BAB II
TINJAUAN PUSTAKA

A. Kecerdasan Buatan

Kecerdasan buatan (*Artificial Intelligence*) adalah ide-ide untuk membuat suatu perangkat lunak komputer yang memiliki kecerdasan sehingga perangkat lunak komputer tersebut dapat melakukan suatu pekerjaan yang dilakukan oleh manusia (Artanti, 2004), dengan kata lain membuat sebuah komputer dapat berpikir dan bernalar seperti manusia. Tujuan dari kecerdsan buatan adalah membuat komputer lebih cerdas, mengerti tentang kecerdasan dan membuat mesin lebih berguna bagi manusia. Kecerdasan buatan dapat membantu meringankan beban kerja manusia misalnya dalam membuat keputusan, mencari informasi secara lebih akurat atau membuat komputer lebih mudah digunakan dengan tampilan yang lebih mudah dipahami. Cara kerja kecerdasan buatan adalah menerima input, untuk kemudian diproses dan kemudian mengeluarkan output yang berupa keputusan.

Menurut John McCarthy 1956, sistem pakar adalah proses untuk mengetahui dan memodelkan proses-proses berpikir manusia dan mendesain mesin agar dapat menirukan perilaku manusia. Kecerdasan atau kepandaian ini di dapat berdasarkan pengetahuan dan pengalaman, untuk itu agar perangkat lunak yang dikembangkan dapat mempunyai kecerdasan maka perangkat lunak tersebut harus diberi suatu pengetahuan dan kemampuan untuk menalar dari pengetahuan yang telah didapat dalam menemukan solusi atau kesimpuan layaknya seorang pakar dalam bidang tertentu yang spesifik.

Kecerdasan buatan menawarkan media dan uji teori kecedasan. Teori ini dapat dinyatakan dalam bahasa program komputer dan dibutikan melalui eksekusinya pada komputer nyata. Implementasi dari kecerdasan buatan saat ini dapat ditemui adalam-daam bidang antara lain:

1. *Fuzzy logic* merupakan suatu metode kecerdasan buatan yan banyak terdapat pada alat elektronik dan robot dimana ala-alat elektronik dan robot terebut mampu berpikir dan bertingkah laku seperti manusia.
2. *Komputer vision* merupakan suatu metode kecerdasan buatan yang memungkinkan sebuah sistem komputer mengenali gamar sebagai inputnya. Contohnya adalah mengenali dan membaca tulisan yang ada gambarnya.
3. *Artificial intelligence* dalam game merupakan metode kecerdasan buatan yang berguna untuk meniru cara berpikir manusia dalam bermain game.
4. *Speech recognition* merupakan suatu metode kecerdasan buatan yang berguna untuk mengenali suara manusia dengan cara dicocokkan dengan acuhan aau pattern yang telah diprogram sebelumnya. Contohnya adalah suara dari user dapat diterjemahkan menjadi sebuah perintah bagi komputer.
5. *Expert system* merupakan metode kecerdasan buatan yang berguna untuk meniru cara berpikir dan penalaran seorang ahli dalam mengambil keputusan berdasarkan situasi yang ada.

B. Sistem Pakar

1. Pakar dan sistem pakar

Sistem pakar adalah sistem informasi berbasis komputer yang menggunakan pengetahuan pakar untuk mencapai performa keputusan tingkat tinggi dalam domain persoalan yang sempit.

Sedangkan Pakar adalah orang yang memiliki pengetahuan, penilaian, pengalaman dan metode khusus, serta kemampuan untuk menerapkan bakat ini dalam memberi nasihat dan memecahkan masalah.Adalah tugas pakar untuk menyediakan pengetahuan tentang bagaimana melaksanakan suatu tugas yang akan dijalankan oleh sistem berbasis-pengetahuan.Pakar mengetahui fakta mana yang penting dan memahami arti hubungan diantaranya. Misalnya dalam mendiagnosis persoalan sistem listrik mobil, pakar mekanik mengetahui bahwa pengikat kipas dapat diputus dan menyebabkan baterai *discharge.*

Keahlian adalah pengetahuan ekstensif yang spesifik terhadap tugas yang dimiliki pakar.Tingkat keahlian menentukan performa keputusan.Keahlian sering dicapai dari pelatihan,membaca,dan mempraktikkan.Keahlian mencakup pengetahuan eksplisit,misalnya teori yang dipelajari dari buku teks atau kelas,dan pengetahuan implisit yang diperoleh dari pengalaman. (Sumber:Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,714-715).

2. Perbedaan Pakar dan Sistem Pakar

Menurut (Efraim Turban, Jay E. Aronson, Ting PengLiang, 2005,7 18) Perbedaan pakar dan sistem pakar dapat dilihat dari berbagai faktor, antara lain:

Tabel 2.1 Perbedaan Pakar Manusia dan Sistem Pakar

Fitur	Pakar manusia	Sistem pakar
Mortalitas	Ya	Tidak
Transfer pengetahuan	Sulit	Mudah
Dokumentasi pengetahuan	Sulit	Mudah
Konsistensi keputusan	Rendah	Tinggi
Unit biaya penggunaan	Tinggi	Rendah
Kreativitas	Tinggi	Rendah
Adaptabilitas	Tinggi	Rendah
Lingkup pengetahuan	Luas	Sempit
Tipe pengetahuan	Umum dan teknis	Teknis
Isi pengetahuan	Pengalaman	Simbol

3. Manfaat dan kemampuan sistem pakar.

Menurut (Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,730-732) ada beberapa manfaat serta kemampuan sistem pakar,antara lain :

a. Meningkatkan *output* dan produktivitas.
b. Menurunkan waktu pengambila keputusan.
c. Meningkatkan kualitas proses dan produk.
d. Mengurangi *downtime*.
e. Menyerap keahlian langka.
f. Fleksibilitas.

g. Operasi peralatan yang lebih mudah.

h. Eliminasi kebutuhan peralatan yang mahal.

i. Operasi di lingkungan berbahaya.

j. Aksesibilitas ke pengetahuan *help desk.*

k. Kemampuan untuk bekerja dengan informasi yang tidak lengkap atau tidak pasti.

l. Kelengkapan Pelatihan.

m. Peningkatan pemecahan masalah dan pengambilan keputusan.

n. Meningkatkan proses pengambilan keputusan.

o. Meningkatkan kualitas keputusan.

p. Kemampuan untuk memecahan persoalan kompleks.

q. Transfer pengetauan ke lokasi terpencil.

r. Peningkatan sistem informasi lain.

4. Keterbatasan sistem pakar.

Metodologi *ES* yang tersedia mungkin tidak langsung dan efektif,bahkan untuk banyak aplikasi dalam kategori umum.Persoalan-persoalan berikut telah memperlambat penyebaran komersial *ES* :

a. Pengetahuan tidak selalu siap sedia.

b. Akan sulit mengekstrak keahlian dari manusia.

c. Pendekatan tiap pakar pada suatu penilaian situasi mungkin berbeda tetapi benar.

d. Sulit, bahkan bagi pakar kemampuan tinggi, untuk mengikhtisarkan penilaian situasi yang baik pada saat berada dalam tekanan waktu.

e. Pengguna siste pakar memiliki batasan kognitif alami.

f. ES bekerja dengan baik hanya dalam domain pengetahuan sempit.

g. Kebanyakan pakar tidak mempunyai sarana mandiri untuk memeriksa apakah kesimpulannya masuk akal.

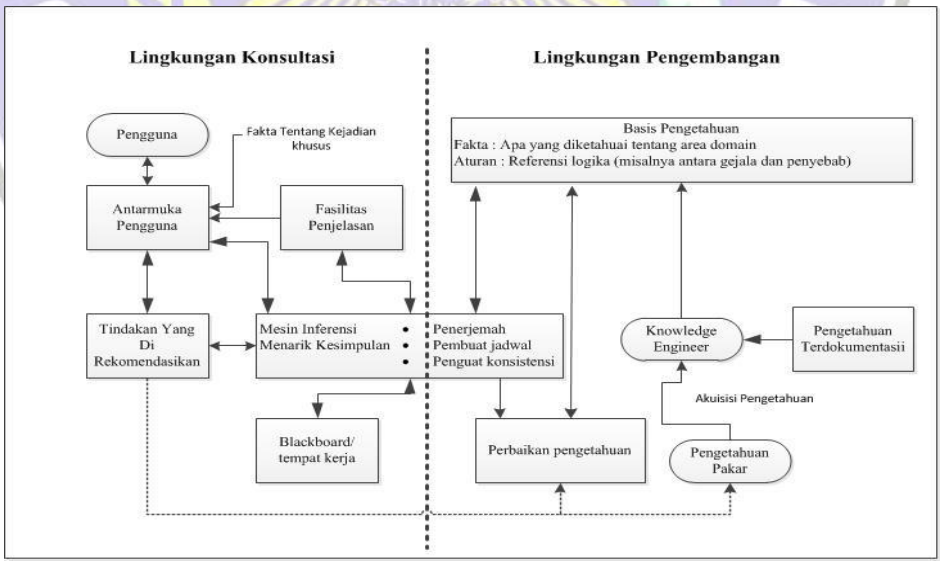
h. Kosakata atau jargon yang digunakan pakar unuk menyatakan fakta dan hubungan seirigkali terbatas dan tidak dipahami pakar lain.

i. Acapkali dibutuhkan *knowledge engineer*-yang langka dan mahal-suatu fakta yang menjadikan konstruksi *ES* mahal.

j. Kurangnya kepercayaan pada bagian pengguna akhir menjadi penghalang penggunaan *ES*.

k. Transfer pengetahuan adalah subjek terhadap sekumpulan bias perseptual dan penilaian.(Sumber:Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,733).

5. Struktur sistem pakar.



Gambar 2.1 Struktur Sistem Pakar

a. (Sumber : Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,733).

Sistem pakar dapat ditampilkan dengan dua lingkungan yaitu: lingkungan pengembangan dan lingkungan konsultasi. Lingkungan pengembangan digunakan oleh ES builder untuk membangun komponen dan memasukkan pengetahuan kedalam basis pengetahuan. Lingkungan konsultasi digunakan nonpakar. Lingkungan konsultasi digunakan nonpakar untuk memperoleh pengetahuan dan nasihat pakar.Lingkungan ini dapat dipisahkan setelah sistem lengkap.

Komponen-komponen yang ada pada sistem pakar :

a. Subsistem akuisisi pengetahuan

Akuisisi pengetahuan adalah akumulasi, transfer, dan transformasi keahlian pemecahan masalah dari pakar atau sumber pengetahuan terdokumentasi ke program komputer, untuk membangun atau memperluas basis pengetahuan .

b. Basis pengetahuan

Basis pengetahuan berisi pengetahuan relevan yang diperlukan untuk memahami, merumuskan, dan memecahkan persoalan. Basis tersebut mencakup dua elemen dasar yaitu :

1) Fakta, misalnya situasi persoalan dan teori area persoalan, dan
2) Heuristik atau aturan khusus yang mengarahkan penggunaan pengetahuan untuk memecahkan persoalan khusus

c. Mesin inferensi

"Otak" ES adalah mesin inferensi,yang dikenal juga sebagai struktur kontrol atau penterjemah aturan (dalam ES berbasis aturan).

Komponen ini sebenarnya adalah program komputer yang meyediakan metodologi untuk mempertimbangkan informasi dalam basis pengetahuan dan *blackboard*,dan merumuskan kesimpulan.

d. Antarmuka pengguna

Sistem pakar berisi prosesor bahasa untuk komunikasi berorientasi-persoalan yang mudah anatara pengguna dan komputer.Komunikasi ini paling baik dilakukan dalam bahasa alami.Dikarenakan batasan teknologi,maka kebanyakan sistem yang ada menggunakan pendekatan pertanyaan dan jawaban untuk berinteraksi dengan pengguan.Acapkali ditambahi dengan menu,formulir elektronik,dan grafik.

e. *Blackboard* (tempat kerja)

Blackboard adalah area kerja memori yang tersimpan sebagai database untuk deskripsi pesoalan terbaru yang ditetapkan oleh data input,digunakan juga untuk perekaman hipotesis dan keputusa sementara.Tiga tipe keputusan dapat direkam dalam *blackboard*: rencana (bagaimana mengatasi persoalan), agenda (tindakan potensial sebelum eksekusi),dan solusi(hipotesis kandidat dan arah tindakan alternatif yang telah dihasilkan sistem sampai dengan saai ini).

f. Subsistem penjelasan (*justifier*)

Kemampuan untuk melacak tanggung jawab suatu kesimpulan terhadap sumbernya adalah penting untuk transfer keahlian dan dalam pemecahan masalah. Subsistem

penjelasan(disebut juga *justifier*) dapat melacak tanggung jawab tersebut dan menjelaskan perilaku ES dengan menjawab pertanyaan berikut secara interaktif :

1) Mengapa suatu pertanyaan ditanyakan oleh sistem pakar ?
2) Bagaimana suatu kesimpulan dicapai ?
3) Mengapa suatu alternatif ditolak ?
4) Apa rencana untuk mencapai solusi ? misalnya,apa yang tetap tersisa sebelum diagnosis akhir ditetapkan.

Dalam sistem pakar sederhana, penjelasan menunjukkan aturan yang digunakan untuk memperoleh rekomendasi tertentu.

g. Sistem perbaikan pengetahuan

Pakar manusia memiliki sistem perbaikan-pengetahuan,yakni mereka dapat menganalisis pengetahuannya sendir dan kegunaannya,belajar darinya,dan meningkatkannya untuk konsultasi mendatang.Serupa pula,evaluasi tersebut diperlukan dalam pembelajaran komputer sehingga program dapat menganalisis alasan keberhasilan atau kegagalannnya.hal ini dapat mengarah kepada peningkatan sehingg menghasilkan basis pengetahuan yang lebih akurat dan pertimbangan yang lebih efektif.Komponen tersebut tidak tersedia dalam sistem pakar komersial saat ini,tetapi sedang dikembangkan dalam ES eksperimental pada beberapa universitas dan lembaga riset.(Sumber:Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,722-724).

6. Cara kerja sistem pakar : mekanisme inferensi

Diantara komponen-komponen dalam gambar 2.1, basis pengetahuan dan mesin inferensi adalah modul paling kritis agar sistem pakar dapat berfungsi dengan baik. Pengetahuan harus direpresentasikan dan diatur secara tepat dalam basis pengetahuan. Mesin inferensi kemudian dapat untuk menarik kesimpulan baru dari fakta dan aturan yang ada. Dalam bagian ini, kami memperkenalkan struktur pengetahuan dan mesin inferensi pada sistem berbasis aturan.(Sumber:Efraim Turban, Jay E. Aronson, Ting Peng Liang,2005,724).

a. Representasi dan organisasi pengetahuan

   Pengetahuan pakar harus direpresentasikan dalam format yang dapat dipahami komputer dan diatur dengan tepat dalam basis pengetahuan sistem pakar.Terdapat banyak cara yang berbeda untuk merepresentasikan pengetahuan manusia, antara lain aturan produksi, jaringan semantik, dan pernyataan logika.

b. Mesin inferensi

   Dalam keputusan kompleks, pengetahuan pakar sering tidak dapat direpresentasikan dalam aturan tunggal. Sebaliknya, aturan dapat digabungkan secara dinamis untuk mencakup berbagai kondisi. Proses penggabungan banyak aturan berdasarkan data yang tersedia, disebut inferensi .Komponen yang melakukan inferensi dalam sistem pakar disebut mesin inferensi. Dua pendekatan populer untuk menarik kesimpulan adalah *forward chaining* dan *backward chaining.*

7. Kategori umum sistem pakar

Berikut adalah tabel yang menunjukkan kategori umum bidang-bidang masalah yang cocok untuk sistem pakar :

Tabel 2.2 Kategori Umum Sistem Pakar.

Kategori	**Persoalan yang ditangani**
Sistem interpretasi	Menyimpulkan deskripsi situasi dari observasi
Sistem prediksi	Menyimpulkan kemungkinan konsekuensi dari suatu situasi
Sistem diagnostik	Menyimpulkan kegagalan sistem dari observasi
Sistem desain	Mengonfigurasikan objek dengan batasan
Sistem perencanaan	Mengembangkan rencana untuk mencapai tujuan
Sistem pengawasan	Membandingkan observasi rencana,memunculkan pengecualian
Sistem debugging	Menyarankan pemulihan untuk kegagalan
Sistem perbaikan	Mengeksekusi rencana untuk mengelola pemulihan yang disarankan
Sistem instruksi	Mendiagnosis,men-debug,dan memperbaiki performa siswa
Sistem kontrol	Menginterpretasikan,memprediksi,memperbaiki,dan mengawasi kelakuan sistem

C. Logika *Fuzzy*

Logika *fuzzy* pertama kali diperkenalkan oleh Prof. Lotfi A. Zadeh pada tahun 1965. Dasar logika *fuzzy* adalah teori himpunan *fuzzy*. Pada teori himpunan *fuzzy*, peranan derajat keanggotaan sebagai penentu keberadaan elemen dalam suatu himpunan sangatlah penting. Nilai keanggotaan atau

derajat keanggotaan atau membership *function* menjadi ciri utama dari penalaran degan logika *fuzzy* tersebut (Purnomo, 2010).

Logika *fuzzy* dapat dianggap sebagai kotak hitam yang menghubungkan antara ruang input dengan ruang output. Kotak hitam tersebut berisi cara atau metode yang dapat digunakan untuk mengolah data *input* menjadi *output* dalam bentuk informasi yang baik.

Ada beberapa alasan mengapa orang menggunakan logika *fuzzy*, yaitu :

1. Konsep logika *fuzzy* mudah dimengerti. Konsep matematis yang mendasari penalaran fuzzy sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat fleksibel
3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi non linear yang sangat kompleks
5. Logika *fuzzy* dapat membangun dan mengaplikasikan pengalamanpengalaman para pakar secara langsung tanpa harus melalui proses pelatihan
6. Logika *fuzzy* dapat bekerjasama dengan teknik-teknik kendali secara konvensional
7. Logika *fuzzy* didasarkan pada bahasa alami

D. Himpunan *Fuzzy*

Pada himpunan tegas (crisp), nilai keanggotaan suatu item x dalam suatu himpunan A,yang sering ditulis dengan μA[x], memiliki 2 kemungkinan yaitu: Satu (1), yang berarti bahwa suatu item menjadi anggota dalam suatu

himpunan atau Nol (0), yang berarti bahwa suatu item tidak menjadi anggota dalam suatu himpunan

Disini bisa dikatakan bahwa pemakaian himpunan *crisp* untuk menyatakan umur sangat tidak adil, adanya perubahan kecil saja pada suatu nilai mengakibatkan perbedaan kategori yamg sangat significant

Himpunan *fuzzy* digunakan untuk mengantisipasi hal tersebut. Seseorang dapat masuk dalam 2 himpunan yang berbeda. Seberapa besar eksistensinya dalam himpunan tersebut dapat dilihat pada nilai keanggotaannya.

Kalau pada himpunan crisp, nilai keanggotaan hanya ada 2 kemungkinan yaitu 0 dan 1, pada himpunan *fuzzy* nilai keanggotaan terletak pada rentang 0 sampai 1. Apabila x memiliki nilai keanggotaan *fuzzy* μA[x] = 0 berarti x tidak menjadi anggota himpunan A, demikian pula apabila x memiliki nilai keanggotaan *fuzzy* μA[x] = 1 berarti x menjadi anggota penuh pada himpunan A.

Terkadang kemiripan antara keanggotaan *fuzzy* dengan probabilitas menimbulkan kerancuan. Keduanya memiliki nilai pada interval [0,1], namun interpretasi nilainya sangat berbeda antara kedua kasus terebut. Keanggotaan *fuzzy* memberikan suatu ukuran terhadap pendapat dan keputusan, sedangkan probabilitas mengindikasikan proporsi terhadap keseringan suatu hasil bernilai benar dalam jangka panjang. Misalnya, jika nilai keanggotaan suatu himpunan *fuzzy* Muda adalah 0,9; maka tidak perlu dipermasalahkan berapa seringnya nilai itu diulang secara individual untuk mengharapkan suatu hasil yang hampir pasti muda. Di lain pihak, nilai probabilitas 0,9 muda berarti 1 0% dari himpunan tersebut diharapkan tidak muda.

Himpunan *fuzzy* memiliki 2 atribut, yaitu:

1. *Linguistic*,yaitu penamaan suatu grup yang mewakili suatu keadaan atau kondisi teretentu dengan menggunakan bahasa alami
2. Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel seperti: 40, 30, 29.

Ada beberapa hal yang perlu diketahui dalam memahami sistem Fuzzy, yaitu:

1. Variabel *Fuzzy*

   Variabel *fuzzy* merupakan variabel yang hendak dibahas dalam suatu sistem *fuzzy*.

   Contoh: umur, temperature, permintaan, dan sebagainya.

2. Himpunan *Fuzzy*

   Himpunan *Fuzzy* merupakan suatu grup yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variabel *fuzzy*.

   Contoh: Variabel umur dibagi menjadi 3 himpunan *fuzzy*, yaitu: muda, parobaya, dan tua.

   Variabel *temperature*, terbagi menjadi 5 himpunan *fuzzy*, yaitu: dingin, sejuk, normal, hangat, dan panas.

3. Semesta pembicaraan

   Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu vaiable *fuzzy*. Semesta pembicaraan merupakan himpunan bilangan *real* yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai semesta pembicaraan ini tidak dibatasi batas atasnya.

Contoh:

Semesta pembicaraan untuk variabel umur: [0+~]

Semesta pembicaraan untuk variabel temterature: [0 40]

4. Domain

Domain himpunan *fuzzy* adalah keseluruhan nilai yang diijinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*. Seperti halnya semesta pembicaraan, domain merupakan himpunan bilangan *real* yang senantiasa nail (bertambah) secara monoton dari kiri ke kanan. Nilai domain dapat berupa bilangan positif maupun negative**.**

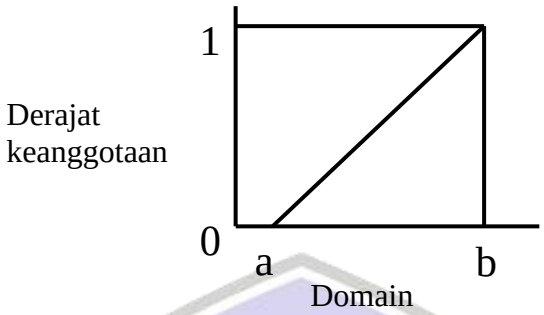
E. Fungsi Keanggotaan

Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Ada beberapa fungsi yang bisa digunakan.

1. Representasi linier

Pada representasi linier, pemetaan input ke derajat keanggotaannya digambarkan sebagai suatu garis lurus. Bentuk ini paling sederhana dan menjadi pilihan yang baik untuk mendekati suatu konsep yang kurang jelas. Ada dua keaadaan himpunan *fuzzy* yang linier. Pertama, kenaikan himpunan dimulai pda nilai domain yang memiliki derajat keanggotaan nol [0] bergerak kekanan menuju ke nilai domain yang memiliki derajat

keanggotaan lebih tinggi atau sering disebut dengan representasi kurva linear naik.



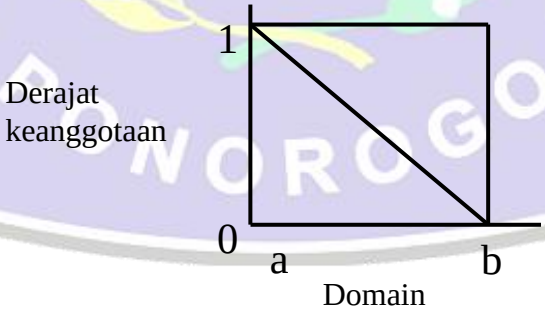
Gambar 2.2 Representasi Kurva Naik

Fungsi keanggotaan

$$\mu(x) = \begin{cases} 0; & x \le a \\ (x-a)/(b-a); & a \le x \le b \\ 1; & x \ge b \end{cases}$$

(2.14)

Kedua, merupakan kebalikan dari yang pertama. Garis lurus dimulai dari nilai domain dengan derajat keanggotaan tertinggi pada sisi kiri, kemudian bergerak menurun ke nilai domain yang memiliki derajat keanggotaan lebih rendah atau sering disebut dengan kurva linear turun.



Gambar 2.3 Representasi Kurva Turun

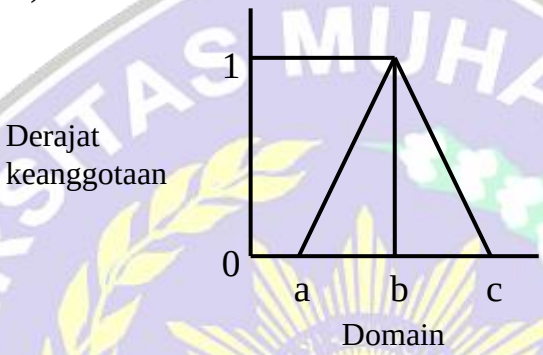
Fungsi keanggotaan:

$$\mu(x) = \begin{cases} (b-x)/(b-a); & a \le x \le b \\ 0; & x \ge b \end{cases}$$

(2.15)

2. Representasi kurva segitiga

Kurva segitiga pada dasarnya merupakan gabungan antara 2 garis (linier).



Gambar 2.4 Representasi Kurva Segitiga

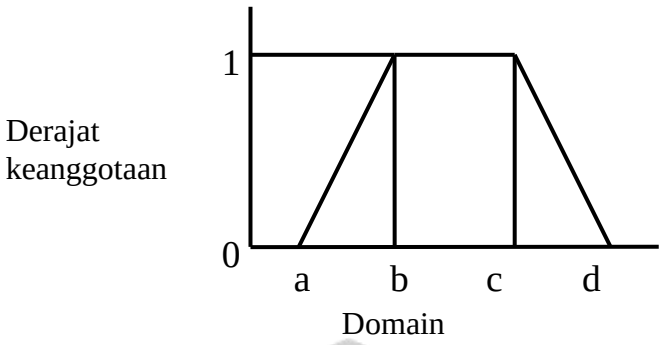
Fungsi keanggotaan:

(2.16)

3. Representasi kurva trapesium

$$\mu[x] = \begin{cases} 0; & x \le a \, atau \, x \ge c \\ (x-a)/(b-a); & a \le x \le b \\ (b-x)/(c-b); & b \le x \le c \end{cases}$$

Kurva trapesium pada dasarnya seperti bentuk segitiga, hanya saja ada titik yang memiliki nilai keanggotaan 1.



Gambar 2.15 Representasi kurva trapesium

Fungsi keanggotaan

$$\mu[x] = \begin{cases} 0; & x \le a \, atau \, x \ge d \\ (x-a)/(b-a); & a \le x \le b \\ 1; & b \le x \le c \\ (d-x)/(d-c); & x \ge d \end{cases}$$

(2.17)

F. Ideentifikasi Kerusakan *Hardware* pada Komputer

Langkah pertama dalam mengembangkan aplikasi adalah mengidentifikasikan masalah yang akan dikaji, adapun masalah-masalah yang akan diambil dalam aplikasi untuk mendeteksi kerusakan hardware pada komputer serta cara mengatasinya sebagai berikut;

1. Komputer tidak mau *start*

   Penyebab gangguan :

   a. Kabel belum terpasang

   b. *UPS* atau *stabilizer* belum dihidupkan

   c. Kabel *power* putus

   d. *Power supplay* rusak

2. Komputer mau hidup tetapi tidak mau *booting*

   Penyebab ganggunan :

   a. Kabel *VGA* atau kendor pemasangannya

   b. Memori rusak

   c. Arus pada *power supplay* tidak memadai

3. Komputer sering *hang*

   Penyebab gangguan :

   a. *CPU* terlalu panas

   b. Kapasitas memori tidak memadai

   c. Memori rusak

   d. Kerusakan pada program atau sistem

   e. Virus

4. *Keyboard* tidak dikenali oleh komputer

   Penyebab gangguan :

   a. *Keyboard* belum terpasang dengan benar

   b. *Keyboard* rusak

   c. Sistem tidak mengenali penambahan *hardware*

   d. *Port* yang rusak

5. Jam dan setting tanggal BIOS selalu berubah-rubah

   Penyebab gangguna :

   a. Baterai *CMOS* sudah tidak berfungsi baik

6. *Crash* setelah memasang *RAM* baru

   Penyebab gangguan :

a. *RAM* tidak kompitabel dengan *motherboard*

b. *RAM* mungkin rusak

7. Monitor tidak mau menyala

   Penyebab gangguan :

   a. Kabel *power* belum terpasang

   b. Kabel *VGA* belum terpasang

   c. *VGA* card rusak

8. *Flasdisk* tidak dikenali di komputer

   Penyebab gangguan :

   a. *Flasdisk* belum terpasang dengan benar

   b. *Flasdisk* rusak

   c. *Port* yang rusak

9. *Mouse* tidak dikenali oleh komputer

   Penyebab gangguan :

   a. *Mouse* belum terpasang dengan benar

   b. *Mouse* rusak

   c. *Port* yang rusak

10. Tampilan tiba-tiba rusak dan computer menjadi hang

    Penyebab gangguan :

    a. Suhu pada *VGA* card panas

    b. *Fan VGA* card tidak hidup

G. *PHP*

PHP merupakan bahasa pemrograman yang berjaa dalam sebuah *web server*. *PHP* diciptakan oleh seorang programmer unix dan perl yang bernaa Rasmush Lerdoft pada bulan Agustus-September tahun 1994. Pada awalnya Rasmush mencoba menciptakan sebuah *script* dalam *website* pribadnya dengan tujuan untu memonitor siapasaja yang pernah mengunjungi websitenya.

Pada awal tahun 1995 *PHP* mulai dknalkan Rasmush kepada programmer pemula, dengan alasan bahwa bahasa yang digunakan dalam *PHP* cukup sederhana dan mudah dipahami. Selanjutnya Rasmush menulis ulang *PHP* dengan bahasa C untuk meningkatkan kecepatan aksesnya. Pada bulan September-Oktober 1995 kode *PHP* ditulis ulang dan digabungkan menjadi *PHP*/FI. Baru kemudian di akhir tahun 1995 dirilis bagi umum secara gratis (Rafiza H;2006;1).

PHP adalah bahasa *server-side scripting* yang menyatu dengan *HTML* untuk membuat halaman *web* yang dinamis. Maksud dari *server side scripting* adalah sitaksdan perintah-perintah yang digunakan sepenuhnya dijalankan di *server* tetapi disertakan pada dokumen *HTML*(Sunarfrihantono Bimo:2003:1)

H. *XAMPP*

Xampp merupakan *software* yang berisi paket pendukung seperti *interpreter PHP, Web Server dan data My Sql. Xampp* meruapakn paket *php* yang berbasis *open source*. Informasi lengkap mengenai produk ini dapat diakses di situs resmi *website*nya, yaitu: http:/www.apachefriends.com xampp

berfungsi sebagai pengembang apikasi (*project*) berbasis *PHP*. *Xampp* mengkombinasikan beberapa paket *software* berbeda kedalam satu paket. Paket-paket yang dimaksud adalah *Apache, My Sql, PHP, Perl, FileZilla FTP Sever, Php my Admin, Open SSL, Free type, Webalizer, mood_perl, Truck MMChace, mcrypt, SQL Lit, JP Grapt, Mercury Mail Transport* Sistem, *PHPB* lender *PHP* Compiler (Riyanto;2009;271).

I. *MySQL*

MySQL adalah seuah program *database* yang mampu menerima dan mengirimkan data denga cepat, multi *user* serta menggunakan perintah standar *SQL* (*Structure Query Language)* MySql merupakan sebuah *database* yang *free*, artinya kia bebas menggunakan database ini untuk kepentingan pribadi dan usaha tanpa harus membeli atau membayar lisensnya. Mysql dirilis oleh seorang programmer database bernama Michael Widenius.

MySQL merupakan seuah data *base server* yang juga dapat berperan sebagai *client* sehingga sering disebut *database client/server yang open source* dengan kemampuan dapat berjala baik baik di Sistem Operasi manapun dengan *platform Wndows* maupun *Linux*(Nugroho Bunafit:2005).

J. *ERD*

Entity Relationship Data (*ERD*) merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. *ERD* untuk memodelkan

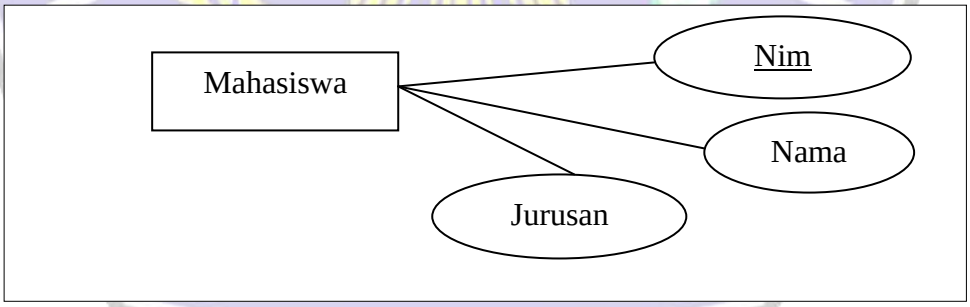
struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol. Ada tiga simbol yang digunakan, yaitu:

1. *Entity*

   Entity adalah orang, tempat, kejadian atau konsep yang informasinya direkam. Pada bidang administrasi siswa misalnya, *entity* adalah siswa, buku, pembayaran, nilai test. Pada bidang kesehatan, *entity* adalah pasien, obat, kamar, diet. Simbol yang digunakan untuk *entity* adalah persegi panjang.

2. *Attribute*

   Setiap *entity* mempunyai *attribute* atau sebutan untuk mewakili suatu entity. Seorang siswa dapat dilihat dari atributenya, misalnya nama, nomor siswa, alamat, nama orang tua, hobby. *Attribute* juga disebut sebagai data elemen, data *field*, data item. (Sumber : Ir. Harianto Kristanto, 2004, 2) Contoh:



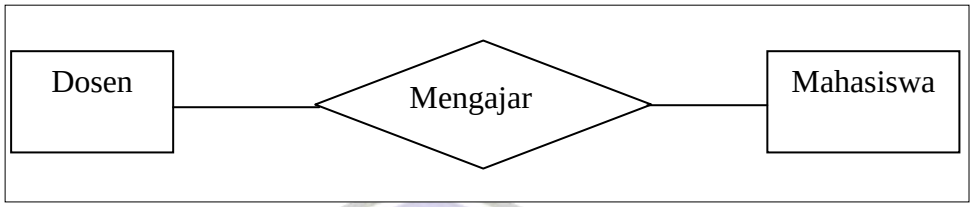
Gambar 2.5 *Atribut* dari Sebuah *Entity*.

3. *Relationship*

   Hubungan yang terjadi antara satu atau lebih *entity*. *Relationship* tidak mempunyai keberadaan fisik, kecuali yang mewarisi hubungan antara *entity* tersebut. *Relationship set* adalah kumpulan *relationship*

yang sejenis. Simbol yang digunakan adalah belah ketupat, *diamond* atau *rectangel*. (Sumber : Linda Marlinda, S.Kom, 2004, 17)

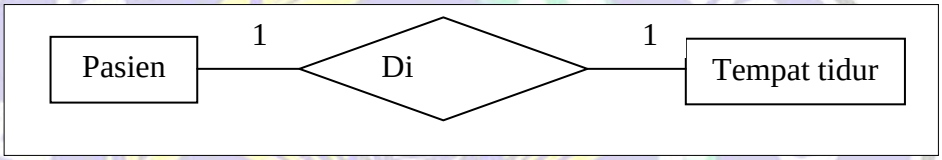
Contoh:



Gambar 2.6 *Relationship*.

a. *One to one* (1:1)

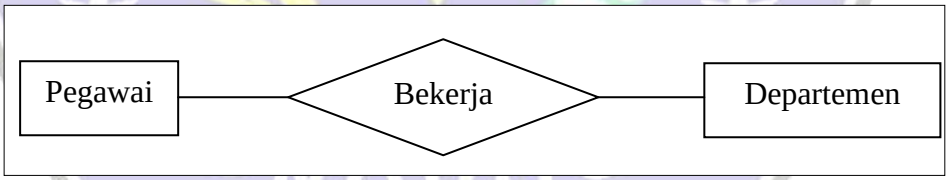
Hubungan satu *entity* dengan satu *entity*



Gambar 2.7 *Relationship One To One*

b. *One to many* (1:M) atau *many to one* (M:1)

Hubungan satu *entity* dengan banyak *entity* atau banyak *entity* dengan satu *entity*.

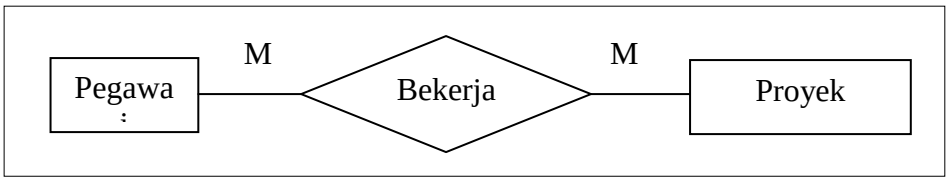


Gambar 2.8 *Relationship One to Many.*

c. *Many to many* (M:M)

Hubungan banyak *entity* dengan banyak *entity*.

.



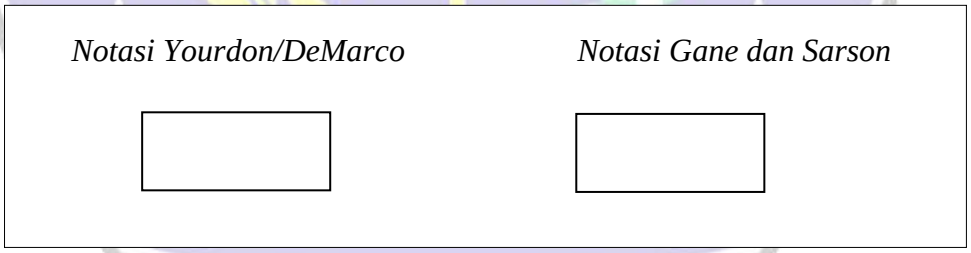
Gambar 2.9 *Relationship Many to Many*.

K. *DFD*

Data Flow Diagram (*DFD*) adalah representasi grafik dari sebuah sistem. *DFD* menggambarkan komponen-komponen sebuah *sistem*, aliran-aliran data dimana komponen-komponen tersebut, dan asal, tujuan dan penyimpanan dari data tersebut.

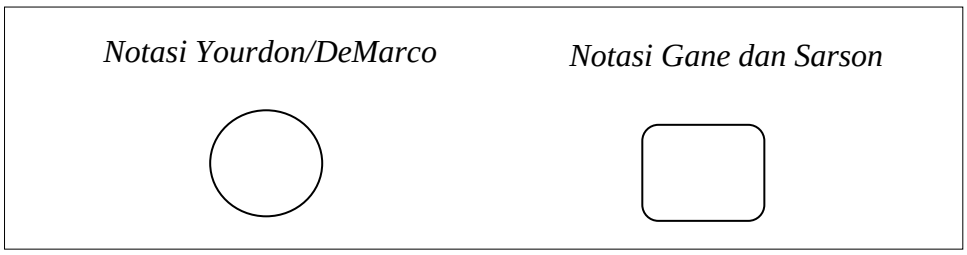
Kita dapat menggunakan *DFD* untuk dua hal utama, yaitu untuk membuat dokumentasi dari sistem informasi yang ada atau untuk menyusun dokumentasi untuk sistem informasi yang baru. Empat simbol yang digunakan dalam *DFD* adalah:

1. Simbol *entitas eksternal/terminator* menggambarkan awal atau tujuan data di luar sistem.



Gambar 2.10 Simbol *Entitas Eksternal/Terminator*.

2. Simbol lingkaran menggambarkan entitas atau proses dimana aliran data masuk ditransformasikan ke aliran data keluar.

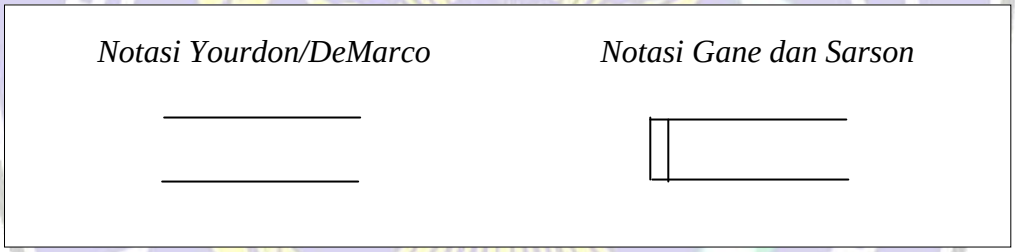


Gambar 2.11 Simbol Lingkaran.

3. Simbol aliran data menggambarkan aliran data

Gambar 2.12 Simbol Aliran Data.

4. Simbol *file* menggambarkan tempat data disimpan.



Gambar 2.11 Simbol *File*.

Abdul Kadir (2007) menjelaskan ada tiga jenis *DFD* yaitu:

1. *Context Diagram* (*CD*)

Jenis pertama *Context Diagram*, adalah *data flow diagram* tingkat atas (*DFD Top Level*), yaitu diagram yang paling tidak detail, dari sebuah sistem informasi yang menggambarkan aliran-aliran data ke dalam dan ke luar sistem dan ke dalam dan ke luar entitas-entitas eksternal. Beberapa hal yang harus diperhatikan dalam menggambar *CD*:

a. Batas sistem adalah batas antara "daerah kepentingan sistem."

b. Lingkungan sistem adalah segala sesuatu yang berhubungan atau mempengaruhi sistem tersebut.

c. *Interface* adalah aliran yang menghubungkan sebuah sistem dengan lingkungan sistem tersebut.

2. *DFD* Fisik

DFD fisik adalah representasi grafik dari sebuah sistem yang menunjukkan *entitas-entitas internal* dan *eksternal* dari sistem tersebut, dan aliran-aliran data ke dalam dan ke luar dari entitas-entitas tersebut. *Entitas-entitas internal* adalah personel, tempat (sebuah bagian), atau mesin (misalnya, sebuah komputer) dalam sistem tersebut yang mentransformasikan data. Maka *DFD* fisik tidak menunjukkan apa yang dilakukan, tetapi menunjukkan dimana, bagaimana, dan oleh siapa proses-proses dalam sebuah sistem dilakukan.

Perlu diperhatikan didalam memberikan keterangan di lingkaran-lingkaran (simbol proses) dan aliran-aliran data (simbol aliran data) dalam *DFD* fisik menggunakan label atau keterangan dari kata benda untuk menunjukkan bagaimana sistem mentransmisikan data antara lingkaran-lingkaran tersebut.

Misal:

Aliran data : Kas, formulir 66W, slip setoran

Proses : Cleck penjualan, kasir, pembukuan, dll.

3. *DFD Logis*

DFD logis adalah grafik dari sebuah sistem yang menunjukkan proses-proses dalam sistem tersebut dan aliran-aliran data ke dalam dan ke luar dari proses-proses tersebut. Kita menggunakan *DFD* logis untuk membuat dokumentasi sebuah sistem informasi karena *DFD* logis dapat mewakili logika tersebut, yaitu apa yang dilakukan oleh sistem tersebut, tanpa perlu menspesifikasi dimana, bagaimana dan oleh siapa proses-proses dalam sistem dilakukan. (Sumber : Abdul Kadir, 2007, 51-53).

L. *FLOWCHART*

1. Pengertian *Flowchart* (Diagram Alur).

Karena komputer membutuhkan hal-hal yang rinci, maka bahasa pemrograman bukanlah alat baik untuk merancang sebuah algoritma awal. Alat yang banyak dipakai untuk membuat algoritma adalah diagram alur (*flowchart*).

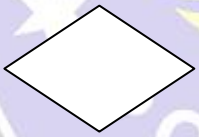
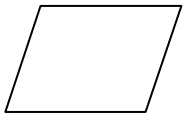
Diagram alur dapat menunjukkan secara jelas arus pengendalian suatu algoritma, yakni melaksanakan suatu rangkaian kegiatan secara logis dan sistematis. Suatu diagram alur dapat memberi gambaran dua dimensi berupa simbol-simbol grafis. Masing-masing simbol telah ditetapkan lebih dahulu fungsi dan artinya. Simbol-simbol tersebut dipakai untuk menunjukkan berbagai kegiatan operasi dan jalur pengendalian. Arti khusus dari sebuah *flowchart* adalah simbol-simbol yang digunakan untuk menggambarkan urutan proses yang terjadi di

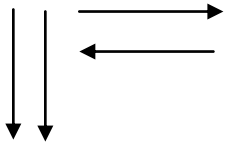
dalam suatu program komputer secara sistematis dan logis. (Sumber : Tata Sutabri, S.Kom. MM, 2004,21)

2. Simbol-simbol *Flowchart*

Sudah dikemukakan di atas bahwa diagram alur atau *flowchart* memiliki beberapa simbol yang biasa digunakan untuk menggambarkan rangkaian proses yang harus dilaksanakan. Simbol tersebut dijelaskan di bawah ini: (Sumber : Tata Sutabri, S.Kom. MM, 2004, 21-22)

Tabel 2.3 Simbol-Simbol *Flowchart*

Simbol *Flowchart*	Fungsi
1. 	*TERMINAL* Simbol ini digunakan untuk mengawali atau mengakhiri suatu proses/kegiatan.
2. 	*PREPARATION* Simbol ini digunakan untuk mempersiapkan harga awal/nilai awal suatu variabel yang akan diproses dan digunakan untuk proses *loop*.
3. 	*DECISION* Simbol ini digunakan untuk pengujian suatu kondisi yang sedang diproses.
4. 	PROSES Simbol ini digunakan untuk menggambarkan suatu proses yang sedang dieksekusi.
5. 	*INPUT/OUTPUT* Simbol ini digunakan untuk menggambarkan proses *input* (*read*) maupun proses *output* (*print*).

6.		*SUBROUTINE* digunakan untuk menggambarkan proses pemanggilan subprogram dari mainprogram.
7.		*FLOW LINE* Simbol ini digunakan untuk menggambarakan arus proses dari suatu kegiatan ke kegiatan lain.
8.		*CONECTOR* Simbol ini digunakan sebagai penghubung antara suatu proses dengan proses lainnya yang ada di dalam suatu lembar halaman.
9.		*PAGE CONECTOR* Simbol ini digunakan sebagai penghubung antara suatu proses dengan proses lainnya, tetapi berpindah halaman.
10.		*MANUAL OPERATION* Simbol ini digunakan untuk menggambarkan suatu kegiatan atau proses yang bersifat manualisasi.
11.		*PRINTER* Simbol ini digunakan untuk menggambarkan suatu dokumen atau suatu kegiatan mencetak suatu informasi dengan mesin *printer*.
12.		*CONSOLE* Simbol ini digunakan untuk menggambarkan suatu kegiatan menampilkan data atau informasi melalui

		monitor atau CRT (*Cathode Ray Tube*).
13.		*DISK* Simbol ini digunakan untuk menggambarkan suatu kegiatan membaca atau menulis data menggunakan media *magnetic disk*.
14.		*MANUAL INPUT* Simbol ini digunakan untuk menggambarkan proses pemasukan data melalui media *keyboard*.
15.		*TAPE* Simbol ini digunakan untuk menggambarkan suatu kegiatan membaca atau menulis data menggunakan media *magnetic tape*.

3. Jenis-jenis *Flowchart*

Bentuk diagram alur (*flowchart*) yang sering digunakan dalam proses pembuatan suatu program komputer adalah sebagai berikut:

a. Program *Flowchart*.

Simbol-simbol yang menggambarkan proses secara rinci dan detil antara intruksi yang satu dengan intruksi yang lainnya dalam suatu program komputer yang bersifat logik.

b. Sistem *Flowchart*.

Simbol-simbol yang menggambarkan urutan prosedur secara detil dalam suatu sistem komputerisasi bersifat fisik.

c. Teknik Pembuatan *Flowchart*.

Sebelum kita membuat sebuah program komputer, yang harus kita lakukan sebelumnya adalah membuat *flowchart*. Jenis *flowchart* yang sering digunakan adalah program *flowchart*.

Teknik pembuatan program *flowchart* ini dibagi menjadi dua bagian, yaitu:

1) *General Way*
2) *Iteration way*

Berikut adalah penjelasan dari kedua teknik tersebut:

1) *General Way*

Teknik pembuatan *flowchart* dengan cara ini lazim digunakan untuk menyusun logika suatu program. Teknik ini menggunakan pengulangan proses secara tidak langsung (*Non-Direct-Loop*).

2) *Iteration Way*

Teknik pembuatan *flowchart* dengan cara ini biasanya dipakai untuk logika program yang cepat dan bentuk permasalahannya kompleks. Pengulangan proses yang terjadi bersifat langsung (*Direct-Loop*). (Sumber : Tata Sutabri, S.Kom. MM; 2004; 24)