BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem

Sistem merupakan sekumpulan elemen-elemen yang saling terintegrasi serta melaksanakan fungsinya masing-masing untuk mencapai tujuan yang telah ditetapkan. Karakteristik sistem terdiri dari :

1. Komponen Sistem

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

2. Batasan Sistem

   Batasan merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang suatu kesatuan. Batasan suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkung luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

4. Penghubung Sistem

   Penghubung merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lainnya.

5. Masukan Sistem

   Masukan sistem adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintance input*) dan masukan sinyal (*signal input*).

6. Keluaran Sistem

   Keluaran sistem adalah hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolah Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah akan mengubah masukan menjadi keluaran.

8. Sasaran Sistem

   Suatu sistem mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya (Sulindawati, 2010 : 135).

II.2. Sistem Pakar

Sistem pakar menirukan perilaku seorang pakar dalam menangani suatu persoalan. Pada suatu kasus seorang pasien mendatangi dokter untuk memeriksa

badannya yang mengalami gangguan kesehatan, maka dokter atau pakar kesehatan akan memeriksa dan melakukan diagnosa.

Bila dokter cukup sibuk dan pelaksana diagnosa digantikan oleh sebuah sistem pakar, maka sistem pakar diharapkan dapat membantu memahami dan menganalisa keadaan pasien dan menemukan penyakit yang diderita pasien itu. Sistem pakar diharapkan juga untuk menghasilkan dugaan atau hasil diagnosa yang sama dengan diagnosa yang dilakukan oleh seorang ahli. Tujuan utama sistem pakar bukan untuk menggantikan kedudukan seorang ahli maupun pakar, tetapi untuk memasyarakatkan pengetahuan dan pengalaman pakar-pakar yang ahli di bidangnya (Andri Saputra, 2011 : 203).

II.2.1. Konsep Dasar Sistem Pakar

Konsep dasar sistem pakar mengandung : keahlian, ahli, pengalihan keahlian, inferensi, aturan dan kemampuan menjelaskan. Keahlian adalah suatu kelebihan penguasaan pengetahuan di bidang tertentu yang diperoleh dari pelatihan, membaca atau pengalaman. Contoh bentuk pengetahuan yang termasuk keahlian adalah :

a. Fakta-fakta pada lingkup permasalahan tertentu.
b. Teori-teori pada lingkup permasalahan tertentu.
c. Prosedur-prosedur dan aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
d. Strategi-strategi global untuk menyelesaikan masalah.
e. Meta-knowledge (pengetahuan tentang pengetahuan).

Bentuk-bentuk ini memungkinkan para ahli untuk dapat mengambil keputusa lebih cepat dan lebih baik daripada seseorang yang bukan ahli. Seorang ahli adalah seseorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan (domain), menyusun kembali pengetahuan jika dipandang perlu, memecah aturan-aturan jika dibutuhkan, dan menentukan relevan tidaknya keahlian mereka. Pengalihan keahlian dari para ahli ke komputer untuk kemudian dialihkan lagi ke orang lain yang bukan ahli, merupakan tujuan utama dari sistem pakar. Proses ini membutuhkan 4 aktivitas yaitu :

a. Tambahan pengetahuan (dari para ahli atau sumber-sumber lainnya).
b. Representasi pengetahuan (ke komputer).
c. Inferensi pengetahuan.
d. Pengalihan pengetahuan ke user.

Pengetahuan yang disimpan di komputer disebut dengan nama basis pengetahuan. Ada 2 tipe pengetahuan, yaitu : fakta dan prosedur (biasanya berupa aturan). Salah satu fitur yang harus dimiliki oleh sistem pakar adalah kemampuan untuk menalar. Jika keahlian-keahlian sudah tersimpan sebagai basis pengetahuan dan sudah tersedia program yang mampu mengakses basisdata, maka komputer harus dapat diprogram untuk membuat inferensi. Proses inferensi ini dikemas dalam bentuk motor inferensi (inference engine). Sebagian besar sistem pakar komersial dibuat dalam bentuk rule-based systems, yang mana pengetahuannya disimpan dalam bentuk aturan-aturan. Aturan tersebut biasanya berbentuk IF-

THEN. Fitur lainnya dari sistem pakar adalah kemampuan untuk merekomendasi. Kemampuan inilah yang membedakan sistem pakar dengan sistem konvensional.

Ada 4 bentuk sistem pakar, yaitu :

a. Berdiri sendiri. Sistem pakar jenis ini merupakan software yang berdiri-sendiri tidak tergantung dengan software yang lainnya.
b. Tergabung. Sistem pakar jenis ini merupakan bagian program yang terkandung didalam suatu algoritma (konvensional), atau merupakan program dimana didalamnya memanggil algoritma subrutin lain (konvensional).
c. Menghubungkan ke software lain . Bentuk ini biasanya merupakan sistem pakar yang menghubungkan ke suatu paket program tertentu, misalnya DBMS.
d. Sistem Mengabdi. Sistem pakar merupakan bagian dari komputer khusus yang dihubungkan dengan suatu fungsi tertentu. Misalnya sistem pakar yang digunakan untuk membantu menganalisis data radar.

Sistem pakar teridiri-dari 2 bagian pokok, yaitu : lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan sebagai pembangunan sistem pakar baik dari segi pembangunan komponen maupun basis pengetahuan. Lingkungan konsultasi digunakan oleh seorang yang bukan ahli untuk berkonsultasi.

Basis pengetahuan berisi pengetahuan-pengetahuan dalam penyelesaian masalah, tentu saja di dalam domain tertentu. Ada 2 bentuk pendekatan basis pengetahuan yang sangat umum digunakan, yaitu :

a. Penalaran berbasis aturan (*Rule-Based Reasoning*)

   Pada penalaran berbasis aturan, pengetahuan direpresentasikan dengan menggunakan aturan berbentuk : IF-THEN. Bentuk ini digunakan apabila kita memiliki sejumlah pengetahuan pakar pada suatu permasalahan tertentu, dan si pakar dapat menelesaikan msalah tersebut secara berurutan. Disamping itu, bentuk ini juga digunakan apabila dibutuhkan penjelasan tentang jejak (langkah-langkah) pencapaian solusi.

b. Penalaran berbasis kasus (*Case-Based Reasoning*).

   Pada penalaran berbasis kasus, basis pengetahuan akan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu solusi untuk keadaan yang terjadi sekarang (fakta yang ada). Bentuk ini digunakan apabila user menginginkan untuk tahu lebih banyak lagi pada kasus-kasus yang hampir sama (mirip). Selain itu, bentuk ini juga digunakan apabila kita telah memiliki sejumlah situasi atau kasus tertentu dalam basis pengetahuan.

Ada 2 cara yang dapat dikerjakan dalam melakukan inferensi, yaitu :

a. Forward Chaining. Pencocokan fakta atau pernyataan dimulai dari bagian sebelah kiri (IF dulu). Dengan kata lain, penalaran dimulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis.

b. Backward Chaining. Pencocokan fakta atau pernyataan di mulai dari bagian sebelah kanan (THEN dulu). Dengan kata lain, penalaran dimulai dari hipotesis terlebih dahulu, dan untuk menguji kebenara hipotesis tersebut dicari harus dicari fakta-fakta yang ada dalam basis pengetahuan

Sistem pakar yang baik harus memenuhi ciri-ciri sebagai berikut :

a. Memiliki fasilitas informasi yang handal.
b. Mudah dimodifikasi.
c. Dapat digunakan dalam berbagai jenis komputer.
d. Memiliki kemampuan untuk belajar beradaptasi.

Ada beberapa masalah yang menjadi area luas aplikasi sistem pakar, antara lain :

a. Interpretasi. Pengambilan keputusan dari hasil observasi, termasuk diantaranya : pengawasan, pengenalan ucapan, analisis citra, interpretasi sinyal, dan beberapa analisis kecerdasan.
b. Prediksi. Termasuk diantaranya : peramalan, prediksi demografis, peramalan ekonomi, prediksi lalulintas, estimasi hasil, militer, pemasaran, atau peramalan keuangan
c. Diagnosis. Termasuk diantaranya : medis, elektronis, mekanis, dan diagnosis perangkat lunak.
d. Perancangan. Termasuk diantaranya : layout sirkuit dan perancangan bangunan.
e. Perencanaan. Termasuk diantaranya : perencanaan keuangan, komunikasi, militer, pengembangan produk, routing, dan manajemen proyek.
f. Monitoring. Misalnya : Computer-Aided Monitoring Systems.
g. Debugging, memberikan resep obat terhadap suatu kegagalan.
h. Perbaikan.
i. Instruksi. Melakukan instruksi untuk diagnosis, debungging, dan perbaikan kerja.

j. Kontrol. Melakukan kontrol terhadap interpretasi interpretasi, prediksi, perbaikan, dan monitoring kelakukan sistem.

Secara garis besar, banyak manfaat yang dapat diambil dengan adanya sistem pakar, antara lain :

a. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
b. Bisa melakukan proses secara berulang secara otomatis.
c. Menyimpan pengetahuan dan keahlian para pakar.
d. Meningkatkan output dan produktivitas.
e. Meningkatkan kualitas.
f. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
g. Mampu beroperasi dalam lingkungan yang berbahaya.
h. Memiliki kemampuan untuk mengakses pengetahuan.
i. Memiliki reliabilitas.
j. Meningkatkan kepabilitas sistem komputer.
k. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
l. Sebagai media pelengkap dalam penelitian.
m. Meningkatkan kapabilitas dalam penyelesaian masalah.
n. Menghemat waktu dalam pengambilan keputusan.

Disamping memiliki beberapa keuntungan, sistem pakar juga memiliki beberapa kelemahan, antara lain :

a. Biaya yang diperlukan untuk membuat dan memeliharanya sangat mahal.

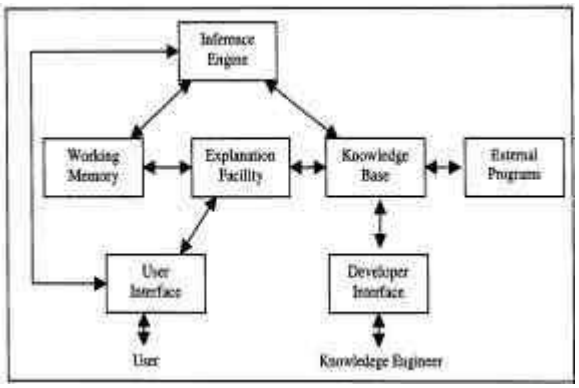
b. Sulit dikembangkan. Hal ini tentu saja erat kaitannya dengan ketersediaan pakar dibidangnya.
c. Sistem Pakar tidak 100% bernilai benar.

II.2.2. Perbandingan sistem konvensional dengan sisrem pakar

Sistem Konvensional Informasi dan pemrosesannya biasanya jadi satu dengan program Biasanya tidak bisa menjelaskan mengapa suatu input data itu dibutuhkan, atau bagaimana output itu diperoleh Pengubahan program cukup sulit & membosankan Sistem hanya akan beroperasi jika sistem tersebut sudah lengkap Eksekusi dilakukan langkah demi langkah Menggunakan data Tujuan utamanya adalah efisiensi Sistem Pakar Basis pengetahuan merupakan bagian dari mekanisme inferensi Penjelasan adalah bagian terpenting dari sistem pakar Pengubahan aturan dapat dilaksanakan dengan mudah Sistem dapat beroperasi hanya dengan beberapa aturan. Eksekusi dilakukan pada keseluruhan basis pengetahuan Menggunakan pengetahuan Tujuan utamanya adalah efektivitas (Ari Fadli, 2010 : 2).

II.2.3. Arsitektur Sistem Pakar

Arsitektur sistem pakar dapat dilihat pada gambar 1 di bawah ini dimana sebuah sistem pakar terdiri dari tiga modul utama, yaitu: knowledge base, working memory dan inference engine yang merupakan bagian utama dari sebuah sistem pakar. Sedangkan bagian-bagian selain ketiga komponen utama itu adalah : user interface, developer interface, explanation facility, dan external programs.



Gambar II.1. Arsitektur Sistem Pakar

II.3. Naïve Bayes

Naïve Bayes merupakan pendekatan statistik untuk melakukan inferensi induksi pada persoalan klasifikasi. Metode ini menggunakan probabilitas bersyarat sebagai dasarnya. Dalam ilmu statistik, probabilitas bersyarat. Naïve Bayes merupakan salah satu metode machine learning yang menggunakan perhitungan probabilitas. Algoritma ini memanfaatkan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi probabilitas di masa depan berdasarkan pengalaman di masa sebelumnya. Dasar dari teorema naïve bayes yang dipakai dalam pemrograman adalah rumus Bayes:

$P(A|B) = (P(B|A) * P(A))/P(B)$..... (1)

Peluang kejadian A sebagai B ditentukan dari peluang B saat A, peluang A, dan peluang B. Pada pengaplikasiannya nanti rumus ini berubah menjadi :

$P(Ci|D) = (P(D|Ci)*P(Ci)) / P(D)$…....(2)

Naïve Bayes Classifier atau bisa disebut sebagai multinomial naïve bayes merupakan model penyederhanaan dari algoritma bayes yang cocok dalam pengklasifikasian text atau dokumen. (Reggy Pasya Trinanda ; 2014 : 2).

II.4. Javascript

Javascript bukan bahasa berorientasi objek, melainkan bahasa berbasis objek. Bahasa berorientasi objek harus mendukung tiga konsep dasar, yaitu pengkapsulan (*encapsulation*), perwarisan (*inheritance*) dan polimorfisme (*polymorphism*). Javascript hanya mendukung pengkapsulan, itupun tidak 100% benar. Program *JavaScript* dituliskan pada file *HTML* (*.html* atau *.htm*) dengan menggunakan tag container <*SCRIPT*>. Dengan kata lain, anda tidak perlu menuliskan program *JavaScript* pada file terpisah (meskipun anda bisa juga melakukannya). Ingat bahwa yang dimaksud dengan tag container adalah tag yang diawali dengan <NAMA_TAG> dan diakhiri dengan </NAMA_TAG>. Beberapa contoh tag container adalah <HTML></HTML>, <HEAD></Body>, dsb (Antony Pranata ; 2001 : 11).

II.5. MySQL

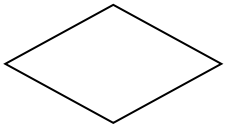
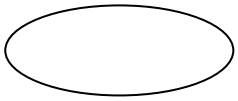
Database MySQL merupakan aplikasi yang bersifat *daemon* atau menetap dalam memori yang berjalan bersama dengan sistem operasi *Microsoft Windows*. *Interface* utama *MySQL database server* adalah *command line* atau berbasis *DOS* sehingga diperlukan pengetahuan khusus mengenai penggunaan perintah atau *commandi* dalam *command shell MySQL*. (Wahana Komputer, 2011).

Menurut Anisya (2013) *MySQL* (biasa dibaca dengan mai-es-ki-el atau bisa juga mai-se-kuel) adalah suatu perangkat lunak *database* relasi (*Relational Database Management System* atau *DBMS*), seperti halnya *ORACLE, POSTGRESQL, MSSQL,* dan lain sebagainya. *SQL* merupakan singkatan dari *Structure Query Language,* didefenisikan sebagai suatu sintaks perintah – perintah tertentu atau bahasa program yang digunakan untuk mengelola suatu *database*. Jadi *MySQL* adalah softwarenya dan *SQL* adalah bahasa perintahnya.

II.6. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antarentitas. Proses memungkinkan analis menghasilkan struktur basisdata yang baik sehingga data dapat disimpan dan diambil secara efisien (Janner Simarmata, 2010 : 67).

Tabel II.1. Simbol ERD

Notasi	Keterangan
	**Entitas,** adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	**Relasi,** menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda
	**Atribut,** berfungsi mendeskripsikan karakter entitas (atribut yang berfungsi sebagai key diberi garis bawah
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut

(Sumber : Janner Simarmata, 2010 : 67)

II.7. Teknik 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 rasional.

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 tekah ditemukan.

II.7.1.Bentuk-bentuk Normalisasi

1. **Bentuk normal tahap pertama (1" Normal Form)**

   Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

2. **Bentuk normal tahap kedua (2^{nd} normal form)**

   Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

3. **Bentuk normal tahap ketiga (3^{rd} normal form)**

   Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

4. Boyce Code Normal Form (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. Bentuk Normal Tahap Keempat dan Kelima

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata, 2010 : 76).

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

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

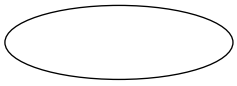
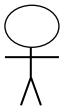
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use case* Diagram

   Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.2. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.

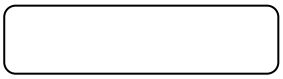
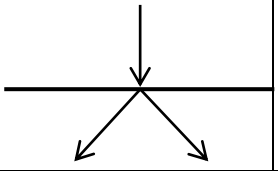
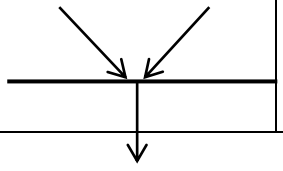
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

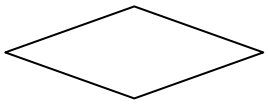
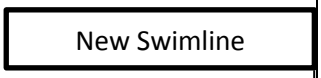
(Sumber : Windu Gata, 2013 : 4)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.3. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.

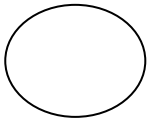
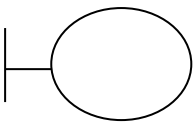
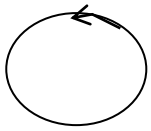
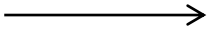
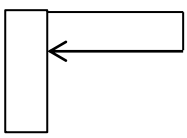
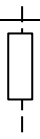
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata, 2013 : 6)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi

	operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata, 2013 : 7)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility,* tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata, 2013 : 9)