BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pakar

II.1.1. Pengertian Sistem Pakar

Sistem Pakar (*Expert System*) adalah sistem yang menggunakan pengetahuan manusia, dimana pengetahuan tersebut dimasukkan ke dalam sebuah komputer, dan kemudian digunakan untuk menyelesaikan masalah-masalah yang biasanya membutuhkan kepakaran atau keahlian manusia. (Muhammad Dahria, dkk; 2013: 1).

II.1.2. Manfaat Sistem Pakar

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

1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
2. Bisa melakukan proses secara berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian para pakar.
4. Meningkatkan output dan produktivitas.
5. Meningkatkan kualitas.
6. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
7. Mampu beroperasi dalam lingkungan yang berbahaya.
8. Memiliki kemampuan untuk mengakses pengetahuan.

9. Memiliki Reabilitas
10. Meningkatkan kapabilitas sistem komputer.
11. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidak pastian.
12. Sebagai media pelengkap dalam pelatihan.
13. Meningkatkan kapabilitas dalam penyelesaian masalahMenghemat waktu dalam pengambilan keputusan (Wahyudi, Jumadi, No. 1 Februari 2011).

II.1.3. Kelebihan Sistem Pakar

Sistem Pakar memiliki beberapa fitur menarik yang merupakan kelebihannya, seperti :

1. Meningkatkan ketersedian (*increased availability*). Kepakaran atau keahlian menjadi tersedia dalam sistem komputer. Dapat dikatakan bahwa sistem pakar merupakan produksi kepakaran secara masal (*massproduction*).
2. Mengurangi biaya *(reduced cost).* Biaya yang diperlukan untuk menyediakan keahlian per satu orang *user* menjadi berkurang.
3. Mengurangi Bahaya *(reduced danger).* Sistem pakar dapat digunakan di lingkungan yang mungkin berbahaya bagi manusia.
4. Permanen *(permanence).* Sistem pakar dan pengetahuan yang terdapat di dalamnya bersifat lebih permanen dibandingkan manusia yang dapat

merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.

5. Keahlian Multipel *(multiple expertise).* Pengetahuan dari beberapa pakar dapat dimuat ke dalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian atau pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar (Rosnelly;2011;5).

II.1.4. Elemen Manusia Pada Sistem Pakar

Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4, yaitu :

1. **Pakar**

   Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu. Pakar juga memiliki kemampuan untuk mengaplikasikan pengetahuannya dan memberikan saran serta pemecahan masalah pada domain tertentu.

   Seorang pakar memiliki kemampuan kepakaran, yaitu :

   a. Dapat mengenali dan merumuskan suatu masalah.
   b. Menyelasaikan masalah dengan cepat dan tepat.
   c. Menjelaskan solusi dari suatu masalah.
   d. Restrukturasi pengetahuan.
   e. Belajar dari pengalaman.

f. Memahami batas kemampuan.

Selain itu, pakar juga memiliki kemampuan untuk mengaplikasikan pengetahuannya dan memberikan saran serta pemecahan masalah pada domain tertentu. Ini merupakan pekerjaan pakar, memberikan pengetahuan tentang bagaimana seseorang melaksanakan tugas untuk menyelesaikan masalah. Penyelesaian masalah ini didukung atau bahkan secara ekstrim akan dilaksanakan oleh sistem berbasis pengetahuan (sistem pakar).

Seorang pakar mengetahui fakta – fakta mana yang penting, sebab akibat, fenomena – fenomena yang terkait dengan fakta, memahami arti hubungan antar fakta, juga hubungan sebab akibat, dan hubungan dengan fenomena – fenomena yang terkait serta mampu menginterprestasikan akibat – akibat yang terjadi karena sesuatu sebab terjadi (Rosnelly; 2011: 10).

2. Pembangun / Pembuat Pengetahuan

Pembangun pengetahuan memiliki tugas utama menerjemahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan masalah maupun sumber terdukumentasi lainnya kedalam bentuk yang bisa diterima oleh sistem pakar. Dalam hal ini pembangun pengetahuan *(knowledge engineer)* menginterpretasikan dan merepresentasikan pengetahuan yang diperoleh dalam bentuk jawaban-jawaban atas pertanyaan-pertanyaan yang diajukan pada pakar atau pemahaman, penggambaran, analogis,

sistematis, konseptual yang diperoleh dari membaca beberapa dokumen cetak seperti *text book,* jurnal, makalah, dan sebagainya. Kurangnya pengalaman *Knowledge engineer* merupakan kesulitan utama dalam mengkonstruksi sistem pakar. Untuk mengatasi hal tersebut, perancang sistem pakar menggunakan *tools* komersial. (Seperti pada editor-editor khusus maupun *logic debuggers*) dan usahanya akan dipusatkan pada pembangunan mesin inferensi (Rosnelly; 2011: 11).

3. **Pembangun / Pembuat Sistem**

   Pembangun sistem adalah orang yang bertugas untuk merancang antarmuka pemakai sistem pakar, merancang pengetahuan yang sudah diterjemahkan oleh pembangun pengetahuan kedalam bentuk yang sesuai dan dapat diterima oleh sistem pakar dan mengimplementasikannya kedalam mesin infrensi. Selain hal tersebut, pembangun sistem juga bertanggung jawab apabila sistem pakar akan diinteregasikan dengan sistem komputerasi lain. Alat pembangun (*tool builder)* dapat dipakai untuk menyajikan atau membangun *tool* yang spesifik. Penjual (*vendor)* dapat memberikan *tool* dan saran, staf pendukung dapat memberikan saran dan bantuan secara teknis dalam proses pembangunan sistem pakar (Rosnelly; 2011: 11).

4. **Pengguna *(User)***

   Banyak sistem berbasis komputer mempunyai susunan pengguna tunggal. Hal ini berbeda jauh dengan sistem pakar yang memungkinkan mempunyai beberapa kelas pengguna. Table II.1 menunjukkan

beberapa contoh hubungan antara kelas pengguna, kepentingan pengguna, dan fungsi sistem pakar (Rosnelly; 2011: 12).

Table. II.1 Hubungan antara pengguna dan fungsi sistem pakar

Pengguna	**Kepentingan**	**Fungsi sistem pakar**
Klien bukan pakar	Mencari saran/nasehat	Konsultan atau penasehat
Mahasiswa	Belajar	Instruktur
Pembangun sistem	Memperbaiki/menambah basis pengetahuan	Rekan (partner)
Pakar	Membantu analisi rutin atau proses komputasi, mecari (mengklasifikasi) informasi, alat bantu diagnose	Rekan kerja atau asisten

Sumber (Rosnelly;2011,12)

II.2. Sony Z3

Sebagai ponsel super terbaru dari Sony, tidak ada perbedaan yang cukup jauh dengan ponsel sebelumnya yang diluncurkan. Namun terdapat beberapa peningkatan yang setidaknya mampu memberikan nilai jual tersendiri. Dari segi ukurannya,Sony Xperia Z3memiliki dimensi 146 x 72 x 7.3 mm dan bobot163 gr. Desain yang lebih tipis membuat Anda sebagaipengguna lebih mudah dalam menggenggam. Untuk ukuran layarnya masih sama dengan sebelumnya, yakni 5.2 inch dengan resolusi Full HD 1080 x 1920 piksel. Dari jeroannya, terdapat peningkatan juga jika dibandingkan dengan sebelumnya. Dengan menggunakan chip processor Qualcomm MSM89734AC Snapdragon 801 Quad-core 2.5 GHz Krait 400 yang didampingi oleh pengolah grafis GPU Adreno 330 dan RAM sebesar 3 GB. Untuk sistem operasinya, secara default sudah terinstall OS

Android KitKat 4.4.4. Dan juga telah diumumkan resmi bahwa untuk seri Z mendapatkan update ke OS Android Lollipop 5.0. Kamera yang digunakan tentunya menggunakan merk Sony sendiri yang sudah diakui dari segi kualitasnya. Kamera belakang yang memiliki resolusi 20.7 MP autofocus dan dilengkapi LED flash mempunyai bukaan lensasebesar 1/2.3 yang membuat cahaya masuk lebih banyak dan bisa memotret objek secara makro lebih baik. Mampu merekam video hingga resolusi 4K 2160p@30fps, Full HD 1080p@60fps, hingga slow-motion 720p@120p bisa ditangani dengan baik. Sedangkan untuk kamera depannya, memiliki resolusi 2.2 MP yang bisa merekam video dengan resolusi Full HD 1080p@30fps. Fasilitas lain yang didapat dari ponsel Android canggih ini adalah memiliki jaringan sinyal yang luas dari GPRS hingga 4G LTE Cat4. Sedangkan fitur lain yang bisa didapatkan yaitu Wi-Fi, Wi-Fi Hotspot, A-GPS with GLONASS, Beidou, FM Radio with RDS, NFC, microUSB v2.0 support USB OTG, Bluetooth v4.0, dll. Memori internal yang disediakan sebesar 16 GB bisa Anda tambahkan hingga 128 GB dengan menggunakan fasilitas slot microSD card. Dan untuk kapasitas baterai, 3100mAh merupakan kapasitas yang besar dan mencukupi untuk menemani Anda berkegiatan sehari-hari (detikponsel.com).

Tetapi semakin canggih sebuah smartphone juga akan semakin banyak mengalami masalah, entah itu permasalahan dari android os atau dari smartphone itu sendiri, seperti halnya seperti sony xperia z3 tidak luput dari masalah.

II.3. Metode Dempster Shaffer

Ada berbagai macam penalaran dengan model yang lengkap dan sangat konsisten, tetapi pada kenyataannya banyak permasalahan yang tidak dapat terselesaikan secara lengkap dan konsisten. Ketidak konsistennya yang tersebut adalah akibat adanya penambahan fakta baru. Penalaran yang seperti itu disebut dengan penalaran *non monotonis*. Untuk mengatasi ketidakkonsistenan tersebut maka dapat menggunakan penalaran dengan teori *Dempster Shafer*.*Dempster Shafer* adalah suatu teori matematika untuk pembuktian berdasarkan *belief function and plausible reasoning* (fungsi kepercayaan dan pemikiran yang masuk akal), yang digunakan untuk mengkombinasikan potongan informasi yang terpisah (bukti) untuk mengkalkulasi kemungkinan dari suatu peristiwa. Teori ini dikembangkan oleh Arthur P. Dempster Dan Glenn Shafer.

Secara umum teori *Dempster Shafer* ditulis dalam suatu *interval Belief* dan *Plausibility*.

a. *Belief (Bel)* adalah ukuran kekuatan *evidence* dalam mendukung suatu himpunan proposisi. Jika bernilai 0 maka mengindikasikan bahwa tidak ada *evidence*, dan jika bernilai 1 menunjukkan adanya kepastian.
b. Plausibility (Pl) dinotasikan sebagai :

   Pl(s) = 1 – Bel (¬s)

   Plausibility juga bernilai 0 sampai 1. Jika yakin akan ¬s, maka dapat dikatakan bahwa Bel(¬s)=1, dan Pl(¬s)=0. Pada teori *Dempster Shafer* dikenal adanya *frame of discrement* yang dinotasikan dengan θ. *Frame* ini merupakan semesta pembicaraan dari sekumpulan hipotesis.

Tujuannya adalah mengaitkan ukuran kepercayaan elemen-elemen θ. Tidak semua *evidence* secara langsung mendukung tiap-tiap elemen. Untuk itu perlu adanya probabilitas fungsi densitas (m). Nilai m tidak hanya mendefinisikan elemen-elemen θ saja, namun juga semua subsetnya. Sehingga jika θ berisi n elemen, maka subset θ adalah 2n . Jumlah semua m dalam subset θ sama dengan 1.

Apabila tidak ada informasi apapun untuk memilih hipotesis, maka nilai : m{θ} = 1,0.Apabila diketahui X adalah subset dari θ, dengan m1 sebagai fungsi densitasnya, dan Y juga merupakan subset dari θ dengan m2 sebagai fungsi densitasnya, maka dapat dibentuk fungsi kombinasi m1 dan m2 sebagai m3, yaitu:

$$m3(Z) = \frac{\Sigma X \cap Y = Z \; m1(X).\, m2(Y).}{1 - \Sigma X \cap Y = \emptyset \; m1(X).\, m2(Y)}$$

(Muhammad Dahria, dkk; 2013: 1).

II.4. Basis Data

Basis data adalah kumpulan data (arsip, atau file) yang saling berhubungan yang disimpan dalam media penyimpanan elektronis agar dapat dimanfaatkan kembali dangan cepat, dan mudah. Sedangkan sistem basis data adalah kumpulan file, atau tabel yang saling berhubungan yang memungkinkan beberapa pemakai, atau program lain untuk mengakses, dan memanipulasi file-file (tabel) tersebut (Eka Kurniawan, 2015).

II.5. Kamus Data

Kamus data (*data dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. 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 aplikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond ; 2010 : 171).

II.6. Normalisasi

Normalisasi diartikan sebagai suatu teknik yang menstrukurkan/ mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan (Sutanta ; 2011 : 174).

Proses normalisasi menghasilkan relasi yang optimal, yaitu :

1. Memiliki struktur *record* yang konsisten secara logik;
2. Memiliki struktur *record* yang mudah untuk dimengerti;
3. Memiliki struktur *record* yang sederhana dalam pemeliharaan;
4. Memiliki struktur *record* yang mudah ditampilkan kembali untuk memenuhi kebutuhan pengguna;
5. Minimalisasi kerangkapan data guna meningkatkan kinerja sistem.

Secara berturut-turut masing-masing level normal tersebut dibahas berikut ini, dimulai dari bentuk tidak normal.

1. Relasi bentuk tidak normal (*Un Normalized Form* / UNF)

   Relasi-relasi yang dirancang tanpa mengindahkan batasan dalam defisi basis data dan karakteristik *Relational Database Management System* (RDBM) menghasilkan relasi *Un Normalized Form* (UNF). Bentuk ini harus di hindari dalam perancangan relasi dalam basis data. Relasi *Un Normalized Form* (UNF) mempunyai kriteria sebagai berikut.

   a. Jika relasi mempunyai bentuk *non flat file* (dapat terjadi akibat data disimpan sesuai dengan kedatangannya, tidak memiliki struktur tertentu, terjadi duplikasi atau tidak lengkap).
   b. Jika relasi membuat *set atribut* berulang *(non single values).*
   c. Jika relasi membuat *atribut non atomic value.*

2. Relasi bentuk normal pertama (*First Norm Form* / 1NF)

   Relasi disebut juga *First Norm Form* (1NF) jika memenuhi kriteria sebagai berikut :

   a. Jika seluruh atribut dalam relasi bernilai *atomic (atomic value)*
   b. Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*
   c. Jika relasi tidak memuat set atribut berulang
   d. Jika semua record mempunyai sejumlah atribut yang sama.

   Permasalahan dalam *First Norm Form* (1NF) adalah sebagai berikut.

   a. Tidak dapat menyisipkan informasi parsial.

b. Terhapusnya informasi ketika menghapus sebuah *record.*

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

   Relasi disebut sebagai *Second Normal Form (*2NF) jika memenuhi kriteria sebagai berikut :

   a. Jika memenuhi kriteria *First Norm Form (*1NF).
   b. Jika semua atribut nonkunci *Functional Dependence* (FD) pada *Primary Key* (PK).

   Permasalahan dalam *Second Normal Form /* 2NF adalah sebagai berikut:

   a. Kerangkapan data *(data redundancy).*
   b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data *(data inconsistency).*
   c. Proses pembaharuan data tidak efisien.
   d. Penyimpangan pada saat penyisipan, penghapusan, dan pembaruan.

   Kriteria tersebut mengidentifikasikan bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Norm Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Norm Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

   a. Identifikasikan *Functional Dependence* (FD) relasi *First Norm Form* (1NF).

b. Berdasarkan informasi tersebut, dekomposisi relasi *First Norm Form* (1NF) menjadi relasi-relasi baru sesuai *Functional Dependence* nya. Jika menggunakan diagram maka simpul-simpul yang berada pada puncak diagram ketergantungan data bertindak *Primary Key* (PK) pada relasi baru.

4. Bentuk normal ketiga (*Third Norm Form* / 3NF)

   Suatu relasi disebut sebagai *Third Norm Form* jika memenuhi kriteria sebagai berikut :

   a. Jika memenuhi kriteria *Second Normal Form* (2NF).
   b. Jika setiap atribut nonkunci tidak (TDF) *(Non Transitive Dependeny)* terhadap *Primary Key* (PK).

   Permasalahan dalam *Third Norm Form* (3NF) adalah keberadaan penentu yang tidak merupakan bagian dari *Primary Key* (PK) menghasilkan duplikasi rinci data pada atribut yang berfungi sebagai *Foreign Key* (FK) (duplikasi berbeda dengan keterangan data).

   Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Norm Form* (3NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

   a. Identifikasi TDF relasi *Second Normal Form* (2NF).
   b. Berdasarkan informasi tersebut, dekomposisi relasi *Second Normal Form* (2NF) menjadi relasi-relasi baru sesuai TDF-nya.

5. Bentuk normal *Boyce-Cood (Boyce-Codd Norm Form* / BCNF)

Bentuk normal *Boyce-Codd Norm Form* (BCNF) dikemukakan oleh R.F. Boyce dan E.F. Codd. Suatu relasi disebut sebagai *Boyce-Codd Norm Form* (BCNF) jika memenuhi kriteria sebagai berikut :

a. Jika memenuhi kriteria *Third Norm Form* (3NF).
b. Jika semua atribut penentu (determinan) merupakan CK.

6. Bentuk normal keempat (*Forth Norm Form* / 4NF)

   Relasi disebut sebagi *Forth Norm Form* (4NF) jika memenuhi kriteria sebagai berikut :

   a. Jika memenuhi kriteria *Boyce-Codd Norm Form*.
   b. Jika setiap atribut didalamnya tidak mengalami ketergantungan pada banyak nilai.

7. Bentuk normal kelima (*Fifth Norm Form* / 5NF)

   Suatu relasi memenuhi kriteria *Fifth Norm Form* (5NF) jika kerelasian antar data dalam relasi tersebut tidak dapat direkonstruksi dari struktur relasi yang sederhana.

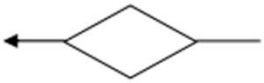
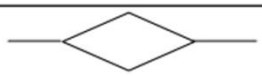
8. Bentuk normal kunci domain (*Domain Key Norm Form* / DKNF)

   Suatu relasi disebut sebagai *Domain Key Norm Form* (DKNF) jika setiap batasan dapat disimpulkan secara sederhana dengan mengetahui sekumpulan nama atribut dan domainnya selama menggunkan sekumpulan atribut pada kuncinya.

II.7 ERD (Entity Relationship Diagram)

Entity Relationship Diagram (ERD) adalah sekumpulan cara atau peralatan untuk mendeskripsikan data-data atau objek-objek yang dibuat berdasarkan dan berasal dari dunia nyata yang disebut entitas (entity) serta hubungan (relationship) antar entitas-entitas tersebut dengan menggunakan beberapa notasi (Edi, Betshani Vol.5).

Komponen-komponen pembentuk ERD dapat di lihat pada gambar di bawah ini.

Notasi	Komponen	Keterangan
	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lain.
	Atribut	Properti yang dimiliki oleh suatu entitas, dimana dapat mendeskripsikan karakteristik dari entitas tersebut.
	Relasi	Menunjukkan hubungan diantara sejumlah entitas yang berbeda.
	Relasi 1 : 1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua
	Relasi 1 : N	Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak entitas pada himpunan entias yang lain
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas pada himpunan entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya

Gambar II.1 Diagram UML
Sumber : (Edi, Betshani, 2011)

II.8. Microsoft *SQL Server*

SQL (*Structured Query Language*) adalah sebuah bahasa yang dipergunakan untuk mengakses data dalam basis data relasional. Bahasa ini secara de facto merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya. SQL terdiri dari dua bahasa, yaitu Data Definition Language (DDL) dan Data Manipulation Language (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis data (SMBD), namun secara umum implementasi setiap bahasa ini memiliki bentuk standar yang ditetapkan oleh ANSI (Adelia,Setiawan, No.2, 2011).

II.9. Bahasa Pemrograman *VB.NET*

Bahasa Pemrograman *VB.NET Microsoft* Visual Basic (sering disingkat sebagai VB saja) merupakan sebuah bahasa pemrograman yang bersifat event driven dan menawarkan *Integrated Development Environment (IDE) visual* untuk membuat program aplikasi berbasis sistem operasi *Microsoft Windows* dengan menggunakan model pemrograman *Common Object Model (COM). Visual Basic* merupakan turunan bahasa *basic* dan menawarkan pengembangan aplikasi komputer berbasis grafik dengan cepat, akses ke basis data menggunakan *Data Access Objects (DAO), Remote Data Objects (RDO),* atau *ActiveX Data Object (ADO),* serta menawarkan pembuatan kontrol *ActiveX* dan objek *ActiveX*.

Visual Basic merupakan turunan bahasa *basic* dan menawarkan pengembangan aplikasi komputer berbasis grafik dengan cepat, akses ke basis data menggunakan *Data Access Objects (DAO), Remote Data Objects (RDO),* atau *ActiveX Data Object (ADO),* serta menawarkan pembuatan kontrol *ActiveX* dan objek *ActiveX*. Beberapa bahasa skrip seperti *Visual Basic for Applications (VBA)* dan *Visual Basic Scripting Edition (VBScript),* mirip seperti halnya Visual Basic, tetapi cara kerjanya yang berbeda. Para *programmer* dapat membangun aplikasi dengan menggunakan komponen-komponen yang disediakan oleh *Microsoft Visual Basic* Program-program yang ditulis dengan *Visual Basic* juga dapat menggunakan *Windows API,* tapi membutuhkan deklarasi fungsi eksternal tambahan (Adelia, Setiawan, No.2, 2011).

II.10. UML (Unified Modelling Language)

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.

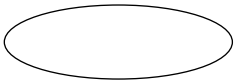
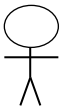
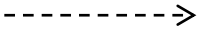
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

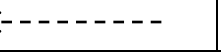
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

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.

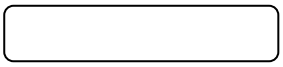
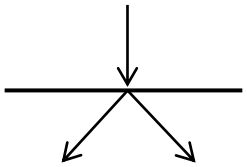
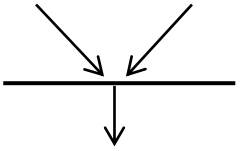
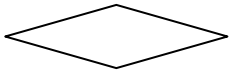
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(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*

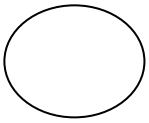
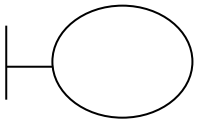
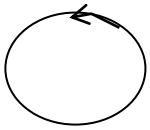
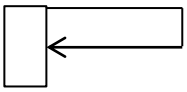
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan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