BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem

Sistem adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu.

Sebagai contoh sistem pernafasan terdiri dari : hidung, tenggorokan, paru-paru, pembuluh darah dan darah (Tata Sutabri;S.Kom,MM;2005;6-9)

II.1.1.Karakteristik Sistem

a. Komponen *(Component)*

   Suatu sistem terdiri dari sejumlah yang saling berinteraksi, bekerja sama membentuk satu kesatuan.Komponen-kompones sistem dapat berupa suatu subistem atau bagian-bagian dari sistem.

b. Batas Sistem *(Boundary)*

   Batas sistem merupakan daerah yang membatasi anatara suatu sistem dengan sistem yang lain nya atau dengan lingkungan luar nya. Batas sistem ini memungkinkan suatu sistem yang dipandang sebagai susatu kesatuan, karena dengan batas sistem ini berfungsi dan tugas dari subsistem yang satu dengan yang lain nya berbeda tetapi tetap saling berinteraksi. Batas susatu sistem menunjukkan ruang lingkup *(scope)* dari sistem tersebut.

c. Lingkungan Luar Sistem *(Environment)*

*Environmen*t merupakan segala sesuatu diluar batas sistem yang mempengarui operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan dan dikendalikan agar tidak mengganggu operasi sistem.

d. Penghubung Sistem *(Interface)*

Merupakan media peghubung antara satu subsistem dengan subsistem yang lainnya untuk membentuk suatu kesatuan sehingga sumber-sumber daya mengalir dari subsitem yang satu ke subsistem yang lainnya. Dengan kata lain *Output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainnya.

e. Masukan Sistem *(Input)*

Merupakan energi yang dimasukkan kedalam sistem. Masukan dapat berupa masukan perawatan *(Maintenance Input)* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi.

f. Keluaran Sistem *(Output)*

Merupakan hasil dari energi yang diolah oleh sistem, meliputi *output* yang berguna yang berguna yang dikenal sebagai sisa pembuangan, contoh nya panas yang dikeluarkan oleh komputer.

g. Pengolahan Sistem *(Process)*

Merupakan bagian yang memproses masukan untuk menjadi keluaran yang diinginkan. Contoh CPU pada komputer, bagian produksi yang mengubah

bahan baku menjadi barang jadi, serta bagian akuntansi yang mengolah data transaksi menjadi laporan keuangan.

h. Sasaran Sistem *(Objective)*

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministic. Kalau suatu sistem tidak memiliki sasaran, maka operasi sistem tidak ada guna nya. Suatu sistem dikatakan berhasil bilamengenai sasaran atau tujuan yang telah di rencanakan. (Tata Sutabri;S.Kom,MM;2005;12-13)

II.2. Sekilas Mengenai *Logistic* (Gudang) Dan *Bad Stock* pada Distributor.

1. Pengertian Distributor.

Siapa yang dimaksud dengan Distributor itu? Sebelum mengetahui hal ini, marilah kita menelaah definisi Distributor menurut pakar yang ahli dalam bidang yang ahli terlebih dulu.Tofler dan Imber dalam buku nya yang terbitan tahun 2002 mengatakan bahwa: “Distributor adalah perusahaan atau perorangan yang bertindak sebagai perantara antara perusahaan manufaktur dan pengecer. Distributor mengadakan pergudangan untuk menyimpan barang dagangan, yang seringkali dibeli dari banyak perusahaan manufaktur berbeda, kemudian dijual (didistribusikan) kepada banyak pengecer maupun grosir”

Holy Icun Yunarto dalam bukunya sales dan *Distribution Management* dan mendefinisikan: “Distributor adalah *intermediary* yang menjalankan banyak fungsi distribusi seperti membeli barang dari produsen, menyimpan barang, memberikan kredit, dan lain-lain”. Jadi, jelaslah bahwa distributor itu sebenarnya usaha yang didirikan dengan tujuan untuk mendistribusikan barang miliknya ke

para penyalur berikut nya yang terdiri atas grosir maupun pengecer.Sedangkan, barang di gudang Distributor adalah aneka barang yang di peroleh dari berbagai *Principal* yang bekerjasama dengannya. Kalau melihat tugas nya, distributor secara tidak langsung adalah tangan kanan *Principal* dalam memasarkan barang di wilayah pemasaran yang telah di tunjuk oleh *Principal*.(Frans M. Royan;2009;70)

2. Bagian *Logistic*

Berikut adalah bagian-bagian dari logistic :

a. Adiministrasi *Logistic* memiliki tugas mencatat setiap barang yang masuk dan keluar dari gudang. Entah itu barang yang dating dari *Principal,* barang yang dikirimkan ke pelanggan, atau barang kembali (*Return good Stock* dan *Bad stock*). Ia juga mencatat faktur-faktur yang telah di realisasi pengirimannya. Kemudian, ia bertanggung jawab pula terhadap keakuratan pencatatan ke pembukuan *logistic,* dan bertanggung jawab kepada atasannya (kepala bagian logistik) terhadap aktifitas pencatatan stok, barang kembali, dan selisih.
b. *Driver* bertugas mengawasi proses muat barang di boks, mengirimkan barang ke pelanggan, menerima barang kembali, dan menurunkan barang dari boks untuk pelanggan sesuai faktur. Ia bertanggung jawab megirim dengan cermat karena kekeliruan dalam pengiriman akan menjadi tanggung jawab nya.
c. *Helper* bertugas membantu menaikkan maupun menurunkan barang dari boks, membantu mengatur barang di gudang, mengambil barang kembali dari pelanggan, dan bertugas untuk mengamankan mobil. Ia bertanggung jawab pada sopir atas barang-barang yang dikirimkannya, juga barang kembali yang diterimanya. (Frans M. Royan;2011;84)

II.3. *Entity Relationship Diagram* (ERD)

Merupakan suatu model untuk menjelaskan hubungan antar dua dalam basis data berdasarkan suatu persepsi bahwa *real word* tersiri dari objek-objek dasar yang mempunyai hibungan atau antar objek-objek tersebut. Relasi antar objek dengan menggunakan symbol-simbol atau garis tertentu.

Entity Relationship Model merupakan suatu model data yang dikembangkan berdasarkan objek (Linda Marlinda, S. Kom;2014;16)

II.3.1. Komponen-komponen yang terdapat dalam *Entity Relationship Model.*

1. Entity

a. Adalah suatu yang dapat dibedakan dalam dunia nyata dimana informasi yang berkaitan dengan yang dikumpulkan

b. *Entity* set adalah kumpulan *entity* yang sejenis.

c. Simbol yang digunakan untuk *entity* adalah persegi panjang.

d. *Entity* set dapat berupa :

– *Entity* yang bersifat fisik, yaitu *entity* yang dapat dilihat.

  Contohnya : rumah, kendaraan, mahasiswa, dosen, dan lain-lain

– *Entity* yang bersifat konsep atau l*ogic*, yaitu *entity* yang tidak dapat dilihat.

  Contoh nya : pekerjaan, perusahaan, rencana, matakuliah, dan lain-lain.

– Simbol yang digunakan untuk *entity* adalah persegi panjang.

  Untuk melihat gambar *entity* ini, lihat gambar II.1. sebagai berikut :

Gambar II.1. *Entity*

Sumber : Linda Marlinda, S. Kom (2004;17)

2. Relationship.

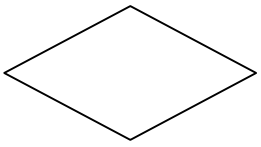
a. Adalah hubungan yang terjadi antara satu atau lebih *entity*

b. *Relationship* tidak memiliki keberadaan fisik, kecuali yang mewarisi hubungan antara *entity* tersebut.

c. *Relationship* set adalah kumpulan *relationship* yang sejenis.

d. Simbol yang digunakan adalah bentuk belah ketupat, *diamond* atau *rectangle*

Untuk melihat gambar relatitionship ini, lihat gambar II.2. sebagai berikut:



Gambar II.2. *Relationship*

Sumber : Linda Marlinda, S.Kom (2004:18)

3. Atribute

a. Adalah karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang atau *relationship* tersebut.

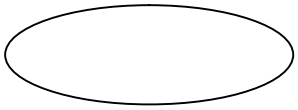
b. *Atribute value* atau nilai *atribute* (nilai *attribute*) adalah suatu data aktual atau informasi yang disimpan di suatu *entity* atau *relationship.*

c. Terdapat dua jenis *attribute,* yaitu:

1. *Indentifer* (key), untuk menetukan suatu *entity* secara unik.
2. *Descriptor (nonkey attribute),* untuk menentukan karakteristik dari suatu *entity* yang tidak unik.

d. Simbol yang digunakan adalah dalam bentuk oval.

Untuk melihat gambar attribute ini, lihat pada gambar II.3. sebagai berikut:



Gambar II.3. *Atribute*

Sumber : Linda Marlinda, S.Kom (2004:18)

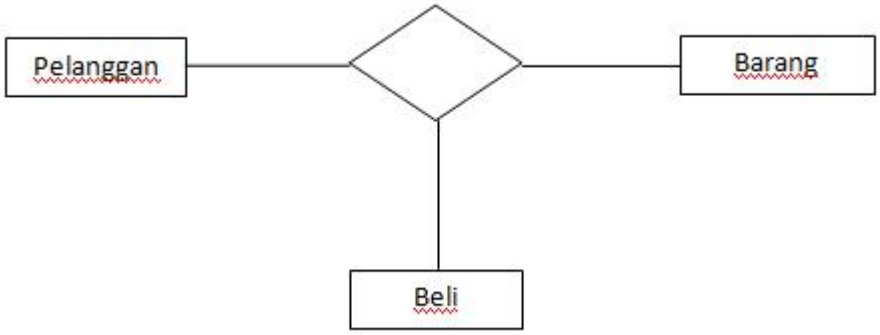
4. *Indicator* Tipe

a. *Indicator type associative object*

Berfungsi sebagai suatu objek dan suatu *relationship.* Untuk melihat gambar *incator type* ini, lihat pada gambar II.4. sebagai berikut:

Contoh :

Menjadi:



Gambar II.4. *Indicator Type*

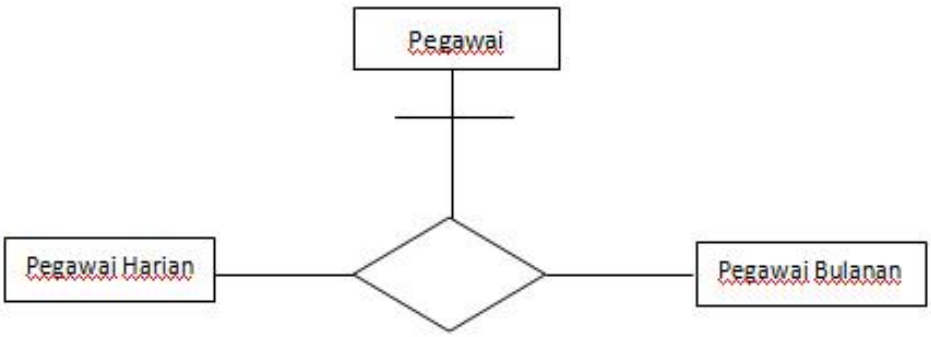
Sumber : Linda Marlinda, S.Kom (2004:18)

b. *Indicator* tipe supertipe

Terdiri dari suatu objek dan sub kategori atau lebih yang di hubungkan dengan *relationship* yang bernama (Linda Marlindo, S.Kom 2004:17-19).

Untuk melihat gambar *indicator* tipe supertipe ini, lihat gambar II.5. sebagai berikut:

Contoh:



Gambar II.5. *Indicator Type*

Sumber : Linda Marlinda, S.Kom (2004:19)

II.4. Kamus Data

Kamus data (KD) atau *dictionary* (DD) atau disebut juga dengan istilah sistem atau *dictionary* adalah katalog fakta tentang data atau kebutuhan-kebutuhan atau informasi dari suatu sistem informasi"(Jogiyanto:2005;725)

KD dibuat pada tahap analis sistem dan digunakan baik pada tahap analis maupun pada tahap perancangan sistem.

II.5. Normalisasi

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

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang telah ditemukan (Janner Simarmata & Imam Prayudi; 2006: 76) .

1. **Bentuk Normal Pertama (1NF/*First Normal Form*)**, bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues atribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal (mungkin saja nilai *null*) pada perpotongan setiap baris dan kolom satu nilai untuk irisan baris dan kolom pada tabel.

2. **Bentuk Normal Kedua (2NF/*Second Normal Form*)**, semua kebergantungan fungsional (*functional dependeny*) yang bersifat sebagian (*partialfunctional dependeny*) telah dihilangkan.
3. **Bentuk Normal Ketiga (3NF/*Third Normal Form*)**, semua kebergantungan transitif (*transitivedependeny*) telah dihilangkan.
4. **Boyce-Codd Normal Form (*BCNF/Boyce-Codd Normal Form*)**, semua anomali yang tersisa dari hasil penyempurnaan kebergantungan fungsional (*functional dependeny*) diatas telah dihilangkan.
5. **Bentuk Normal Keempat (4NF/*Fifth Normal Form*)**, semua anomali yang berasal dari kebergantungan banyak-nilai (*multivalues dependeny*) telah dihilangkan (Adi Nugroho; 2010: 34).

Tujuan normalisasi adalah membuat kumpulan tabel relasional yang bebas dari data berulang yang dapat dimodifikasi secara benar dan konsisten. Ini berarti bahwa semua tabel pada basisdata relasional harus berada pada bentuk normal ketiga (3NF). Sebuah tabel relasional berada pada 3NF jika dan hanya jika semua kolom bukan kunci adalah (a) saling independen dan (b) sepenuhnya tergantung pada kunci utama. Saling independen berarti bahwa tidak ada kolom bukan kunci yang tergantung pada senbarang kombinasi kolom lainnya. Dua bentuk normal pertama adalah langkah antara untuk mencapai tujuan, yaitu mempunyai semua tabel dalam 3NF (Stephens and Plew, 2000) (Janner Simarmata & Imam Prayudi ; 2006: 77).

II.6. Sekilas Tentang *Visual Basic*

Mocrosoft Visual studio adalah sebuah *integrated Development Environment* buatan *Microsoft Corporation*.Microsoft Visual Studio dapat digunkan untuk mengembangkan aplikasi dalam native code (dalam bentuk bahasa mesin yang yang berjalan diatas Windows) ataupun *managed code* (dalam bentuk *Microsoft Intermediate Language* di atas.NET Framwork). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan diatas di atas.NET Framwork). Visual Studio mencakup sebuah kode editor yang didukung oleh fitur *intellisense* atau yang disebut dengan *code refactoring. Debugge*r telah terintegrasi bekerja pada level source level *debugger* dan level *debugger* mesin. Tool Built ini mencakup form desainer untuk membangun sebuah aplikasi GUI, web desainer, class desainer dan database *scheme* desainer. (Wahana Komputer;2012;2)

II.7. Sekilas Tentang *SQL Server*

Sql Server 2000 terdiri atas beberapa edisi, yaitu *enterprise Edition*, *Standart Edition*, *Personal Edition*, *Developer Edition*, *Microsoft Dekstop Engine* (MSDE), *Windows CE Edition* dan *Evaluation Edition.*

SQL Server 2000 *Enterprise Edition* dan *Standart Edition* lebih cocok untuk dijalankan pada sistem operasi server seperti Windows 2000 Server atau Windows Server 2003. *Personal Edition* dapat dijalankan pada sistem operasi non server seperti Windows 98, Windows 2000 Profesional, dan Windosw XP.

Penulis menyarankan anda untuk menggunakan *SQL server* 2000 *Enterprice Edition* agar dapat memperoleh semua kemampuan yang ditawarkan oleh *SQL Server* 2000. (PT. Elexmedia Komputindor;2005;5)

II.8. Pengenalan UML

UML *(Unifield Modeling Language)* adalah suatu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek (Munawar ; 2005 : 17). Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang 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 mereka dengan yang lain. Adapun tipe diagram UML yang ada seperti pada tabel II.1.

Tabel II.1. Tipe Diagram UML

Diagram	Tujuan	Keterangan
Activity	Prilaku prosedural dan parallel	Sudah ada di UML 1
Class	Class, fitur dan relasinya	Sudah ada di UML 1
Communication	Interaksi diantara objek. Lebih menekankan kepada link	Di UML 1 disebut collaboration
Component	Struktur dan koneksi dari komponen	Sudah ada di UML 1
Composite Structure	Dekomposisi sebuah class saat runtime	Baru untuk UML 2
Deployment	Penyebaran/instalasi ke klien	Sudah ada di UML 1
Interaction	Gabungan dari activity dan	Baru untuk UML 1

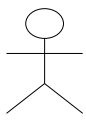
Overview	sequence diagram	
Object	Contoh konfigurasi instance	Tidak resmi ada di UML 1
Package	Struktur hierarki saat kompilasi	Tidak resmi ada di UML 1
Sequence	Interaksi antara objek. Lebih menekankan pada urutan.	Sudah ada di UML 1
State Machine	Bagaimana event mengubah sebuah objek	Sudah ada di UML 1
Timing	Interaksi antar objek. Lebih menekankan pada waktu	Sudah ada di UML 1
Use Case	Bagaimana user berinteraksi dengan sebuah sistem	Sudah ada di UML 1

Sumber : Munawar (2005 : 23)

II.10.1. Notasi Dasar UML

1. Aktor

Aktor adalah abstration dari orang dan system yang lain yang mengaktifkan fungsi dari target sistem. Orang atau system bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*. Berikut notasi *aktor* dalam UML :



Gambar II.6 Notasi Actor pada UML

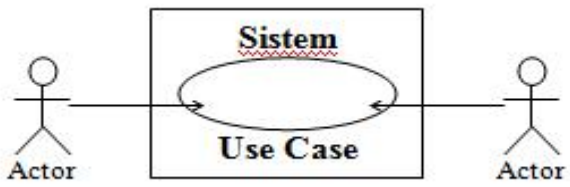
Sumber : Munawar (2005: 64)

1. Class Diagram

Class, dalam notasi UML digambarkan dengan kotak. Nama *class* menggunakan huruf besar diawal kalimatnya dan diletakkan diatas kotak. Bila *class* mempunyai nama yang terdiri dari 2 suku kata atau lebih, maka semua suku kata digabungkan tanpa spasi dengan huruf awal tiap suku kata menggunakan huruf besar. Notasi *class* dalam UML dapat dilihat pada gambar II.2 berikut :

1. Use case

Use case adalah alat bantu terbaik guna menstimulasi pengguna potensi untuk mengatakan tentang suatu system dari sudut pandangnya. Tidak selalu mudah bagi pengguna untuk menyatakan bagaimana mereka bermaksud menggunakan sebuah sistem. Karena system pengembangan tradisional sering ceroboh dalam melakukan analisis, akibatnya pengguna seringkali susah menjawabnya tatkala diminta masukan tentang sesuatu. Notasi *use case* dapat dilihat pada gambar II.3 :



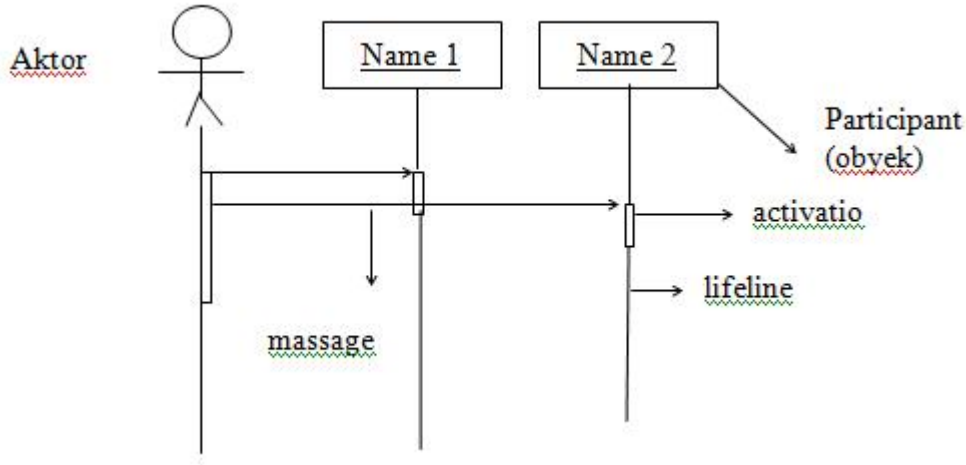
Gambar II.7. Notasi *Use Case* pada UML

Sumber : Munawar (2005 : 64)

1. *Sequence* Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan jumlah contoh objek dan messege (pesan) yang diletakkan diantara objek-objek ini dalam *use case* .

Komponen utama *sequence* diagram terdiri atas objek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress *vertical.*



Gambar II.8 Contoh *Sequence Diagram*

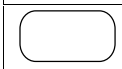
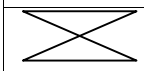
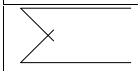
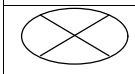
Sumber : Munawar (2005 : 89)

1. *Activity* Diagram

Actifity Diagram adalah teknik untuk mendiskripsikan logika prosedural. Proses bisnis dan aliran dalam banyak kasus. *Actifity* Diagram mempunya seperti halnya *flowchart,* akan tetapi perbedaannya dengan *flowchart* adalah *actifity* diagram bisa mendukung prilaku pararel sedangkan flowchart tidak bisa. Simbol-

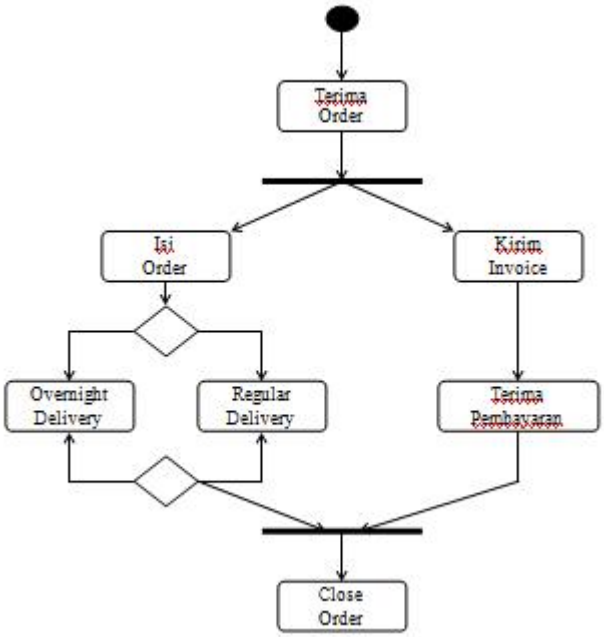
simbol yang sering digunakan pada saat pembuatan actifity diagram dapat dilihat pada tabel II.8. berikut :

Tabel II.2. Simbol-simbol yang sering dipakai pada *Activity Diagram*

Simbol	Keterangan
	Titik awal
	Titik akhir
	Activity
	Pilihan untuk pengambilan keputusan
	Fork; digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	Rake; menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (Flow Final)

Sumber :Munawar (2005 : 109)

Adapun contoh dari *Activity Diagram* dapat di lihat pada Gambar II.10



Gambar II.10 Contoh *Activity* Diagram Sederhana

Sumber :Munawar (2005: 111)