetap tersebut. Akumulasi depresiasi berarti kumpulan dari seluruh beban depresiasi selama beberapa periode akutansi (Rudianto ; 2009 : 272).

II.4. Penyusutan (Deprisiasi)

II.4.1. Pengertian Deprisiasi

Deprisiasi adalah pengalokasian harga perolehan aktiva tetap menjadi beban ke dalam periode akutansi yang menikmati masa manfaat dari aktiva tetap tersebut (Rudianto ; 2009 : 276).

Terdapat tiga faktor yang perlu dipertimbangkan dalam menentukan beban deprisiasi setiap periode, yaitu :

1. Harga Perolehan

   Adalah keseluruhan uang yang dikeluarkan untuk memperoleh suatu aktiva tetap sampai siap digunakan oleh perusahaan.

2. Nilai Sisa (residu)

   Adalah taksiran harga jual aktiva tetap tersebut pada akhir masa manfaat aktiva tetap tersebut. Setiap perusahaan akan memiliki taksiran yang berbeda satu dengan yang lainnya untuk suatu jenis aktiva tetap yang sama. Jumlah

taksiran nilai residu juga akan sangat dipengaruhi umur ekonomisnya, inflasi, nilai tukar mata uang, bidang usaha, dan sebagainya.

3. Taksiran Umur Kegunaan

   Adalah taksiran masa manfaat dari aktiva tetap tersebut. Masa manfaat adalah taksiran umur ekonomis dari aktiva tetap tersebut, bukan umur teknis. Taksiran masa manfaat dapat dinyatakan dalam satuan periode waktu, satuan hasil produksi atau satuan jam kerja (Rudianto ; 2009 : 276).

II.4.2. Metode Perhitungan Deprisiasi

Untuk mengalokasikan harga perolehan suatu aktiva tetap kedalam periode-periode yang menikmati aktiva tetap tersebut, bukan hanya menggunakan suatu metode saja. Terdapat beberapa metode yang dapat digunakan untuk menghitung beban deprisiasi periodik (Rudianto ; 2009 : 276), yaitu :

1. Metode Garis Lurus (*Straight Line Method*)

   Adalah suatu metode perhitungan deprisiasi aktiva tetap dimana setiap periode akutansi diberikan beban yang sama secara merata. Beban deprisiasi dihitung dengan cara mengurangi harga perolehan dengan nilai sisa dan dibagi dengan umur ekonomis dari aktiva tetap tersebut.

$$\text{Deprisiasi} = \frac{\text{Harga perolehan} - \text{Nilai Sisa}}{\text{Taksiran Umur Ekonomis}}$$

Sumber : (Rudianto ; 2005 : 277)

Metode perhitungan deprisiasi dengan metode garis lurus akan menghasilkan beban deprisiasi aktiva tetap yang sama dari tahun ke tahun. Metode ini juga dapat menghasilkan beban deprisi berupa suatu persentase dari harga perolehan aktiva tetap tersebut.

2. Metode Jam Jasa (*Service HourMethod*)

Adalah suatu metode perhitungan deprisiasi aktiva tetap, di mana beban deprisiasi pada suatu periode akutansi dihitung berdasarkan berapa jam periode akutansi tersebut mempergunakan aktiva tetap itu. Semakin lama aktiva tetap itu dipergunakan didalam suatu periode, akan semakin besar pula beban depresiasinya. Demikian pula sebaliknya, besarnya beban deprisiasi aktiva tetap dihitung dengan cara mengurangkan taksiran nilai residu dari harga perolehannya dan membagi hasilnya dengan taksiran jumlah jam pemakaian total dari aktiva tetap tersebut sepanjang umur ekonomisnya. Hasil pembagian tersebut adalah beban deprisiasi per jam. Jumlah tersebut dijadikan dasar untuk mengalihkan dengan jumlah jam aktual pemakaian aktiva tetap tersebut di dalam suatu periode, sehingga diketahui beban deprisiasi aktiva tetap pada suatu periode.

$$\text{Deprisiasi} = \frac{\text{Harga perolehan} - \text{Nilai Sisa}}{\text{Taksiran Jam Pemakaian Total}}$$

Sumber : (Rudianto ; 2005 : 278)

Beban deprisiasi aktiva tetap yang dihitung dengan metode jam jasa akan menghasilkan tarif deprisiasi per jam atau per satuan waktu tertentu. Dan berdasarkan tarif deprisiasi tersebut, beban deprisiasi suatu periode dihitung dengan mengalikan tarif tersebut dengan jumlah jam atau waktu yang digunakan didalam periode tersebut.

3. Metode Hasil Produksi (*Productive Output Method*)

   Adalah suatu periode perhitungan deprisiasi aktiva tetap, di mana beban deprisiasi pada suatu periode akutansi dihitung berdasarkan berapa banyak produk yang dihasilkan periode akutansi tersebut dengan menggunakan aktiva tetap itu. Semakin banyak produk yang dihasilkan di dalam suatu periode, akan semakin besar pula beban deprisiasinya. Demikian pula sebaliknya. Besarnya beban deprisiasi aktiva tetap dihitung dengan cara mengurangkan taksiran nilai residu dari harga perolehannya dan membagi hasilnya dengan taksiran jumlah produk yang akan dihasilkan dari aktiva tetap tersebut sepanjang umur ekonomisnya. Hasil dari pembagian tersebut akan diketahui beban deprisiasi per unit produk. Jumlah tersebut dijadikan dasar untuk mengalikan dengan jumlah unit produk yang dihasilkan secara aktual di dalam suatu periode, sehingga diketahui beban deprisiasi aktiva tetap pada suatu periode.

$$\text{Deprisiasi} = \frac{\text{Harga perolehan} - \text{Nilai Sisa}}{\text{Taksiran Jumlah Total Produk}}$$

Sumber : (Rudianto ; 2005 : 279)

Beban deprisiasi aktiva tetap yang dihitung dengan metode hasil produksi akan menghasilkan tarif deprisiasi per unit atau per satuan tertentu. Dan berdasarkan tarif depresiasi tersebut, beban deprisiasi suatu periode dihitung dengan mengalikan tarif tersebut dengan jumlah unit atau satuan lain yang digunakan di dalam periode tersebut.

4. Metode Beban Menurun *(Reducing Charge Method)*:

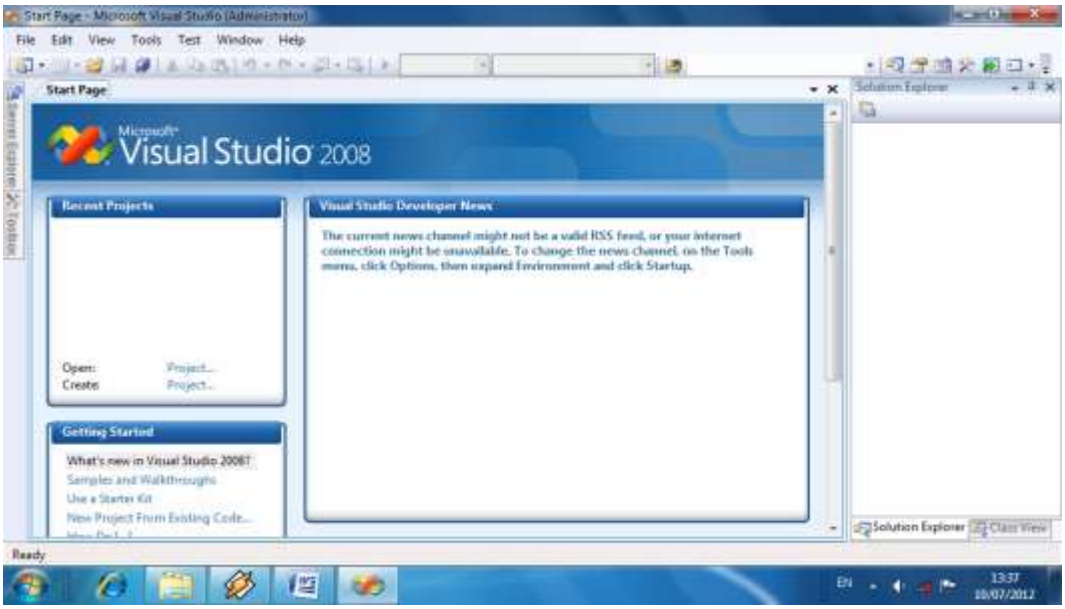
   a. Metode Jumlah angkah tahun *(Sun Of Years Digits Method)*
   b. Metode Saldo Menurun *(Declining Balance Method)*
   c. Metode Saldo Menurun Ganda *(Double Declining Balance Method)*
   d. Metode Tarif Menurun *(Declining Rate on Cost Method)*

II.5. Visual Basic 2008

Program *Visual Basic* 2008 merupakan salah satu paket bahasa pemrograman dari *Visual Studio* 2008. Banyak fasilitas yang akan kita dapatkan melalui rilis *Visual Basic* versi ini. *Visual Studio* 2008 merupakan suatu *software* untuk membuat sebuah aplikasi seperti aplikasi *windows*, jadi melalui software ini kita bias membuat sebuah aplikasi seperti aplikasi *database*, aplikasi *inventory* dan sebagainya. Kebanyakan orang lebih suka menyebut sebuah aplikasi sebagai sebuah program atau *software*, padahal ketiga istilah ini memiliki arti yang sama (Rahmat Priyanto ; 2009 : 1).

II.5.1. Mengenal Visual Studio 2008

Semenjak *Visual Studio.NET, Microsoft* telah banyak melakukan pengembangan dan perubahan pada tampilan software ini. Jadi apabila anda sudah terbiasa menggunakan rilis *Visual Basic* sebelumnya, anda harus mulai beradaptasi dengan tampilan baru *Visual Basic*. Pada dasarnya tampilan baru ini memudahkan kita dalam menggunakan *software Visual Basic* (disingkat VB). Penulis asumsikan bahwa anda telah menginstal *Visual Studio* 2008 (minimal versi standart) pada komputer yang anda pergunakan. Ketika pertama kali di jalankan, *Visual Studio* akan menampilkan sebuah lembar kerja. Dalam lembar kerja ini kita dapat melihat berbagai macam menu dan toolbar, dengan sebuah halaman *Start Page* (Halaman Pembuka) di dalamnya. Tampilan awal *Visual Studio 2008* dapat dilihat pada gambar II.1.



Gambar II.1 Tampilan Awal Visual Studio 2008

Sumber : (Rahmat Priyanto ; 2009 : 2)

Melalui lembar kerja ini kita dapat membuat aplikasi baru, ada dua buah istilah yang perlu kita ketahui dalam Visual Studio 2008 yaitu :

1. Project

   Merupakan sebutan bagi sebuah *software* yang sedang melalui tahap pembuatan menggunakan *visual studio*, belum menjadi sebuah aplikasi. Terdapat berbagi jenis *project* diantarnya *project* pembuatan aplikasi *windows, project* pembuatan aplikasi *console*, dan sebagainya.

2. Solution

   Adalah kumpulan beberapa buah project, sebuah solution dapat terdiri atas satu buah project atau beberapa buah project, bergantung pada kebutuhan. Sebuah project harus disimpandalam sebuah solution (Rahmat Priyanto ; 2009 : 2)

II.6. *MySQL*

MySQL adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya disebabkan *MySQL* menggunakan *SQL* sebagai bahasa dasar untuk mengakses *database*nya. Selain itu, ia bersifat *Open Source* (Anda tidak perlu membayar untuk menggunakannya) pada berbagai platform (kecuali untuk jenis enterprise, yang bersifat komersial).

MySQL termasuk jenis RDBMS (*Relational Database Management System*). Itulah sebabnya, istilah table, baris, dan kolom dipergunakan pada *MSQL*. Pada *MySQL*, sebuah *database* mengandung satu atau sejumlah table.

Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom (Abdul Kadir ; 2008 : 348).

II.7. Konsep UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia perkembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi perkembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan dengan yang lain (Munawar ; 2005 : 17).

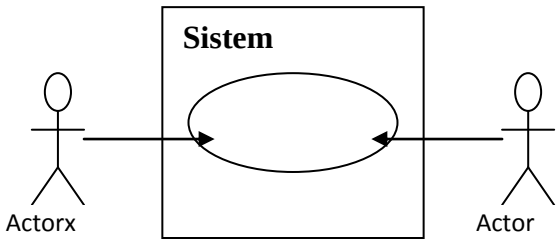
II.7.1. Diagram – Diagram Pada Metode UML

1. Use Case Diagram

   Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara deskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem yang disebut *scenario*. Setiap *scenario* mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras dan urutan waktu. Dengan demikian secara singkat bisa dikatakan *use case* adalah serangkaian *scenario* yang digabungkan bersama-sama oleh pengguna tujuan umum pengguna. Dalam

pembicaraan tentang *use case*, pengguna biasanya disebut dengan *actor*. *Actor* adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.

Notasi *use case* menunjukkan 3 aspek dari sistem yaitu *actor use case* dan *system / sub system boundary*. *Actor* mewakili peran orang, *system* yang lain atau alat ketika berkomunikasi dengan *use case*. Ilustrasi *actor, use case* dan *system* ditunjukkan pada gambar II.2.



Gambar II.2. Use Case Diagram

Sumber : (Munawar ; 2005 : 64)

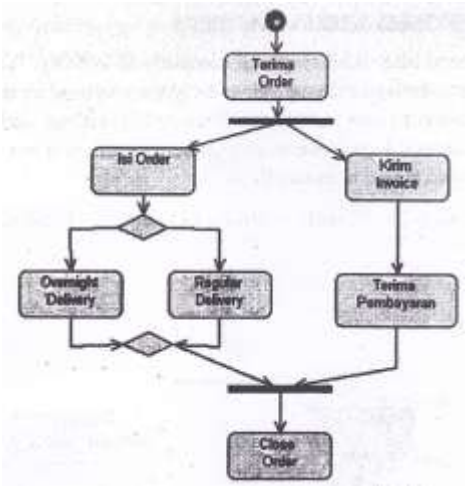
Untuk mengidentifikasi *actor*, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks targer sistem. *Actor* adalah *abstraction* dari orang dan sistem yang lain mengaktifkan fungsi dari target sistem. Orang atau sistem bila muncul dalam beberapa peran. Perlu dicatat bahwa *actor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*.

Use case adalah abstraksi dari interaksi antara sistem dan *actor*. Oleh kerena itu sangat penting untuk memilih abstraksi yang cocok. *Use case* dibuat berdasarkan keperluan *actor*. *Use case* harus merupakan ’apa’ yang

dikerjakan *software* aplikasi, bukan ’bagaimana’ *software* aplikasi mengerjakannya. Setiap use case harus diberi nama yang menyatakan apa hal yang dicapai dari hasil interaksinya dengan *actor*. Namun *use case* boleh terdiri dari beberapa kata dan tidak boleh ada dua *use case* yang memiliki nama yang sama (Munawar ; 2005 : 63-66).

2. Activity diagram

Activity diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaanya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan flowchart tidak bisa (Munawar ; 2005 : 87). Berikut gambar *activity diagram* sederhana.



Gambar II.3. Gambar Activity Diagram Sederhana

Sumber : (Munawar ; 2005 : 111)

3. Class Diagram

Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut *Atribut* dan metode atau operasi :

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Susunan kelas suatu sistem yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main, kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem, kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
3. Kelas yang diambil dari pendefinisian *use case*, kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*.
4. Kelas yang diambil dari pendefinisian data, kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Jenis-jenis kelas diatas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca *file* teks, dan lain sebagainya sesuai kebutuhan. Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan apa yang disebut dengan *cohension* dan *coupling*.

Cohension adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain sedangkan *coupling* adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan yang lain dalam sebuah kelas. Sebagai aturan secara umum maka sebuah metode yang dibuat harus memiliki kadar *cohesion* yang kuat dan kadar *coupling* yang lemah (Rosa A.S dan M. Shalahuddin ; 2011 : 122-123).

Tabel II.I. simbol-simbol yang ada pada diagram kelas

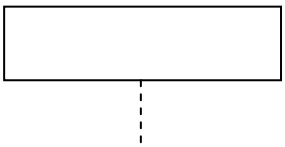
Simbol	Deskripsi
Kelas **nama_kelas** +atribut +operasi()	Kelas pada struktur sistem
Antarmuka / *interface* nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicy*
Asosiasi berarah / *directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicy*
Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
Keberuntungan /*dependency*	Relasi antar kelas dengan makna ketergantungan antar kelas
Agregasi /*aggregation*	Relasi antar kelas dengan makna

4. *Squence diagram* digunakan untuk menggambarkan perilaku pada sebuah sekenario. Diagram ini menunjukkan sebuah contoh objek dan pesan yang diletakkan diantara objek-objek ini didalam *use case*.

   Komponen utama *Squence diagram* terdiri dari atas objek yang dituliskan dengan kotak segiempat bernama. *Messege* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical* (Munawar ; 2005 : 109).

   a. Objek / *participant*

      Objek diletakkan di dekat bagian atas diagram dengan urutan dari kiri ke kanan. Mereka diatur dalam urutan guna menyederhanakan diagram. Setiap *participant* dihubungkan garis titik-titik yang disebut *lifeline.* Sepanjang *lifeline* ada kotak yang disebut *activation*. *Activation* mewakili sebuah eksekusi operasi dari *participant*. Panjang kotak ini berbanding lurus dengan durasi *activation*. *Activation* mewakili sebuah eksekusi operasi dari participant. Panjang kotak ini berbanding lurus dengan durasi *activation.* Bentuk *participant* dapat dilihat pada gambar II.4.



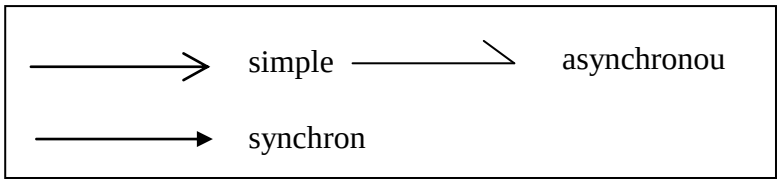
Gambar II.4. Bentuk Participant

Sumber : (Munawar ; 2005 : 88)

b. *Messege*

Sebuah *messege* bergerak dari suatu *participant* ke *participant* yang lain dan dari *lifeline* ke *lifeline* yang lain. Sebuah *participant* bisa mengirim sebuah *message* kepada dirinya sendiri.

Sebuah *message* bisa jadi *simple*, *synchronous* atau *asynchoronous*. *Message* yang *simple* adalah sebuah perpindahan (transfer), contoh dari satu *participant* ke *participant* yang lainnya. Jika suatu *participant* mengirimkan sebuah *message* tersebut akan ditunggu sebelum di proses dengan urusannya. Namun jika *message asynchoronous* yang dikirimkan, maka jawabannya atas *message* tersebut tidak perlu ditunggu. Simbol *message* pada *squnence diagram* dapat dilihat pada gambar II.5.



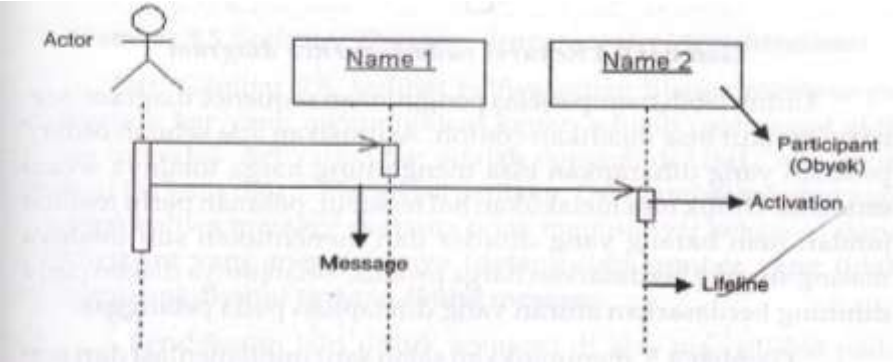
Gambar II.5. Bentuk Messege

Sumber : (Munawar ; 2005 : 88)

c. *Time*

Time adalah diagram yang mewakili waktu pada arah vertikal. Waktu dimulai dari atas ke bawah. *Message* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibanding *message* yang lebih dekat kebawah. Terdapat dua dimensi pada *squence diagram* yaitu dimensi dari kiri ke kanan menunjukkan tata letak participant dan dimensi dari

atas ke bawah menunjukkan lintasan waktu. Simbol-simbol yang ada pada *squence diagram* ditunjukkan pada gambar II.6.



Gambar II.6. Bentuk Time

Sumber : (Munawar ; 2005 : 89)

II.8. Konsep Sistem Database

Database adalah kumpulan data terhubung *(interrelated data)* yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data *(controlled redundancy)* 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 rupa sehingga penambahan, pengambilan, dan modifikasi dapat dilakukan dengan mudah dan terkontrol. Sementara itu, sistem database adalah sekumpulan database yang dapat dipakai secara bersama-sama, personal-personal yang merancang dan mengelola database, teknik-teknik untuk merancang dan mengelola database, serta computer untuk mendukungnya (Tata Sutabri ; 2005 : 161).

II.8.1. Normalisasi

Normalisasi adalah suatu teknik yang menstruktukan data dalam cara tertentu untuk membantu mengurangi atau mencegah timbulnya masalah yang berhubungan dengan pengolahan data dalam database (Tata Sutabri ; 2005 : 181).

Tahap normalisasi terdiri dari beberapa bentuk :

1. 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.

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

   Bentuk normal kesatu mempunya ciri : setiap data dibentuk dalam *file 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 bernilai ganda (*multivalue*). Tiap *field* hanya satu pengertian, bukan merupakan kumpulan kata yang mempunyai arti mendua, 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.

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

   Bentuk normal kedua memiliki syarat : bentuk data telah memenuhi kriteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung fungsi pada kunci utama/*primary key* sehingga untuk membentuk normal kedua haruslah sudah ditentukan kunci *field*. Kunci *field* haruslah unik dan dapat mewakili

atribut lain yang menjadi anggotanya. Tahap normalisasi terdiri dari beberapa bentuk.

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

   Untuk menjadi bentuk normal ketiga, relasi haruslah dalam bentuk normal kedua dan semua atriut dalam primer tidak punya hubungan yang transif. Dengan kata lain, setiap atribut bukan kunci haruslah bergantung hanya pada *primary key* dan pada *primary key* secara menyeluruh. Contoh pada bentuk kedua di atas termasuk juga bentuk normal ketiga seluruh atribut yang ada disitu bergantung penuh pada kunci primernya.

5. Bentuk Normal *Boyce Codd* (BCNF/ *Boyce Codd Normal Form*)

   Boyce Codd Normal Form mempunyai paksaan yang lebih kuat dari pada bentuk 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).

II.9. Kamus Data

Kamus data (KD) data dictionary (DD) atau disebut juga dengan istilah system data dictionary adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan KD, analisis sistem dapat mendefenisikan data yang mengalir di sistem dengan lengkap. KD dibuat pada tahap analisis 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, yaitu tentang data yang masuk ke sistem dan tentang informasi yang di butuhkan oleh pemakai sistem. Pada tahap perancangan sistem, KD digunakan untuk merancang input, merancang laporan-laporan dan *database.* KD di buat berdasarkan arus data yang ada di DAD. Arus data di DAD sifatnya adalah global, hanya ditunjukkan nama arus datanya saja (Yogiyanto HM ; 2005 : 725).

II.9.1. Isi Kamus Data

Apa yang perlu dicatat di kamus data? KD harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya. Untuk maksud keperluan ini maka KD harus membuat hal-hal berikut ini (Yogiyanto HM ; 2005 : 726) :

1. Nama Arus Data

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

2. Alias

   Alias atau nama lain dari data dapat dituliskan bila nama lain ini ada. Alias perlu ditulis karena data yang sama 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 ini mempunyai struktur data yang sama, tetapi mempunyai struktur yang berbeda.

3. Bentuk Data

Telah diketahui bahwa arus data dapat mengalir :

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

Dokumen hasil cetakan komputer :

a. Laporan tercetak
b. Tampilan dilayar monitor
c. Variabel
d. Parameter
e. *Field*.

Bentuk dari data ini perlu dicatat di KD, karena dapat digunakan untuk mengelompokkan KD kedalam kegunaannya sewaktu perancangan sistem. KD mencatat data yang mengalir dalam bentuk dokumen dasar atau formulir

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

4. Arus Data

   Arus data dapat menunjukkan dari mana data mengalir dan kemana data akan menuju. Keterangan arus data ini perlu dicatat di KD supaya memudahkan mencari arus daata ini di DAD.

5. Penjelasan

   Untuk lebih memperjelas lagi tentang makna dari arus data yang dicatat di KD, maka bagian penjelasan dapat diisi dengan keterangan-keterangan tentang arus data tersebut. Sebagai misalnya nama dari arus data adalah TEMBUSAN PERMINTAAN PERSEDIAAN, maka dapat lebih dijelaskan sebagai tembusan dari faktur penjualan untuk meminta barang dari gudang.

6. Periode

   Periode ini menunjukkan kapan terjadinya arus data ini. Periode perlu dicatat di KD karena dapat digunakan untuk mengidentifikasi kapan *input* data harus dimasukkan ke sistem, kapan proses dari program harus dilakukan dan laporan-laporan harus dihasilkan.

7. Volume

   Volume yang perlu dicatat di KD adalah tentang volume rata-rata dan volume puncak dari arus data. Volume rata-rata menunjukkan banyaknya rata-rata arus data yang mengalir dalam suatu periode tertentu dan volume puncak menunjukkan volume yang terbanyak. Volume ini digunakan untuk mengidentifikasikan besarnya simpanan luar yang akan digunakan, kapasitas dan jumlah dari alat *input*, alat proses dan alat *output*.

8. Stuktur data

   Struktur data menunjukkan arus data yang dicatat di KD terdiri dari item-item data apa saja (Yogiyanto HM ; 2005 : 726-728).

II.10. *Entity Relationship Diagram* (ERD)

ERD merupakan notasi grafis dalam pemodelan data konseptual yang mendeskripsikan hubungan antar penyimpanan. ERD digunakan untuk memodelkan struktur data dan hubungan antar data, karena hal ini relatif kompleks (Kusrini dan Andri Koniyo ; 2007 : 99). ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar dua data. Pada dasarnya ada 3 macam simbol yang digunakan, yaitu :

1. Entity

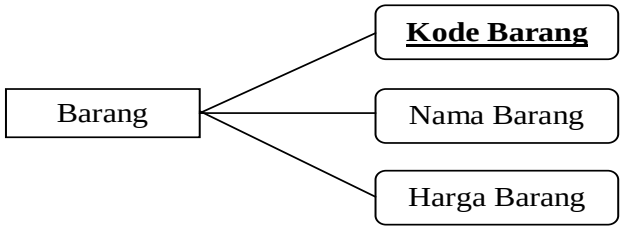
   Entity adalah suatu objek yang dapat diidentifikasikan dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat. Entitas dapat digambarkan dalam bentuk persegi empat.

Gambar II.7. Entitas

Sumber : (Kusrini dan Andi Kuniyo ; 2007 : 99)

2. Atribut

Entitas mempunyai elemen yang disebut atribut dan berfungsi mendeskripsikan karakter entitas, misalnya atribut nama barang dari entitas barang. Setiap ERD bisa berisi lebih dari satu atribut. Entitas digambarkan dalam bentuk elips.

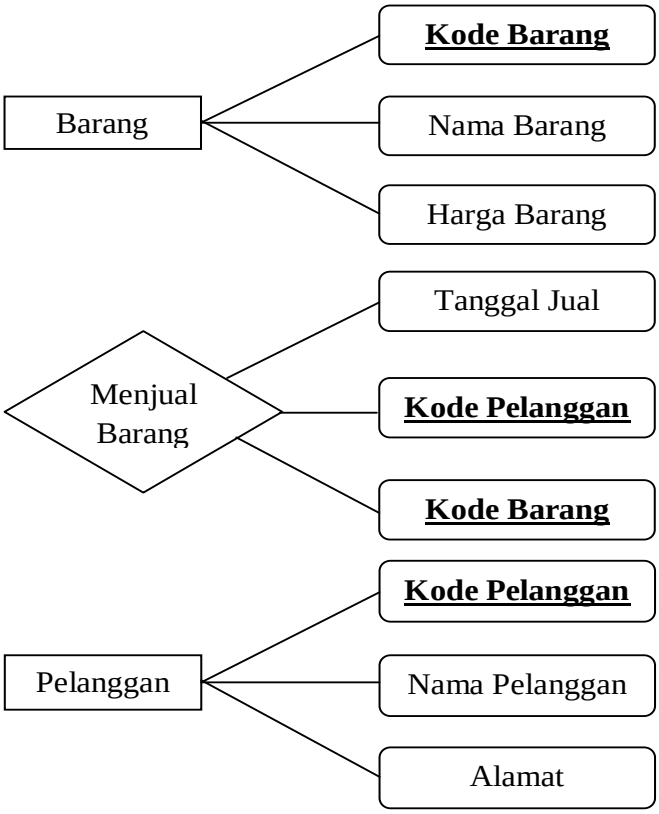


Gambar II.8. Atribut

Sumber : (Kusrini dan Andi Kuniyo ; 2007 : 100)

3. Hubungan / *relationship*

Sebagimana halnya entitas, hubungan pun harus dibedakan antara hubungan atau bentuk hubungan antar entitas dengan isi dari hubungan itu sendiri. Misalnya dalam kasus hubungan antar entitas barang dan entitas pelanggan adalah menjual barang, sedangkan isi hubungannya dapat berupa tanggal jual atau yang lainnya. *Relationship* digambarkan dalam bentuk intan (*diamons*).



Gambar II.9. Relationship

Sumber : (Kusrini dan Andi Kuniyo ; 2007 : 100)

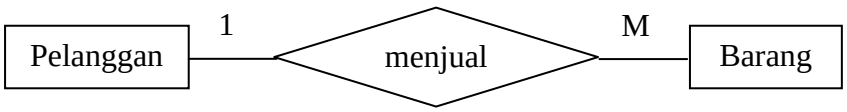
Jenis-jenis hubungan:

a. Satu ke satu, misalnya suatu perusahaan mempunyai aturan satu supir hanya boleh menangani satu kendaraan karena alasan tertentu.

Gambar II.10. Relational 1 to 1

Sumber : (Kusrini dan Andi Kuniyo ; 2007 : 100)

b. Satu ke banyak atau banyak ke satu, misalya suatu perusahaan selalu berasumsi bahwa satu pelanggan dapat membeli banyak barang



Gambar II.11. Relational 1 to Many

Sumber : (Kusrini dan Andi Kuniyo ; 2007 : 101)