BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan

Sistem pendukung keputusan adalah sistem berbasis komputer yang membantu para pengambil keputusan mengatasi berbagai masalah melalui interaksi langsung dengan sejumlah *database* dan perangkat lunak analitik. Tujuan dari sistem adalah untuk menyimpan data dan mengubahnya ke informasi yang terorganisir yang dapat diakses dengan mudah, sehingga keputusan-keputusan yang diambil dapat dilakukan dengan cepat, akurat dan murah.

Sistem pendukung keputusan ini beroperasi dalam konteks sistem informasi global untuk melayani unit bisnis yang spesifik dalam suatu perusahaan. Sistem pendukung keputusan tidak terlepas dari sistem informasi global yang lebih komprehensif. Sistem pendukung keputusan yang berhasil harus mempercepat aliran informasi ke pengambil keputusan. Data yang disimpan harus berkesinambungan secara terjadwal dan dapat diakses dengan mudah (Dermawan Wibisono ; 2013 : 129).

II.1.1. Karakteristik Sistem Pendukung Keputusan

Karakteristik dan kapabilitas sistem pendukung keputusan antara lain :

1. Dukungan untuk pengambilan keputusan, terutama pada situasi semi terstruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan

informasi terkomputerisasi. Masalah-masalah tersebut tidak dapat dipecahkan oleh sistem komputer lain atau oleh metode atau alat kuantitatif standar.

2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok. Masalah yang kurang terstruktur sering memerlukan keterlibatan individu dari departemen dan tingkat organisasional yang berbeda atau bahkan dari organisasi lain. DSS mendukung im virtual melalui alat-alat Web kolaboratif.
4. Dukungan untuk keputusan independen dan atau sekuensial. Keputusan dapat dibuat satu kali, beberapa kali, atau berulang (dalam interval yang sama).
5. Dukungan disemua fase proses pengambilan keputusan : intelegensi, desain, pilihan, dan implementasi
6. Dukungan diberbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu. Pengambil keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara cepat, dan dapat mengadaptasi DSS untuk memenuhi perubahan tersebut. DSS bersifat fleksibel dan karena itu pengguna dapat menambahkan, menghapus, menggabungkan, mengubah, atau menyusun kembali elemen-elemen dasar, DSS juga fleksibel dalam hal dapat dimodifikasi untuk memecahkan masalah lain yang sejenis.
8. Peningkatan terhadap keefektifan pengambilan keputusan (akurasi, *timeliness*, kualitas).
9. Kontrol penuh oleh pengambil keputusan terhadap semua langkah proses pengambilan keputusan dalam memecahkan suatu masalah. DSS secara

khusus menekankan untuk mendukung pengambil keputusan, bukannya menggantikan.

II.1.2. Komponen Sistem Pendukung Keputusan

Sistem Pendukung Keputusan disusun dari beberapa subsistem yaitu :

1. Subsistem manajemen data

   Basis data yang relevan dan dikelola menggunakan *software* yang disebut *database* management system (DBMS)

2. Subsistem manajemen model

   Adalah paket *software* yang berisi model-model yang disebut dengan *modelbase management system* (MBMS)

3. Subsistem manajemen pengetahuan

   Subsistem yang memberikan intelegensi dan mendukung subsistem yang lain.

4. Subsistem antarmuka pengguna

   Pengguna berkomunikasi dan memerintah SPK melalui system ini.

5. Pengguna

   Orang yang berhadapan dengan pengambil keputusan (Sri Hartati ; 2011 : 37).

II.2. Nilai Tukar Kurs

Nilai tukar mata uang suatu Negara merupakan salah satu indicator penting dalam suatu perekonomian. Nilai tukar juga mempunyai implikasi yang luas, baik dalam konteks ekonomi domestic maupun internasional, mengingat hampir semua Negara di dunia melakukan transaksi internasional. Valuta asing

yang sering disebut dengan akronim valas pada dasarnya adalah mata uang asing (*foreign currencies*). Dalam pandangan awam semua valuta asing dapat digunakan sebagai alat pembayaran luar negeri. Namun ternyata hanya mata uang tertentu yang dapat digunakan sebagai mata uang untuk membayar transaksi internasional. Untuk itu sesuai dengan kewenangannya International Monetary Fund merekomendasikan beberapa mata uang untuk membayar transaksi internasional. (Adi Teguh Suprapto ; 2005 : 1).

II.3. *Hebb Neural Network*

Neural network pertama kali diperkenalkan oleh McCulloh dan Pitts pada tahun 1943. McCulloch dan Pitts menyimpulkan bahwa kombinasi beberapa neuron sederhana menjadi sebuah sistem neural akan meningkatkan kemampuan komputasinya. Bobot dalam jaringan yang diusulkan oleh McCulloch dan Pitts diatur untuk melakukan fungsi logika sederhana. Fungsi aktivasi yang dipakai adalah fungsi *threshold*.

Neural Network mencoba untuk meniru perilaku sel–sel otak manusia. Sel neuron menerima sinyal dari dendrit. Ketika neuron menerima sinyal, *neuron* dapat menyala. Ketika neuron menyala, sinyal ditransmisikan melalui akson. Pada akhirnya, sinyal akan meninggalkan *neuron* karena perjalanan ke akson terminal. Sinyal ditransmisikan ke neuron lain atau saraf. Sinyal yang ditrnasmisikan oleh *neuron* adalah sebuah sinyal analog. Sedangkan pada komputer modern adalah mesin digital yang memerlukan sinyal digital dalam memproses informasi. Sehingga digunakan digit digital 0 (mati) dan 1 (hidup).

Neural network biasanya diimplementasikan dengan menggunakan komponen elektronika atau disimulasikan dalam sebuah perangkat lunak pada komputer digital. Untuk mencapai tampilan yang baik, neural network memakai interkoneksi yang sangat besar antara sel-sel komputasi yang disebut "*neuron*" atau "unit pemroses". Sebagai mesin yang adaptif, sebuah *neural network* adalah sebuah prosessor besar terdistribusi yang paralel yang tersusun dari unit pemroses sederhana yang mempunyai kecenderungan untuk menyimpan pengalaman dan pengetahuan dan membuatnya siap untuk digunakan. Hal itu menyerupai otak dalam dua aspek:

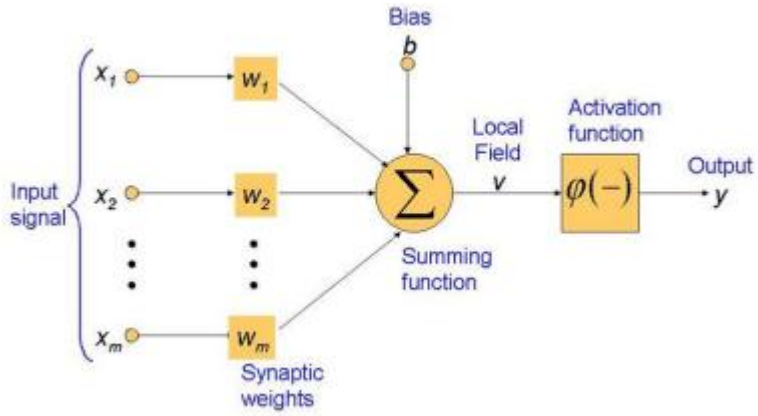
a. Pengetahuan dibutuhkan oleh jaringan dari lingkungannya melalui proses pembelajaran.
b. Kekuatan koneksi *interneuron*, dikenal sebagai bobot sinapsis, digunakan untuk menyimpan pengetahuan yang dibutuhkan.

Neural network adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan saraf biologi. Neural network telah dikembangkan sebagai generalisasai model matematika dari jaringan saraf biologi, berdasarkan asumsi:

a. Pemrosesan informasi terjadi pada neuron.
b. Sinyal dikirimkan antarneuron melalui penghubung dendrit dan akson.
c. Penghubung antarelemen memiliki bobot yang akan menambah dan mengurangi sinyal.

d. Untuk menentukan output, setiap neuron memiliki fungsi aktivasi yang dikenakan pada jumlah semua inputnya. Besar output akan dibandingkan dengan nilai threshold tertentu.

Secara matematis, proses ini dijelaskan dalam gambar II.1



Gambar II.1. Model matematis dari neural network (Achmad Udin Zailani ; 2011 : 3-4)

II.3.1. Arsitektur Jaringan

Baik tidaknya suatu model neural network salah satunya ditentukan oleh hubungan antarneuron atau yang biasa disebut sebagai arsitektur jaringan. Neuronneuron tersebut berkumpul dalam lapisan-lapisan yang disebut neuron layer. Lapisan-lapisan penyusun neural network dibagi menjadi tiga yaitu:

a. Input Layer adalah unit-unit dalam lapisan input disebut unit-unit input yang bertugas menerima pola inputan dari luar yang menggambarkan suatu permasalahan.

b. Hidden Layer adalah unit-unit dalam lapisan tersembunyi disebut unit-unit tersembunyi, yang mana nilai output-nya tidak dapat diamati secara langsung.

c. Output Layer adalah unit-unit dalam lapisan output disebut unit-unit output, yang merupakan solusi neural network terhadap suatu permasalahan.

Dalam menentukan jumlah layer, input tidak terhitung sebagai layer karena layer tersebut tidak melakukan proses komputasi. Atau bisa dikatakan bahwa jumlah layer pada jaringan ditentukan berdasarkan lapisan yang berisikan bobot antar koneksi dari kumpulan neuron-neuron. Hal inilah yang mendasari bahwa bobot pada neural network berisikan informasi yang sangat penting.

Adapun jenis arsitektur yang sering digunakan dalam neural network antara lain:

a. Jaringan lapisan tunggal

   Jaringan lapisan tunggal mempunyai satu lapisan bobot terkoneksi. Pada lapisan ini, unit input dapat dibedakan dengan unit output. Dimana unit input merupakan unit yang menerima sinyal dari dunia luar sedangkan unit output adalah unit dimana respon dari jaringan dapat terlihat.

b. Jaringan lapisan banyak

   Jaringan lapisan banyak adalah jaringan dengan satu atau lebih lapisan diantara lapisan input dan lapisan output yang biasa disebut lapisan tersembunyi (hidden layer).

c. Jaringan dengan lapisan kompetitif

Bentuk lapisan kompetitif merupakan neural network yang sangat besar. Interkoneksi antarneuron pada lapisan ini tidak ditunjukkan pada arsitektur seperti jaringan yang lain. (Achmad Udin Zailani ; 2011 : 3-4).

II.3.2. Fungsi Aktivasi

Fungsi aktivasi merupakan bagian penting dalam tahapan perhitungan output dari suatu algoritma. Beberapa fungsi aktivasi yang sering digunakan adalah sebagai berikut:

a. Fungsi threshold (batas ambang)

$$f(x) = \begin{cases} 1 & jika\ x \geq 0 \\ 0 & jika\ x < 0 \end{cases}$$

Untuk beberapa kasus, fungsi threshold yang dibuat tidak nilai 0 atau 1, tapi bernilai -1 atau 1 (sering disebut threshold bipoar). Jadi

$$f(x) = \begin{cases} 1 & jika\ x \geq 0 \\ -1 & jika\ x < 0 \end{cases}$$

b. Fungsi sigmoid

$$f(x) = \frac{1}{1 + e^{-x}}$$

Fungsi sigmoid sering dipakai karena nilai fungsinya yang terletak antara 0 dan 1 dan dapat diturunkan dengan mudah.

$$f'(x) = f(x)(1 - f(x))$$

c. Fungsi identitas

$$f(x) = x$$

Fungsi identitas sering dipakai apabila menginginkan output jaringan berupa sembarangan bilangan riil (bukan hanya pada range [0,1] atau [-1,1] (Achmad Udin Zailani ; 2011 : 5-6).

II.4. Pengertian Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai jenis komputer dan berbagai sistem operasi termasuk telepon genggam. Java dikembangkan oleh *Sun Microsystem* dan dirilis tahun 1995. Java merupakan suatu teknologi perangkat lunak yang digolongkan *multi platform*. Selain itu, Java juga merupakan suatu *platform* yang memiliki *virtual machine* dan *library* yang diperlukan untuk menulis dan menjalankan suatu program.

Bahasa pemrograman java pertama lahir dari *The Green Project*, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan *versi* yang dinamakan Oak. Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy, serta Sembilan pemrograman lainnya dari *Sun Microsystem.* Salah satu hasil proyek ini adalah mascot Duke yang dibuat oleh Joe Palrang (Wahana Komputer ; 2010 : 1).

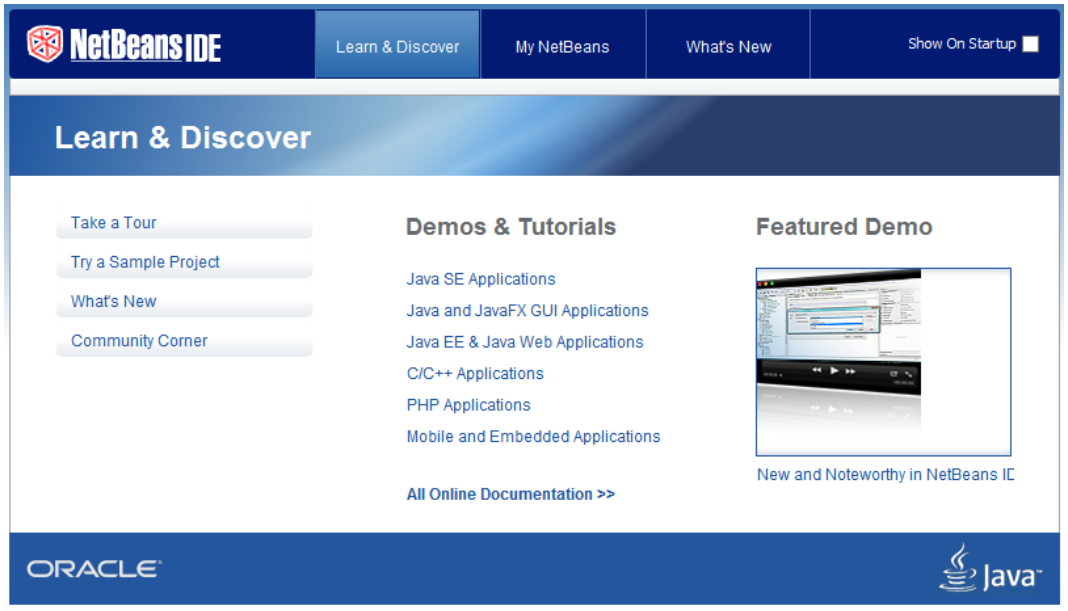
II.5. Pengertian NetBeans

NetBeans merupakan salah satu proyek *open source* yang disponsori oleh *Sun Microsystem*. Proyek ini bediri pada tahun 2000 dan telah menghasilkan 2 produk, yaitu NetBeanss IDE dan NetBeans Platform. NetBeans IDE merupakan

produk yang digunakan untuk melakukan pemrograman baik menulis kode, meng-*compile*, mencari kesalahan dan mendistribusikan program. Sedangkan NetBeans Platform adalah sebuah modul yang merupakan kerangka awal / pondasi dalam bangun aplikasi desktop yang besar.

NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi desktop), pemrograman enterprise, dan pemrograman perangkat mobile. Saat ini NetBeans telah mencapai versi 6.8 (Wahana Komputer ; 2010 : 15).

Tampilan awal Netbeans 7.3 terlihat seperti gambar II.1 di bawah ini:



Gambar II.1. Tampilan Awal Netbeans 7.3
Sumber: www.netbeans.org

II.6. Pengertian Database

D*atabase* adalah sekumpulan *file* data yang saling berhubungan dan diorganisasi sedemikian rupa sehingga memudahkan untuk mendapat dan memproses data. Lingkungan sistem *database* menekankan data yang tidak

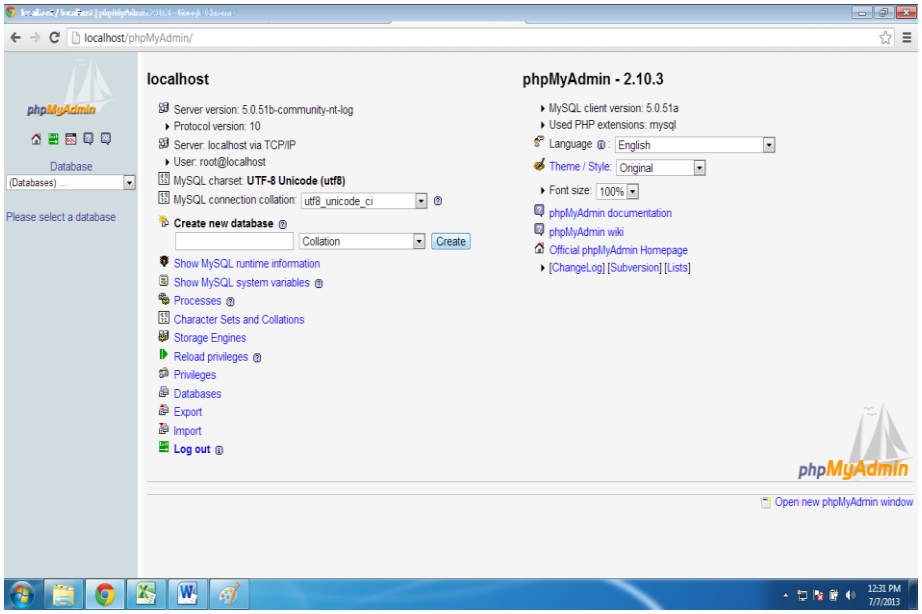
tergantung (*idenpendent data*) pada aplikasi yang akan menggunakan data. Data adalah kumpulan fakta dasar (mentah) yang terpisah.

Sebuah *database* harus dibuat dengan rapi agar data yang dimasukkan sesuai dengan tempatnya. Sebagai contoh, di sebuah perpustakaan, penyimpanan buku dikelompokkan berdasar jenis atau kategori-kategori tertentu, misalnya kategori buku komputer, buku pertanian, dan lain-lain. Kemudian dikelompokkan lagi berdasarkan abjad judul buku, ini dilakukan agar setiap pengunjung dapat dengan mudah mencari dan mendapatkan buku yang dimaksud (Wahana Komputer ; 2006 ; 1).

II.7. Pengertian MySQL

Mysql pertama kali dirintis oleh seorang programmer database bernama Michael Widenius, yang dapat anda hubungi di emailnya monty@analytikerna. Mysql *database server* adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. meskipun begitu, tidak menuntut *resource* yang besar. Mysql adalah *database* yang paling populer di antara *database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multi user*. MySQL memiliki dua bentuk lisensi, yaitu *free software* dan *shareware*. penulis sendiri dalam menjelaskan buku ini menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*general public license*) (Wahana Komputer; 2010 : 5).

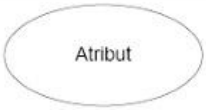


Gambar II.2. Tampilan MySQL
(Sumber : Wahana Komputer ; 2010 : 5)

II.8. *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 yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

(Sumber : Janner Simarmata ; 2010 : 67)

II.9. Kamus Data

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

II.10. 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.10.1.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 (2nd 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 (3rd 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.11. 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.

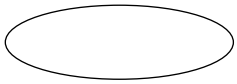
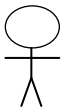
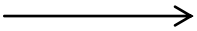
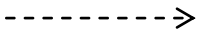
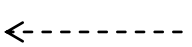
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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*

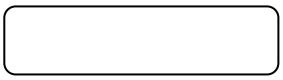
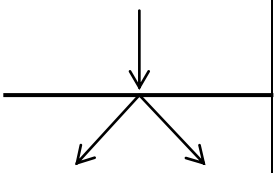
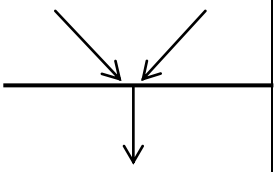
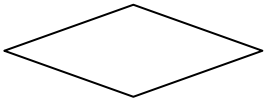
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi 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*

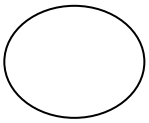
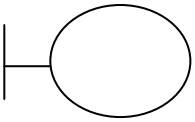
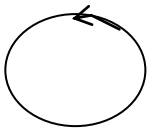
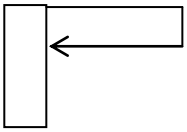
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan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