

BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Menurut Kusriani & Andi Koniyo (2007 : 5), kata *sistem* mempunyai beberapa pengertian, tergantung dari sudut pandang mana kata tersebut didefinisikan. Secara garis besar ada dua kelompok pendekatan, yaitu :

1. Pendekatan sistem yang lebih menekankan pada elemen-elemen atau kelompoknya, yang dalam hal ini sistem itu didefinisikan sebagai suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu aturan tertentu.
2. Pendekatan sistem sebagai jaringan kerja dari prosedur, yang lebih menekankan urutan operasi di dalam sistem. Prosedur (*procedure*) didefinisikan oleh F. Nieshl sebagai urutan operasi kerja (tuliskan-menulis), yang biasanya melibatkan beberapa orang di dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi bisnis yang terjadi.

Pendekatan sistem yang lebih menekankan pada elemen-elemen atau komponennya mendefinisikan sistem sebagai sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Dengan demikian di dalam suatu sistem, komponen-komponen ini tidak dapat berdiri

sendiri-sendiri, tetapi sebaliknya, saling berhubungan hingga membentuk satu kesatuan sehingga tujuan sistem itu dapat tercapai.

II.1.1. Karakteristik Sistem

Menurut Kusriani & Andi Koniyo (2007 : 6), sistem mempunyai beberapa karakteristik atau sifat-sifat tertentu, antara lain :

1. Komponen Sistem (*Component*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang saling bekerja sama membentuk suatu komponen sistem atau bagian-bagian dari sistem.

2. Batasan Sistem (*Boundary*)

Merupakan daerah yang membatasi suatu sistem dengan sistem yang lain atau dengan lingkungan kerjanya.

3. Subsistem

Bagian-bagian dari sistem yang beraktivitas dan berinteraksi satu sama lain untuk mencapai tujuan dengan sasarannya masing-masing.

4. Lingkungan Luar Sistem (*Environment*)

Suatu sistem yang ada di luar dari batas sistem yang dipengaruhi oleh operasi sistem.

5. Penghubung Sistem (*Interface*)

Media penghubung antara suatu subsistem dengan subsistem lain. Adanya penghubung ini memungkinkan berbagai sumber daya mengalir dari suatu subsistem ke subsistem lainnya.

6. Masukan Sistem (*Input*)

Energi yang masuk ke dalam sistem, berupa perawatan dan sinyal.

Masukan perawatan adalah energi yang dimasukan supaya sistem tersebut dapat berinteraksi.

7. Keluaran Sistem (*Output*)

Hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

8. Pengolahan Sistem (*Process*)

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

9. Sasaran Sistem (*Object*)

Tujuan yang ingin dicapai oleh sistem, akan dikatakan berhasil apabila mengenai sasaran atau tujuan.

II.1.2. Klasifikasi Sistem

Menurut Kusri & Andi Koniyo (2007 : 7), suatu sistem dapat diklasifikasikan menjadi seperti berikut :

1. Sistem abstrak dan fisik

Sistem abstrak adalah suatu sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik, sedangkan sistem fisik adalah sistem yang ada secara fisik

2. Sistem alamiah dan sistem buatan manusia

Sistem alamiah adalah sistem yang terjadi melalui proses alam sedangkan sistem buatan manusia adalah sistem yang dirancang oleh manusia.

3. Sistem tertentu dan sistem tak tentu

Sistem tertentu adalah suatu sistem yang operasinya dapat diprediksi secara tepat sedangkan sistem tak tertentu adalah sistem dengan perilaku ke depan yang tidak dapat diprediksi.

4. Sistem tertutup dan sistem terbuka

Sistem tertutup adalah sistem yang tidak terpengaruh oleh lingkungan luar atau otomatis, sedangkan sistem terbuka adalah sistem yang berhubungan dan berhubungan oleh lingkungan luar.

II.1.3. Pengertian Informasi

Menurut Kusrini & Andi Koniyo (2007 : 7), informasi adalah data yang sudah diolah menjadi sebuah bentuk yang berarti bagi pengguna, yang bermanfaat dalam pengambilan keputusan saat ini atau mendukung sumber informasi. Data belum memiliki nilai, sedangkan informasi sudah memiliki nilai. Informasi dikatakan bernilai bila manfaatnya lebih besar dibandingkan biaya untuk mendapatkannya.

II.1.4. Kualitas Informasi

Menurut Kusrini & Andi Koniyo (2007 : 8), informasi yang berkualitas memiliki 3 karakteristi, yaitu :

1. Akurat (*Accurate*)

Informasi harus bebas dari kesalahan, tidak bias ataupun menyesatkan. Akurat juga berarti bahwa informasi itu harus dapat dengan jelas mencerminkan maksudnya.

2. Tepat pada waktunya (*timeliness*)

Informasi yang datang pada penerima tidak boleh terlambat, didalam pengambilan keputusan, informasi yang sudah usang tidak lagi bernilai. Bila informasi datang terlambat sehingga pengambilan keputusan terlambat dilakukan, hal itu dapat berakibat fatal bagi perusahaan.

3. Relevan (*relevance*)

Informasi yang disampingkan harus mempunyai keterkaitan dengan masalah yang akan dibahas dengan informasi tersebut. Informasi harus bermanfaat bagi pemakai, disamping karakteristik, nilai informasi juga ikut menentukan kualitasnya. Nilai informasi (*value of information*) ditentukan oleh dua hal, yaitu manfaat dan biaya untuk mendapatkannya. Suatu informasi dikatakan bernilai bila manfaatnya lebih besar dibandingkan biaya untuk mendapatkannya.

II.1.5. Sistem Informasi Akuntansi

Menurut Anastasia Diana & Lilis Setiawati (2011:3) Sistem merupakan serangkaian bagian yang saling tergantung dan bekerja sama untuk mencapai tujuan tertentu. Suatu sistem pasti tersusun dari sub-sub sistem yang lebih kecil yang saling tergantung dan bekerja sama untuk mencapai tujuan.

Menurut Kusrini & Andi Koniyo (2007 : 10) Sistem Informasi akuntansi merupakan sebuah sistem informasi yang mengubah data transaksi bisnis menjadi informasi keuangan yang berguna bagi pemakainya.

Tujuan dari sistem informasi akuntansi adalah :

1. Mendukung operasi sehari-hari.
2. Mendukung pengambilan keputusan manajemen.

3. Memenuhi kewajiban yang berhubungan dengan pertanggung jawaban.

Komponen-komponen yang terdapat dalam sistem informasi akuntansi adalah sebagai berikut:

1. Orang-orang yang mengoperasikan sistem tersebut.
2. Prosedur-prosedur, baik manual maupun yang terotomatisasi, yang dilibatkan dalam pengumpulan, pemrosesan dan penyimpanan data aktivitas-aktivitas organisasi.
3. Data tentang proses-proses bisnis.
4. *Software* yang dipakai untuk memproses data organisasi.
5. Infrastruktur teknologi informasi.

Didalam organisasi, sistem informasi akuntansi berfungsi untuk :

1. Mengumpulkan dan menyimpan aktivitas yang dilaksanakan disuatu organisasi, sumber daya yang dipengaruhi oleh aktivitas-aktivitas tersebut dan para pelaku aktivitas tersebut.
2. Mengubah data menjadi informasi yang berguna bagi manajemen.
3. Menyediakan pengendalian yang memadai.

Sistem informasi akuntansi merupakan pendukung aktivitas organisasi, yang termasuk pendukung aktivitas organisasi adalah:

1. Infrastruktur perusahaan : akuntansi, hukum, administrasi umum.
2. Sumber daya manusia : perekrutan, pengontrolan, pelatihan dan kompensasi kepada pegawai.
3. Teknologi : peningkatan produk dan jasa (penelitian).
4. Pembelian.

Sementara itu aktivitas utamanya adalah:

1. *Inbound Logistics* : Penerimaan, penyimpanan dan distribusi bahan-bahan masukan.
2. Operasi : aktivitas untuk mengubah masukan menjadi barang atau jasa.
3. *Outbound Logistics* : distribusi produk ke pelanggan.
4. Pemasaran dan Penjualan.
5. Pelayanan : Dukungan purna jual dan *maintenance*.

Dari penjelasan tersebut dapat diambil simpulan bahwa akuntansi adalah proses pengidentifikasian, pengukuran dan melaporkan informasi ekonomi untuk memungkinkan adanya pembuatan pertimbangan dan keputusan bagi yang menggunakan informasi.

Kutipan diatas menjelaskan bahwa sistem informasi akuntansi adalah kumpulan dari sumber-sumber seperti orang dan peralatan yang dirancang untuk mentransformasikan data keuangan dan data lainnya menjadi informasi, dan informasi ini akan dikomunikasikan kepada para pembuat keputusan.

Manfaat Sistem Informasi Akuntansi tersebut adalah sebagai berikut :

1. Mengamankan harta/kekayaan perusahaan.
2. Menghasilkan beragam informasi untuk pengambilan keputusan.
3. Menghasilkan informasi untuk pihak eksternal
4. Menghasilkan informasi untuk penilaian kinerja karyawan atau divisi.
5. Menyediakan data masa lalu untuk kepentingan audit (pemeriksaan)

6. Menghasilkan informasi untuk penyusunan data evaluasi anggaran perusahaan.
7. Menghasilkan informasi yang diperlukan dalam kegiatan perencanaan dan pengendalian.

II.2. Hutang Piutang Usaha dalam Akuntansi

Menurut (Hery, 2011 : 202) Utang usaha (*accounts payable*) timbul pada saat barang atau jasa diterima sebelum melakukan pembayaran. Dalam transaksi perusahaan dagang, sering kali perusahaan membeli barang dagangan secara kredit dari pemasok untuk dijual kembali kepada para pelanggannya. Dalam hal ini, perusahaan akan mencatat pembelian barang dagangan tersebut dalam pembukuan dengan cara mendebet akun pembelian (jika sistem persediaan periodik) atau akun persediaan barang dagangan (jika sistem persediaan perpetual) dan mengkredit akun utang usaha. Utang usaha ini biasanya akan segera dilunasi oleh perusahaan dalam jangka waktu yang sangat singkat sesuai dengan persyaratan kredit (*credit term*) yang tertera dalam faktur tagihan (*invoice*). Pada waktu utang usaha jatuh tempo dan dilunasi, akun utang usaha akan didebet dan akun kas akan dikredit. Untuk pembayaran yang dilakukan dalam periode diskon, akun potongan pembelian (jika sistem persediaan periodik) atau akun persediaan barang dagangan (jika sistem persediaan perpetual) juga akan dikredit dalam jurnal sebesar potongan yang diterima.

Sedangkan pendapatan diterima di muka (*unearned revenue*) timbul pada saat pembayaran diterima sebelum barang atau jasa diberikan contohnya adalah

sewa diterima di muka (*unearned rent*), dimana pihak yang menyewakan biasanya akan menerima terlebih dahulu uang muka dari pihak penyewa untuk pemakaian sewa beberapa bulan kedepan. Uang yang diterima di muka ini, bagi pihak yang menyewakan (yang menerima uang muka) adalah merupakan utang, karena uang telah diterima atas periode sewa yang belum berjalan. Pihak yang menyewakan akan mencatat penerimaan uang muka tersebut dalam pembukuan dengan cara mendebet akun kas dan mengkredit akun sewa diterima dimuka. Untung secara berangsur-angsur akan berkurang secara proporsional dari bulan ke bulan sepanjang pemakaian sewa yang telah terjadi. Setiap bulannya, begitu sewa telah di dapatkan oleh penyewa, maka bagi pihak yang menyewakan (yang menerima uang muka tadi), utangnya akan berubah menjadi pendapatan sewa, yaitu sebesar sewa yang telah dinikmati oleh penyewa secara proporsional.

Contoh dalam pencatatan transaksi utang dalam akuntansi dapat di lihat pada gambar II.1. berikut ini

Tanggal	Nama Perkiraan	Debet	Kredit
1 Sept' 08	Kas Utang Wesel (Penerbitan Wesel)	100.000.000	100.000.000
31 Des'08	Beban Bunga Uang Bunga (bunga perjalanan = 4 bulan x 12% 12 bulan x 100 juta)	4.000.000	4.000.000
1 Jan'09	Utang Bunga Beban Bunga (jurnal Pembalik)	4.000.000	4.000.000
29 Feb'09	Utang Wesel Beban Bunga * Kas (Pembayaran utang wesel) *6 bulan x 12% 12 bulan x 100 juta	100.000.000 6.000.000	106.000.000

Gambar II.1. : Contoh Transaksi Utang Usaha

(Sumber : Hery, 2011 : 207)

Menurut (Hery, 2011 : 36) Istilah piutang mengacu pada sejumlah tagihan yang akan diterima oleh perusahaan (umumnya dalam bentuk kas) dari pihak lain, baik sebagai akibat penyerahan barang dan jasa secara kredit (untuk piutang pelanggan yang terdiri atas piutang usaha dan memungkinkan piutang wesel), memberikan pinjaman (untuk piutang karyawan, piutang debitur yang biasanya langsung dalam bentuk piutang wesel, dan piutang bunga), maupun sebagai akibat kelebihan pembayaran kas kepada pihak lain (untung piutang pajak)

1. Piutang Usaha (*Accounts Receivable*) adalah jumlah yang akan ditagih dari pelanggan sebagai akibat penjualan barang atau jasa secara kredit. Piutang usaha memiliki saldo normal untuk aktiva. Piutang usaha biasanya diperkirakan akan dapat ditagih dalam jangka waktu yang relatif pendek, biasanya dalam waktu 30 sampai 60 hari. Setelah ditagih, secara pembukuan, piutang usaha akan berkurang disebelah kredit. Piutang usaha diklasifikasikan dalam neraca sebagai aktiva lancar (*current asset*).

2. Piutang Wesel (*Notes Receivable*).

Yaitu tagihan perusahaan kepada pembuat wesel. Pembuat wesel tagihan disini adalah pihak yang telah berhutang kepada perusahaan baik melalui pembelian barang atau jasa secara kredit maupun melalui pinjaman sejumlah uang. Pihak yang berhutang berjanji kepada perusahaan (selaku pihak yang diuntungkan) untuk membayar sejumlah uang tertentu berikut bunganya dalam kurun waktu yang telah disepakati. Janji pembayaran tersebut ditulis secara formal dalam sebuah wesel atau promes (*promissory note*).

3. Piutang Lain-Lain (*other Receivables*)

Piutang lain-lain umumnya diklasifikasikan dan dilaporkan secara terpisah dalam neraca. Contohnya adalah piutang bunga, piutang *dividen* (tagihan kepada *investee* sebagai hasil atas investasi), piutang pajak (tagihan perusahaan kepada pemerintah berupa restitusi atau pengembalian atas kelebihan pembayaran pajak), dan tagihan kepada karyawan.

Sebagai ilustrasi dari penjelasan diatas,terdapat contoh kasus pada gambar II.2 berikut, asumsi bahwa PT. Lintas Indonusa, pada akhir tahun 2010 memiliki saldo piutang usaha sebesar Rp. 86.300.000,- dan cadangan kredit macet atas piutang usaha inidiestimasi berdasarkan masing-masing kelompok umurnya :

Kisaran Umur Piutang	Saldo	Estimas Kredit macet	
		Prosentase	Jumlah
Belum jatuh tempo	Rp. 75.000.000,-	2%	Rp.1.500.000,-
Telah jatuh tempo 1-30 hari	4.000.000,-	5%	200.000,-
Telah jatuh tempo 31-60 hari	3.100.000,-	10%	310.000,-
Telah jatuh tempo 61-90 hari	1.900.000,-	20%	380.000,-
Telah jatuh tempo 91-180 hari	1.200.000,-	30%	360.000,-
Telah jatuh tempo 181-365 hari	800.000,-	50%	400.000,-
Jatuh tempo di atas 365 hari	300.000,-	80%	240.000,-
	Rp. 86.300.000,-		Rp.3.390.000,-

Gambar II.2 : Contoh Trasaksi Piutang Usaha

(Sumber : Hery, 2011 : 54)

II.3. Sejarah *Visual Basic*

Menurut (Harip Santoso , 2006 : 3) Diluncurkan *VB.Net 2002* (Pemrograman berbasis Teknologi.Net), dapat dianggap sebagai pertanda berakhirnya pengembangan *Visual Studio 6*, karena secara *defacto* VB6 sudah tidak dikembangkan lagi oleh *Microsoft*. Walaupun VB6 tidak dikembangkan lagi, tetapi kesinambungan masa hidup VB6 masih dijaga oleh *Microsoft*.

Contohnya dengan diluncurkannya *Service Pack 5* untuk *Visual Studio 6* agar bahasa pemrograman yang terdapat pada *visual Studio 6* dapat mengakses *database SQL Server 2005*.

Kini teknologi pemrograman telah dialihkan ke pemrograman berbasis .NET, teknologi yang intinya adalah menulis program agar dapat dijalankan pada piranti lunak maupun keras yang berbeda tanpa harus mengubah program yang sudah dikompilasi. Misalnya, program yang dibuat pada sistem operasi *windows*, nantinya dapat dijalankan pada sistem operasi *Unix/Linux* dan sebaliknya, dimana hal ini dimungkinkan dengan diperkenalkannya konsep *.NET Framework*.

Service Pack yang dikeluarkan oleh *Microsoft* lebih pada penyediaan fasilitas piranti tambahan sebagai media untuk mengakses/membuat hubungan dengan piranti *software* yang baru, misalnya *SQL Server 2005*. jadi walaupun anda menginstal *Service Pack*, tidak berarti *Visual Studio 6* (VB 6) anda akan lebih canggih. itulah yang dimaksud dengan VB 6 tidak dikembangkan lagi oleh *Microsoft*.

Visual Basic 6 diluncurkan pada tahun 1998 dan terdiri dari tiga versi, yaitu *Learning Edition*, *Professional Edition*, dan *Enterprise Edition*. Adapun inti peningkatan yang terdapat pada VB 6 dibandingkan dengan versi sebelumnya adalah, peningkatan akses data, penambahan *tools*, dan kontrol baru untuk mengakses database, misalnya *ADODC*.

Awal sejarah *Visual Basic* dimulai pada pertengahan tahun 1991 dengan diluncurkannya VB versi 1.0. yang merupakan pengembangan dari *Quick Basic*, yaitu bahasa pemrograman *Basic* yang dimiliki oleh *Microsoft* untuk penulisan

program pada sistem operasi *DOS*. *QuickBasic* masih bersifat teks program, dan belum bersifat grafik. Singkatnya, piranti mouse belum dapat digunakan dalam kaitan dengan interaksi antara pengguna dan aplikasi program

VB 2.0 diluncurkan pada tahun 1992, dimana pada masa itu mulai diperkenalkan *ODBC (Open Database Connectivity)* sebagai metode untuk pengaksesan data, perkembangan VB 2.0 tidak terlalu nampak pada masa itu, karena bahasa pemrograman berbasis *xBase (dBase, foxpro, clipper)* mendominasi penulisan aplikasi program. Disamping itu sistem operasi *DOS* masih bergandengan erat dengan sistem operasi *Netware* yang dimiliki oleh *Novell* sebagai sistem operasi dasar pada aplikasi berbasis *LAN (Local Area Network)*. (Harip Santoso 2006 : 4).

II.4. Microsoft SQL Server

Menurut (Harip Santoso , 2006 : 5) Sejarah *SQL Server* berbeda jauh dengan sejarah *Visual Basic*. Bila *Visual Basic* berasal dari pengembangan *QuickBasic* yang juga merupakan produk *Microsoft*, maka *SQL Server* adalah hasil kerja sama antara *Microsoft* dengan *Sybase* untuk memproduksi sebuah software penyimpanan data (*database*) yang bekerja pada sistem operasi *OS/2*.

Sistem Operasi *OS/2* merupakan sistem operasi baru dari hasil kerja sama antara *Microsoft* dengan *IBM*. Sistem operasi *OS/2* mengenal bentuk-bentuk perintah *DOS*, sekaligus memiliki kemampuan *multitasking*. Untuk mendapat pengakuan pasar, maka *Microsoft* bekerja sama dengan *Ashton-Tate* yang telah dikenal dengan produk *dBase*. Sayangnya, kerja sama ini tidak berlangsung lama,

karena setelah peluncuran *SQL Server* versi 1.0 pada tahun 1989 kerja sama itu pun berakhir.

Peluncuran *SQL Server* 1.0 dilanjutkan dengan peluncuran *SQL Server* versi 1.1 pada tahun 1990. Adapun fitur terpenting dari produk ini adalah dukungan untuk *platform* baru dari sisi client yang dikenal sebagai sistem operasi Windows 3.1

Pada tahun 1991, *SQL Server* versi 1.11 diluncurkan dan berisi fasilitas perawatan database. Pada tahun yang sama, *Microsoft* mengembangkan sendiri sistem operasi *multi user* yang dikenal sebagai *Windows NT*, *Microsoft SQL Server* versi 4.2 dirilis pada tahun 1992 dan berisi *tool administrator* database berbasis *GUI-Windows*.

II.5. Database

Menurut (Budi Raharjo ; 2011 : 3) istilah database banyak memiliki definisi. Untuk sebagian kalangan sederhana *database* diartikan sebagai kumpulan data (buku, nomor telepon, daftar pegawai, dan lain sebagainya). Ada juga yang menyebut *database* dengan definisi lain yang lebih formal dan tegas. *Database* didefinisikan sebagai kumpulan data yang terintegrasi dan diatur sedemikian rupa sehingga data tersebut dapat dimanipulasi, diambil dan dicari secara cepat.

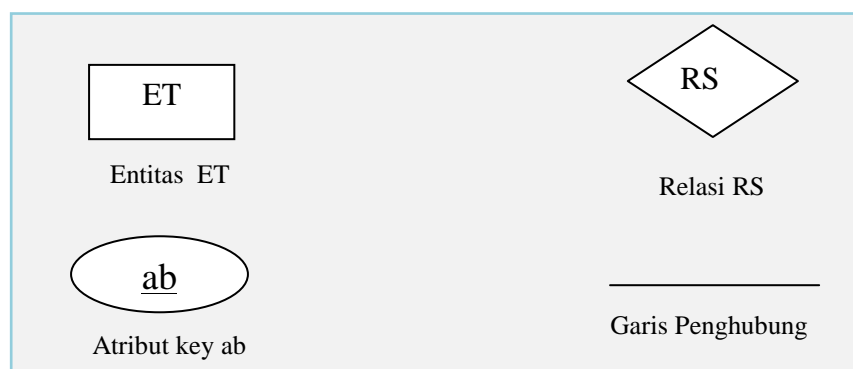
Selain berisi data, *database* juga berisi *metadata*. Metadata adalah data yang menjelaskan tentang struktur dari data itu sendiri. Sebagai contoh, Anda dapat memperoleh informasi tentang nama-nama kolom dan tipe yang ditampilkan tersebut disebut *metadata*.

II.5.1. Pemodelan Data

Menurut (Yudi Priyadi ; 2014 : 10) Terdapat beberapa penjelasan mengenai pemodelan basis data. Suatu basis data dapat digunakan secara bebas untuk menggambarkan dan memberikan deskripsi mengenai kumpulan informasi yang tersimpan dalam *data storage* komputer. Secara sederhana, definisi untuk model basis data adalah sekumpulan notasi atau simbol untuk menggambarkan data dan relasinya, berdasarkan suatu konsep dan aturan tertentu suatu pemodelan.

II.5.2. Notasi Diagram E-R

Menurut (Yudi Priyadi ; 2014 : 20) Pemodelan basis data dengan menggunakan diagram relasi antar entitas, dapat dilakukan dengan menggunakan suatu pemodelan basis data yang bernama Diagram *Entity-Relational* (selanjutnya disingkat Diagram E-R). Pada Gambar II.2, terdapat suatu simbol/notasi dasar yang digunakan pada Diagram E-R, yaitu entitas, relasi, atribut, dan garis penghubung.



Gambar II.2 : Notasi Dasar Diagram E-R

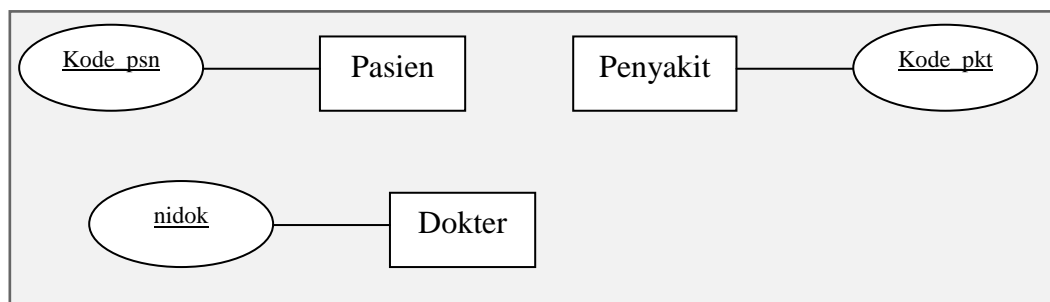
(Sumber : Yudi Priyadi ; 2014 : 20)

1. Entitas

Merupakan notasi untuk mewakili suatu objek dengan karakteristik sama, yang dilengkapi oleh atribut, sehingga pada suatu lingkungan nyata setiap objek akan berbeda dengan objek lainya. Pada umumnya, objek dapat berupa benda, pekerjaan, tempat dan orang.

2. Atribut

Merupakan notasi yang menjelaskan karakteristik suatu entitas dan juga relasinya. Atribut dapat sebagai key yang bersifat unik, yaitu *Primary Key* atau *Foreign Key*. Selain itu, atribut juga dapat sebagai atributdeskriptif saja, yaitu sebagai pelengkap deskripsi suatu entitas dan relasi.

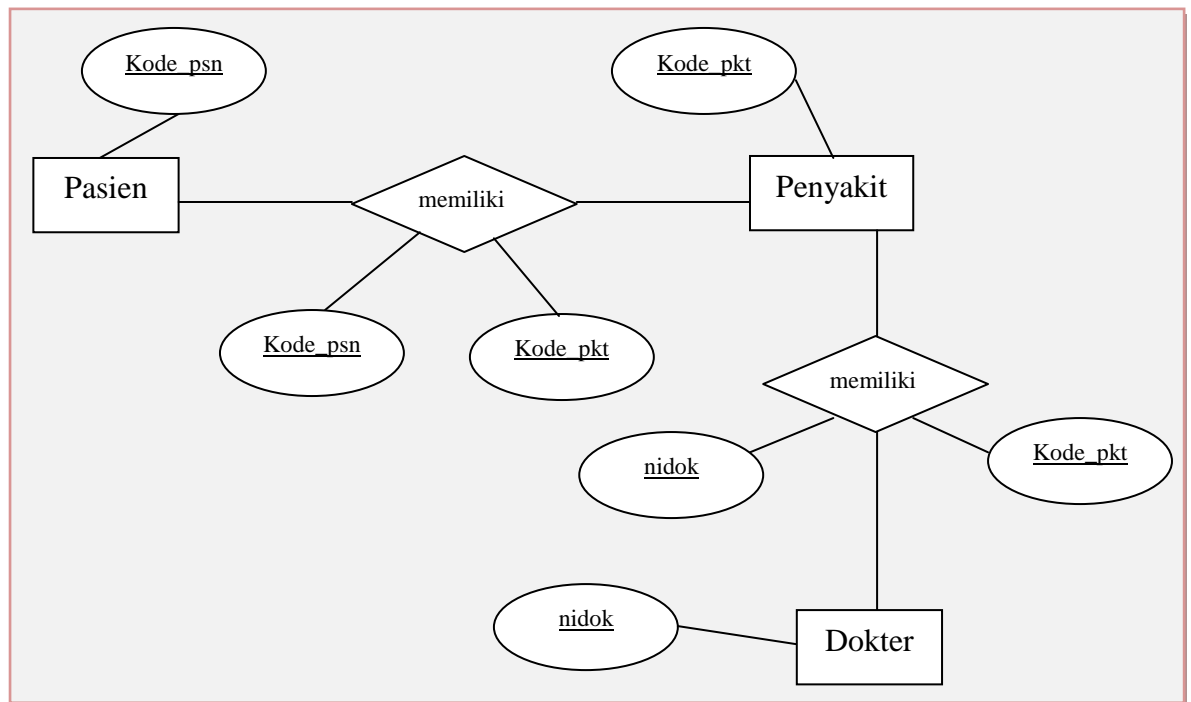


Gambar II.3 : Atribut Key pada Entitas

(Sumber : Yudi Priyadi ; 2014 : 23)

3. Relasi

Merupakan notasi yang digunakan untuk menghubungkan beberapa entitas berdasarkan fakta pada suatu lingkungan.



Gambar II.4 : Pemilihan Relasi untuk Entitas

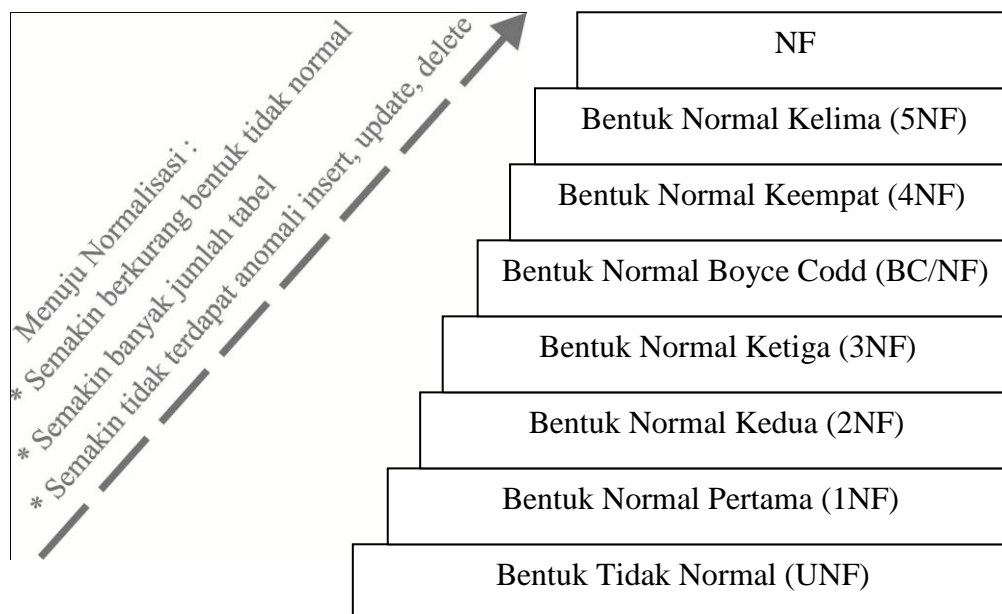
(Sumber : Yudi Priyadi ; 2014 : 25)

4. Garis penghubung

Merupakan notasi untuk merangkaikan keterkaitan antar notasi yang digunakan dalam Diagram E-R, yaitu entitas, relasi dan atribut.

II.5.3. Normalisasi

Menurut (Yudi Priyadi ; 2014 : 67) Normalisasi merupakan proses sistematis yang dilakukan pada struktur tabel basis data menjadi struktur tabel yang memiliki integritas data, sehingga tidak memiliki data anomali pada saat melakukan *insert*, *delete*, dan *update*. Pada Gambar II.4, tahapan proses sistematis yang dilakukan mulai dari bentuk tidak normal menjadi bentuk normal memiliki suatu syarat yang harus dipenuhi pada saat menuju suatu bentuk yang lebih baik (*well structured relation*).



Gambar II.5 : Tahapan Proses Bentuk Normalisasi

(Sumber : Yudi Priyadi ; 2014 : 67)

Setiap syarat dalam tahapan suatu bentuk normal memiliki keterkaitan, hal ini disebabkan karena pada setiap bentuk normal mengalami penyempurnaan untuk bentuk normal selanjutnya. Bentuk tidak normal akan semakin berkurang, setelah melalui tahapan perubahan bentuk normalisasi, sehingga berdampak pada jumlah tabel yang semakin banyak, tetapi menuju perbaikan ke dalam bentuk *well structured relation*. Hal ini terjadi akibat dari pengelompokan data suatu tabel agar memiliki ketergantungan secara fungsional.

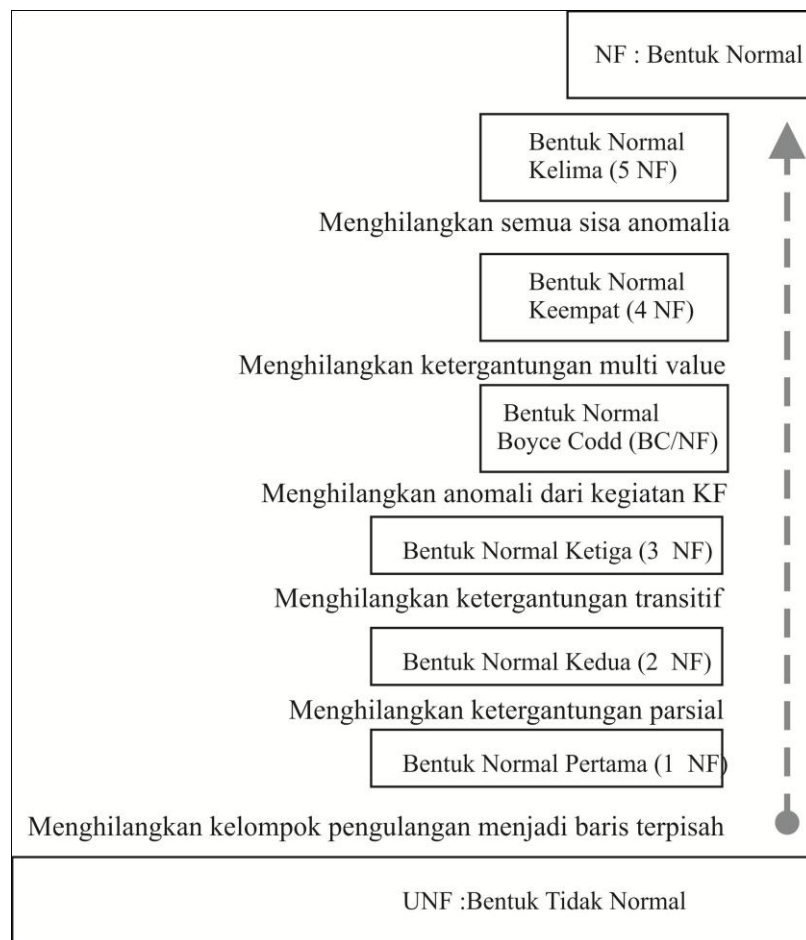
II.5.4. Aturan Proses Normalisasi

Menurut (Yudi Priyadi ; 2014 : 68) Secara sederhana, kegiatan normalisasi adalah melakukan dekomposisi atau penguraian tabel beserta datanya, menjadi tabel yang normal menurut konsep RDBMS. Merujuk pada gambar II.6, dekomposisi diawali dengan melakukan analisis pada suatu tabel atau beberapa

contoh formulir yang sudah memiliki data lengkap dalam basis data, tetapi masih dalam bentuk yang tidak normal (UNF). Oleh karena itu agar dapat memenuhi syarat bentuk normal pertama (1NF), pada setiap barisnya diisikan suatu *value* dengan kelompok data yang sama, berdasarkan suatu atribut *key*. Dengan demikian, kelompok pengulangan dalam suatu baris dapat dihilangkan, karena sudah tidak terdapat *value* yang kosong untuk setiap *field* dan *recordnya*

Setelah memenuhi syarat bentuk normal pertama (1NF), proses berikutnya adalah menghilangkan ketergantungan secara parsial, yaitu dengan cara melakukan dekomposisi tabel menjadi beberapa kelompok tabel berdasarkan *field* yang memiliki status sebagai *key*. Hal ini dapat dilakukan oleh salah satu *field* saja, dengan tetap tidak mengubah arti relasi dan ketergantungannya. Oleh sebab itu, disebut ketergantungan fungsional sebagian (*partiallly functional*), sehingga syarat bentuk normal kedua (2NF) sudah tercapai.

Bentuk normal kedua (2NF) merupakan syarat yang harus dimiliki untuk menuju bentuk normal ketiga (3NF). Pada proses ini, dilakukan dengan menghilangkan ketergantungan secara transitif, yaitu suatu konsep untuk tabel dari hasil relasi yang didalamnya terdapat ketergantungan secara tidak langsung pada beberapa atributnya. Pada umumnya proses normalisasi sudah dapat tercapai pada bentuk normal ketiga (3NF), yaitu dengan menghasilkan tabel yang tidak mengalami anomali basis data pada saat proses *insert*, *delete*, dan *update*.



Gambar II.6 : Tahapan Aturan Proses Normalisasi

(Sumber : Yudi Priyadi ; 2014 : 69)

II.6. *Unified Modeling Language (UML)*

Menurut (Rosa A.S & M. Shalahuddin ; 2011 : 118) Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakanteknik pemrograman berorientasi objek, yaitu *Unified Modeling Language (UML)*. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari

sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

UML hanya berfungsi untuk melakukan pemodelan, jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metode berorientasi objek.

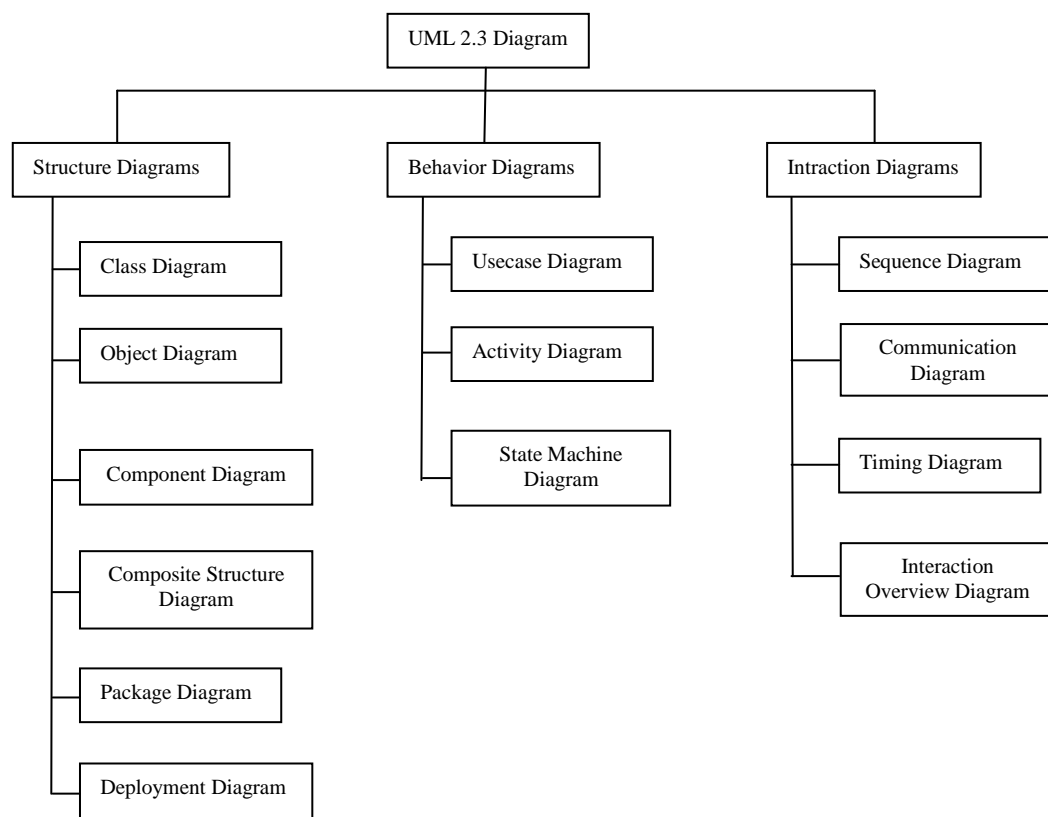
Menurut (Prabowo Pudjo Widodo & Herlawati ; 2011 : 6) UML diaplikasikan untuk maksud tertentu, biasanya antara lain :

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

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 berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota team untuk bekerja sama dengan bahasa model yang sama dengan mengaplikasikan beragam sistem. Intinya UML merupakan alat komunikasi yang konsisten dalam mendukung para pengembang sistem saat ini.

II.6.1 Diagram-Diagram UML

Menurut (Rosa A.S & M. Shalahuddin ; 2011 : 120) Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar II.7 di bawah ini



Gambar II.7 : Diagram UML

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 121)

Berikut ini penjelasan singkat dari pembagian kategori tersebut

1. *StructureDiagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

2. *Behavior Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

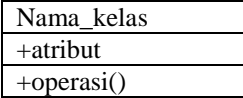




1. *Class Diagram*

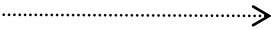
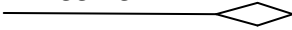
Diagram kelas atau *Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

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

Berikut gambar II.8 menerangkan simbol-simbol pada diagram kelas :

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / <i>interface</i> 	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
Asosiasi / <i>association</i>  Asosiasi berarah/ <i>directed association</i> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)

Kebergantungan 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / <i>aggregation</i> 	Semua bagian (<i>whole part</i>)


Gambar II.8 : Diagram Kelas

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 124)

2. *Object Diagram*

Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek harus dipastikan semua kelas yang sudah didefinisikan pada diagram kelas harus dipakai objeknya, karena jika tidak, pendefinisian kelas itu tidak dapat dipertanggung jawabkan. Untuk apa mendefinisikan sebuah kelas sedangkan pada jalannya sistem, objeknya tidak pernah dipakai.

Berikut adalah gambar II.9 menerangkan simbol-simbol diagram objek

Simbol	Deskripsi
Objek <div style="border: 1px solid black; padding: 2px; margin: 5px 0;"> Nama_objek : nama_kelas Atribut = nilai </div>	Objek dari kelas yang berjalansaat sistem dijalankan
Link 	Relasi antar objek

Gambar II.9 : Diagram Paket

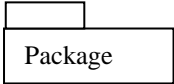

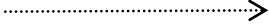


(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 124)

3. *Component Diagram*

Diagram komponen atau component diagram dibuat untuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. Komponen dasar yang biasanya ada dalam suatu sistem adalah sebagai berikut :

- 1) Komponen *user interface* yang menangani tampilan
- 2) Komponen *bussiness procesiing* yang menangani fungsi-fungsi proses bisnis
- 3) Komponen data yang menangani manipulasi data
- 4) Komponen *security* yang menangani keamanan sistem

Komponen lebih terfokus pada penggolongan secara umum fungsi-fungsi yang diperlukan, berikut gambar II.10 yang menerangkan simbol-simbol yang ada pada diagram komponen

Simbol	Deskripsi
Package 	<i>Package</i> merupakan sebuah bungkusan dari satu atau lebih komponen
Komponen 	Komponen Sistem
Kebergantungan / <i>dependency</i> 	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai
Antar muka / <i>interface</i>  nama_interface	Sama dengan konsep <i>interface</i> pada pemrograman berorientasi objek, yaitu sebagai antarmuka komponen agar tidak mengakses langsung komponen
Link 	Relasi antar komponen

Gambar II.10 : Diagram Komponen

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 126)

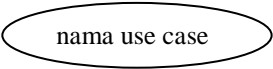


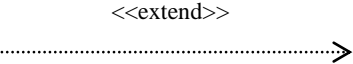
4. *Use Case Diagram*

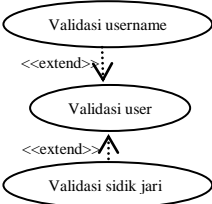
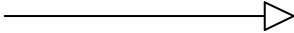
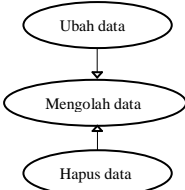
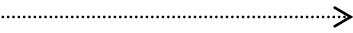
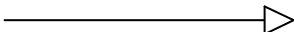
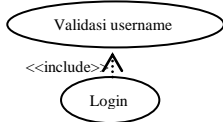
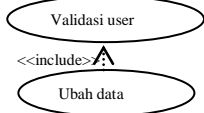
Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behaviour*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa

saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

- 1) Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
- 2) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut gambar II.11 menerangkan simbol-simbol pada diagram *use case*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama <i>use case</i>
Aktor / <i>actor</i>  nama aktor	Orang, proses, atau sistem yang lain berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan di buat itu sendiri
Asosiasi / <i>association</i> 	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> , atau <i>usecase</i> memiliki interaksi dengan aktor
Ekstensi / <i>extend</i> 	Relasi usecase tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanppa <i>use case</i> tambahan itu, mirip dengan prinsip inheritance pada pemrograman berorientasi objek, biasanya <i>use case</i> tambahan memiliki nama depan yang sama dengan <i>use case</i> yang ditambahkan misal

	 <p>arah panah mengarah pada <i>use case</i> yang ditambahkan</p>
<p>Generalisasi / <i>generalization</i></p> 	<p>Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya misalnya :</p>  <p>Arah panah mengarah pada use case yang menjadi generalisasinya (umum)</p>
<p>Menggunakan / <i>include</i> / <i>uses</i></p> <p>«include»</p>  <p>«uses»</p> 	<p>Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini</p> <p>Ada 2 sudut pandang yang cukup besar mengenai include di usecase</p> <ol style="list-style-type: none"> 1. include berarti use case yang ditambahkan akan selalu dipanggil saat use case dijalankan misal pada kasus berikut :  <ol style="list-style-type: none"> 2. include berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang di tambahkan telah di jalankan sebelum use case tambahan di jalankan, misal pada kasus berikut :  <p>Kedua interpretasi di atas dapat dianut</p>

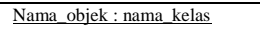

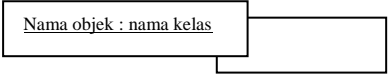


	salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.
--	---

Gambar II.11 : Diagram Use case

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 131)

5. Communication Diagram

Diagram komunikasi mengelompokkan message pada kumpulan diagram sekuen menjadi sebuah diagram. Dalam diagram komunikasi yang dituliskan adalah operasi / metode yang di jalankan antara objek yang satu dengan objek lainnya secara keseluruhan, oleh karna itu dapat di ambil dari jalanya interaksi pada semua diagram sekuen. Berikut adalah gambar II.12 yang menerangkan simbol-simbol yang ada pada diagram komunikasi :

Simbol	Deskripsi
Objek 	Objek yang melakukan interaksi pesan
Link 	Relasi antar objek yang menghubungkan objek satu dengan lainnya atau dengan dirinya sendiri 
Arah pesan / stimulus 	Arah pesan yang terjadi, jika pada suatu link ada dua arah pesan yang berbeda, maka arah juga deigambarkan dua arah pada dua sisi link 

Gambar II.12 : Diagram Komunikasi


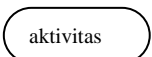



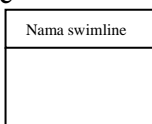
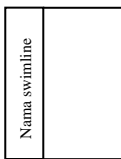
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 140)

6. Activity Diagram

- Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga banyak digunakan untuk mendefenisikan hal-hal berikut :

- 1) Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan
- 2) Urutan atau pengelompokan tampilan dari sistem/user interface dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan
- 3) Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

Berikut adalah gambar II.13 yang menggambarkan simbol-simbol yang ada pada diagram aktivitas :

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / decision 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / join 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi


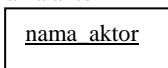

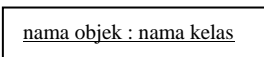

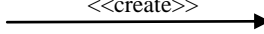
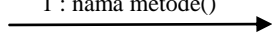
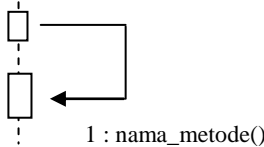
Gambar II.13 : Diagram Aktivitas

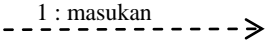
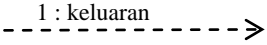
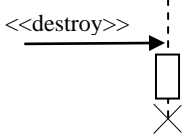
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 134)

8. Sequence Diagram

Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan

diterima antar objek. Banyaknya diagram objek yang digambarkan adalah sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefenisikan interaksi jalanya pesan sudah dicakup dapa diagram sekuen sehingga semakin banyak use case yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak. Berikut adalah gambar II.14 yang menerangkan simbol-sombol yang ada pada diagram sekuen :

Simbol	Deskripsi
<p>Aktor</p>  <p>nama aktor</p> <p>atau</p>  <p>tampa waktu aktif</p>	<p>Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya di nyatakan menggunakan kata benda di awali frase nama aktor</p>
<p>Garis hidup / lifeline</p> 	<p>Menyatakan kehidupan suatu objek</p>
<p>Objek</p> 	<p>Menyatakan objek yang berinteraksi pesan</p>
<p>Waktu aktif</p> 	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi pesan</p>
<p>Pesan tipe create</p> 	<p>Objek yang lain, arah panah mengarah pada objek yang dibuat</p>
<p>Pesan tipe call</p> 	<p>Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri</p>  <p>Arah panah mengarah pada objek yang memiliki operasi / metode, karena ini memanggil operasi / metode maka operasi /</p>

	metode yang di panggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Gambar II.14 : Sequence Diagram