BAB II

LANDASAN TEORI

2.1 Kecerdasan Buatan (*Artificial Intelligence*)

Sistem pakar atau *expert system* merupakan suatu cabang dari ilmu komputer yang berdasarkan kecerdasan buatan (*artificial intelligence*). *Artificial intelligence* adalah sebuah rancangan program yang memungkinkan komputer melakukan suatu tugas atau mengambil keputusan dengan meniru suatu cara berpikir dan penalaran manusia.

Adapun beberapa definisi *artificial intelligence* adalah sebagai berikut :

a. Definisi *artificial intelligence* menurut Jog [4] adalah :

   "*Artificial Intelligence didefinisikan sebagai suatu mesin atau alat pintar (biasanya adalah komputer) yang dapat melakukan suatu tugas yang bilamana tugas tersebut dilakukan oleh manusia akan dibutuhkan suatu kepintaran untuk melakukannya"*.

b. Definisi *artificial intelligence* menurut Kus [6] adalah :

   "*Kecerdasan buatan atau artificial intelligence merupakan salah satu bagian ilmu komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia"*.

c. Definisi *artificial intelligence* menurut Sar [9] adalah :

   "Kecerdasan buatan adalah salah satu bidang ilmu komputer yang mendayagunakan komputer sehingga dapat berperilaku cerdas seperti manusia".

Dengan perancangan *artificial intelligence* yang baik, maka diharapkam peran manusia dapat diminimalkan dan dapat meringankan beban kerja manusia. Cara kerja *artificial intelligence* adalah menerima *input*, kemudian diproses dan kemudian mengeluarkan *output* berupa suatu keputusan.

Implementasi dari *artificial intelligence* saat ini umum ditemui dalam bidang-bidang seperti berikut :

a. *Computer Vision* : yaitu suatu metode *artificial intelligence* yang memungkinkan sebuah sistem komputer mengenali gambar sebagai inputnya. Contohnya adalah mengenali dan membaca tulisan yang ada didalam gambar.
b. *Fuzzy Logic* : yaitu suatu metode *artificial intelligence* yang banyak terdapat pada alat-alat elektronik dan robotika. Dimana alat-alat elektronik atau robotika tersebut mampu berpikir dan bertingkah laku sebagaimana layaknya manusia.
c. *Game* : yaitu suatu metode *artificial intelligence* yang berguna untuk meniru cara berpikir seorang manusia dalam bermain *game*. Contohnya adalah program *Perfect Chessmate* yang mampu berpikir setara dengan seorang *grandmaster* catur.
d. *General Problem Solving* : yaitu suatu metode *artificial intelligence* yang berhubungan dengan pemecahan suatu masalah terhadap suatu situasi yang akan diselesaikan oleh komputer. Biasanya permasalahan tersebut akan diselesaikan secara *trial and error* sampai sebuah solusi dari sebuah

masalah didapatkan. Contohnya adalah program *Eureka* yang dapat memecahkan model *linier programming.*

e. *Speech Recognition* : yaitu suatu metode *artificial intelligence* yang berguna untuk mengenali suara manusia dengan cara dicocokkan dengan acuan yang telah diprogramkan sebelumnya. Contohnya adalah suara dari *user* dapat diterjemahkan menjadi sebuah perintah bagi komputer.

f. *Expert System* : yaitu suatu metode *artificial intelligence* yang berguna untuk meniru cara berfikir dan penalaran seorang ahli dalam mengambil keputusan berdasarkan situasi yang ada. Dengan *expert system* seorang *user* dapat melakukan konsultasi kepada komputer, seolah-olah *user* tersebut berkonsultasi langsung kepada seorang ahli. Contohnya adalah program aplikasi yang mampu meniru seorang ahli medis dalam mendeteksi demam berdarah berdasarkan keluhan-keluhan pasiennya.

2.2 Sistem Pakar (*Expert System*)

2.2.1 Konsep Dasar Sistem Pakar

Sistem pakar (*expert system*) adalah suatu sistem komputer berbasis pengetahuan yang menyamai kemampuan pengambilan keputusan dari seorang pakar. Adapun beberapa definisi tentang sistem pakar (*expert system*) lainnya, adalah sebagai berikut :

a. Menurut Kus [6], sistem pakar didefinisikan sebagai berikut :

   "Sistem pakar adalah suatu sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli".

b. Menurut Muh [8], sistem pakar didefinisikan sebagai berikut :

"Sistem pakar adalah salah satu cabang dari artificial intelligence yang membuat penggunaan secara luas knowledge yang khusus untuk penyelesaian masalah tingkat manusia yang pakar".

Sistem pakar telah dibuat untuk memecahkan masalah-masalah dalam berbagai bidang, antara lain matematika, teknik, kedokteran, ilmu komputer, sampai bidang hukum. Walaupun sistem pakar sebagai sistem komputer yang dalam berbagai hal bekerjanya jauh lebih baik dari manusia atau ahli, tetapi kita tidak bisa menghilangkan begitu saja faktor manusia dan digantikan oleh sistem komputer, karena pada banyak situasi keahlian manusia tetap dibutuhkan, sebab kemampuan komputer terbatas.

Pengetahuan dari suatu sistem pakar mungkin dapat direpresentasikan dalam sejumlah cara. Salah satu metode yang paling umum untuk merepresentasikan pengetahuan adalah dalam bentuk tipe aturan (*rule*) *IF..THEN* (jika..maka). Walaupun cara diatas sangat sederhana, namun banyak hal yang berarti dalam membangun sistem pakar dengan mengekspresikan pengetahuan pakar dalam bentuk aturan diatas.

Konsep dasar dari suatu sistem pakar mengandung beberapa unsur atau elemen, yaitu keahlian, ahli, pengalihan keahlian, inferesi, aturan dan kemampuan menjelaskan.

Keahlian merupakan suatu penguasaan pengetahuan dibidang tertentu yang didapatkan dari pelatihan, membaca atau pengalaman.

Seorang ahli adalah seorang yang mempunyai pengetahuan tertentu dan mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan (*domain*), menyusun kembali pengetauan jika dipandang perlu, dan menentukan relevan atau tidaknya keahlian mereka.

Pengalihan keahlian dari para ahli untuk kemudian dialihkan lagi ke orang lain yang bukan ahli, merupakan tujuan utama dari sistem pakar. Proses ini membutuhkan empat aktivitas, yaitu tambahan pengetahuan dari para ahli atau sumber-sumber lainnya, representasi pengetahuan kedalam komputer, inferensi pengetahuan dan pengalihan pengetahuan ke pengguna. Pengetahuan yang disimpan dikomputer dinamakan dengan basis pengetahuan (*knowledge base*).

Salah satu fitur yang harus dimiliki oleh sistem pakar adalah kemampuan untuk menalar (*reasoning*). Jika keahlian-keahlian sudah tersimpan sebagai basis pengetahuan dan sudah tersedia program yang mampu mengakses basis data, maka komputer harus dapat diprogram untuk membuat inferensi. Proses ini dibuat dalam bentuk motor inferensi (*inference engine*).

Sistem pakar menggunakan basis pengetahuan (*knowldege base*) sebagai dasar pemikirannya. *Knowledge base* tersebut terdiri dari heuristik dan sejumlah *rule-rule* dan aturan-aturan yang tersusun secara sistematis dan spesifik, juga relasi antara data dan aturan (*rule*) dalam pengambilan kesimpulan. *Knowledge base* tersebut disimpan dalam sebuah basis data pada *database*.

Sedangkan sebagai pusat pemrosesannya adalah *inference engine*, yaitu suatu rancangan aplikasi yang berfungsi untuk memberikan pertanyaan dan menerima *input* dari *user*, kemudian melakukan proses logika sesuai dengan

knowledge base yang tersedia, untuk selanjutnya menghasilkan *output* berupa suatu kesimpulan atau bisa juga berupa keputusan sebagai hasil akhir konsultasi.

Knowledge acquisition source berfungsi sebagai penterjemah dari *knowledge base* menjadi sebuah bahasa yang dapat dimengerti oleh *user*. Bagian ini diperlukan karena *knowledge base* yang disimpan dalam sebuah *database*, disimpan dalam format tertentu, sedemikian rupa sehingga *user* sulit mengartikannya.

Disk (*working memory*) adalah sejumlah modul *memory* yang menyimpan informasi sementara dari suatu proses konsultasi. Setiap proses baru dijalankan, *memory* tersebut akan di set ke kondisi awal. Dalam menjalankan proses, *memory* tersebut menyimpan informasi dari *rule-rule* yang dipakai dalam *knowledge base*.

Oleh karena itu, langkah-langkah perancangan sistem pakar sebaiknya mengikuti urutan sebagai berikut :

1. Menentukan batasan-batasan atau bidang konsentrasi dari sebuah sistem pakar yang akan dirancang.
2. Memilih jenis keputusan apa yang diambil.
3. Membuat pohon keputusan (*decision tree*).
4. Menuliskan *IF-THEN rules*.
5. Merancang antarmuka pengguna (*user interface*).

2.2.2 Ciri-ciri Sistem Pakar

Sistem pakar memiliki ciri-ciri sebagai berikut :

1. Terbatas pada bidang yang spesifik.
2. Dapat memberikan penalaran untuk data-data yang tidak pasti.

3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami.
4. Berdasarkan pada *rule* atau kaidah tertentu.
5. Outputnya bersifat nasihat atau anjuran dan tergantung dari dialog *user*.
6. *Knowledge base* dan *inference engine* terpisah.

2.2.3 Orang Yang Terlibat Dalam Sistem Pakar

Untuk memahami perancangan sistem pakar, perlu dipahami mengenai siapa saja yang berinteraksi dengan sistem. Mereka adalah :

1. Pakar (*domain expert*) adalah seseorang ahli yang dapat menyelesaikan masalah yang sedang diusahakan untuk dipecahkan oleh sistem.
2. Pembangun pengetahuan (*knowledge engineer*) merupakan seseorang yang menerjemahkan pengetahuan seorang pakar dalam bentuk deklaratif sehingga dapat digunakan oleh sistem pakar.
3. Pengguna (*user*) merupakan seseorang yang berkonsultasi dengan sistem untuk mendapatkan saran yang disediakan oleh pakar.
4. Pembangun sistem (*system engineer*) merupakan seseorang yang membuat antarmuka pengguna, merancang bentuk basis pengetahuan secara deklaratif dan mengimplementasikan mesin inferensi.

2.2.4 Kategori Masalah Sistem Pakar

Masalah-masalah yang dapat diselesaikan dengan sistem pakar, diantaranya adalah :

1. Interpretasi, yaitu membuat kesimpulan atau deskripsi dari sekumpulan data mentah, termasuk diantaranya juga pengawasan, pengenalan ucapan, analisis citra, interpretasi sinyal dan beberapa analisis kecerdasan.
2. Proyeksi, yaitu memprediksi akibat-akibat yang dimungkinkan dari situasi-situasi tertentu, diantaranya peramalan, prediksi demografis, peramalan ekonomi, prediksi lalu lintas atau peramalan keuangan.
3. Diagnosis, yaitu menentukan sebab malfungsi dalam situasi kompleks yang didasarkan pada gejala-gejala yang teramati, diantaranya medis, elektronis, mekanis dan diagnosis perangkat lunak.
4. Desain, yaitu menentukan konfigurasi komponen-komponen sistem yang cocok dengan tujuan-tujuan kinerja tertentu yang memenuhi kendala-kendala tertentu, diantaranya layout sirkuit dan perancangan bangunan.
5. Perencanaan, yaitu merencanakan serangkaian tindakan yang akan dapat mencapai sejumlah tujuan dengan kondisi awal tertentu, diantaranya perencanaan keuangan, komunikasi militer dan manajemen proyek.
6. *Monitoring*, yaitu membandingkan tingkah laku suatu sistem yang teramati dengan tingkah laku yang diharapkan darinya, daiantaranya *Computer Aided Monitoring System*.
7. *Debugging* dan *repair*, yaitu menentukan dan mengimplementasikan cara-cara untuk mengatasi malfungsi, diantaranya memberikan resep obat terhadap suatu kegagalan.

8. Instruksi, yaitu mendeteksi dan mengoreksi defisiensi dalam pemahaman domain subjek, diantaranya melakukan instruksi untuk diagnosis, *debugging* dan perbaikan kinerja.
9. Pengendalian, yaitu mengatur tingkah laku suatu *environment* yang kompleks seperti kontrol terhadap interpretasi-interpretasi, prediksi, perbaikan dan monitoring kelakuan sistem.
10. Seleksi, mengidentifikasi pilihan terbaik dari sekumpulan kemungkinan.
11. Simulasi, pemodelan interaksi antara komponen-komponen sistem.

2.2.5 Kelebihan Sistem Pakar

Seorang pakar dengan sistem pakar mempunyai banyak perbedaan. Perbandingan kemampuan antara seorang pakar dengan sebuah sistem pakar dapat dilihat seperti pada tabel 2.1 berikut ini :

Tabel 2.1 Perbandingan kemampuan seorang pakar dengan sistem pakar

Faktor	*Human Expert*	*Expert System*
Time availibility	Hari Kerja	Setiap Saat
Geografis	Lokal/tertentu	Dimana saja
Kemananan	Tidak tergantikan	Dapat diganti
Perishable/dapat habis	Ya	Tidak
Performansi	Variable	Konsisten
Kecepatan	Variable	Konsisten
Biaya	Tinggi	Terjangkau

Dari tabel 2.1 diatas dapat dikembangkan penjelasan lebih lanjut tentang kelebihan sistem pakar.

1. Sistem pakar bisa digunakan setiap hari menyerupai sebuah mesin sedangkan seorang pakar tidak mungkin bekerja terus menerus setiap hari tanpa beristirahat.
2. Sistem pakar merupakan suatu *software* yang dapat diperbanyak dan kemudian dibagikan keberbagai lokasi maupun tempat yang berbeda-beda untuk digunakan, sedangkan seorang pakar hanya berkerja pada satu tempat dan pada saat yang bersamaan.
3. Suatu sistem pakar dapat diberi pengamanan untuk menentukan siapa saja yang mempunyai hak akses untuk menggunakannya dan jawaban yang diberikan oleh sistem terbebas dari proses intimidasi atau ancaman, sedangkan seorang pakar bisa saja mendapat ancaman atau tekanan pada saat menyelesaikan permasalahan.
4. Pengetahuan (*knowledge*) yang disimpan pada sistem pakar tidak akan bisa hilang atau lupa, yang dalam hal ini tentu harus didukung oleh *maintenance* yang baik, sedangkan pengetahuan seorang pakar manusia lambat laun akan hilang karena meninggal, usia yang semakin tua, maupun menderita suatu penyakit.
5. Kemampuan memecahkan masalah pada suatu sistem pakar tidak dipengaruhi oleh faktor dari luar seperti intimidasi, perasaan kejiwaan, faktor ekonomi ataupun perasaan tidak suka. Akan tetapi sebaliknya dengan seorang pakar yang disebutkan dipengaruhi oleh faktor-faktor luar seperti yang disebutkan diatas ketika sedang menyelesaikan atau memecahkan suatu masalah, sehingga dapat memunculkan jawaban yang

berbeda-beda atas pertanyaan yang diajukan walaupun masalahnya sama. Atau dengan kata lain, seorang pakar boleh jadi tidak konsisten.

6. Umumnya kecepatan dalam memecahkan masalah pada suatu sistem pakar relatif lebih cepat dibandingkan oleh seorang pakar manusia. Hal ini sudah dibuktikan pada beberapa sistem pakar yang tekenal didunia.
7. Biaya menggaji seorang pakar lebih mahal bila dibandingkan dengan penggunaan program sistem pakar (dengan asumsi bahwa program sistem pakar itu sudah ada).

2.2.6 Keuntungan Sistem Pakar

Sistem pakar merupakan paket perangkat lunak atau paket program komputer yang ditujukan sebagai penyedia nasihat dan sarana bantu dalam memecahkan masalah. Ada beberapa keunggulan dari sistem pakar, diantaranya dapat :

1. Menghimpun data dalam jumlah yang sangat besar.
2. Menyimpan data tersebut untuk jangka waktu yang panjang dalam suatu bentuk tertentu.
3. Mengerjakan perhitungan secara cepat dan tepat serta mencari kembali data yang tersimpan dengan kecepatan tinggi.

Selanjutnya ada banyak keuntungan bila menggunakan sistem pakar, keuntungan-keuntungan tersebut diatantaranya adalah :

1. Menjadikan pengetahuan dan nasihat lebih mudah didapat.
2. Meningkatkan *output* dan produktivitas.
3. Menyimpan kemampuan dan pengetahuan pakar.

4. Meningkatkan penyelesaian masalah.
5. Meningkatkan reliabilitas.
6. Memberikan jawaban yang cepat.
7. Merupakan panduan yang cerdas.
8. Dapat bekerja dengan informasi yang kurang lengkap dan tidak pasti.

2.2.7 Kelemahan Sistem Pakar

Adapun kelemahan-kelemahan yang dimiliki oleh sebuah sistem pakar diantaranya adalah :

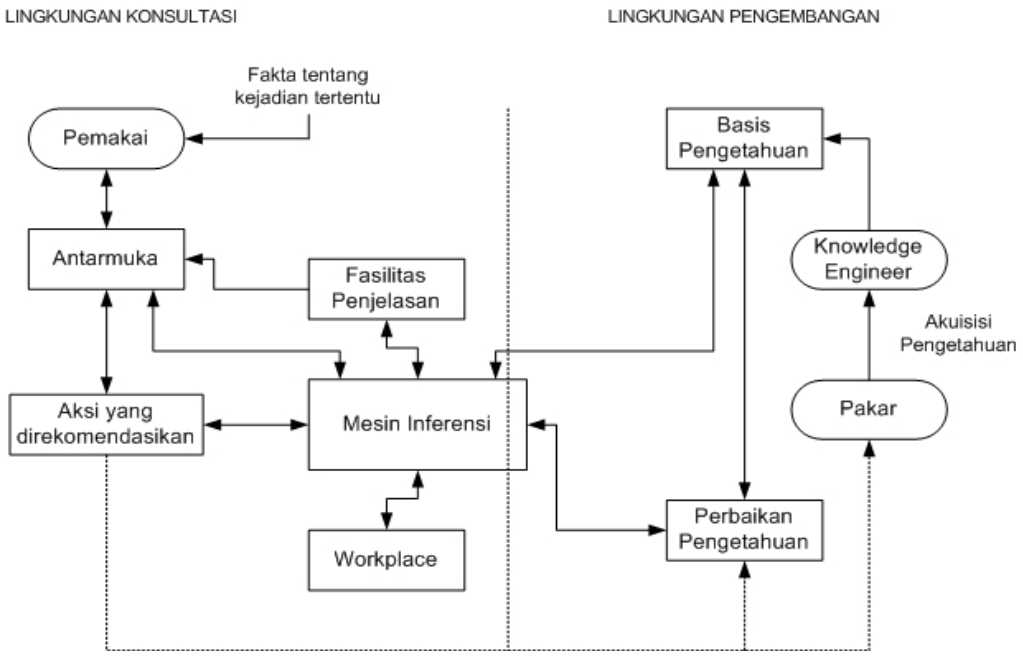
1. Masalah dalam mendapatkan pengetahuan dimana pengetahuan tidak selalu bisa didapatkan dengan mudah, karena kadangkala pakar dari masalah yang kita buat tidak ada, dan kalaupun ada kadang-kadang pendekatan yang dimiliki oleh pakar berbeda-beda.
2. Untuk membuat suatu sistem pakar yang benar-benar berkualitas tinggi sangatlah sulit dan memerlukan biaya yang sangat besar untuk pengembangan dan pemeliharaanya.
3. Boleh jadi sistem tak dapat membuat keputusan.
4. Sistem pakar tidaklah 100% menguntungkan, walaupun seorang tetap tidak sempurna atau tidak selalu benar. Oleh karena itu perlu diuji ulang secara teliti sebelum digunakan. Dalam hal ini manusia tetap merupakan faktor dominan.

Kelemahan-kelemahan atau kekurangan dari sistem pakar tersebut bukanlah sama sekali tidak bisa diatasi, tetapi dengan terus melakukan perbaikan

dan pengolahan berdasarkan pengalaman yang telah ada maka hal itu diyakini akan dapat diatasi walaupun dalam waktu yang panjang dan terus menerus.

2.2.8 Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian utama yaitu *development environment* dan *consultation environment*. *Development environment* dipakai oleh pembangun sistem pakar untu membangun komponen-komponen dan mengenalkan suatu sistem pakar pengetahuan kepada *knowledge base*. *Consultatition environment* dipakai oleh *user* untuk mendapatkan suatu pengetahuan yang berhubungan dengan suatu keahlian. Komponen-komponen sistem pakar dalam kedua bagian tersebut dapat dilihat dalam Gambar 2.1 dibawah ini :



Gambar 2.1 Struktur Sistem Pakar

Komponen-komponen yang terdapat dalam sistem pakar adalah seperti yang terdapat pada gambar 2.1, yaitu basis pengetahuan (*knowledge base*),

akuisisi pengetahuan, mesin inferensi (*inference enine*), area memori kerja (*workplace*), antarmuka pengguna (*user interface*), fasilitas penjelasan, perbaikan pengetahuan.

Secara umum pemakai berinteraksi dengan sistem pakar melalui *interface* pemakai. *Interface* tersebut terdapat berupa pertanyaan-jawaban, menu pilihan, bahasa alami atau grafik. Sedangkan subsistem penjelasan dalam sistem pakar memungkinkan program untuk menjelaskan penalarannya kepada pemakai yang antara lain menyangkut kebutuhan sistem terhadap data tertentu.

2.2.8.1 *Knowledge Base*

Knowledge base adalah suatu struktur data yang menyimpan informasi data, *rule*, relasi antara data dengan *rule* dalam pengambilan kesimpulan. *Knowledge base* dapat dikatakan sebagai kumpulan informasi dan pengalaman seorang ahli pada suatu bidang tertentu. *Knowledge base* terdiri dari fakta seperti halnya buku teori dan heuristik yang merupakan langkah-langkah seorang ahli dalam mengambil keputusan.

Setiap kasus yang ditemui, diproses berdasarkan pengalaman yang pernah tercatat, baru dapat diambil kesimpulan. Apabila terjadi suatu kasus yang belum pernah dialami, maka sistem pakar tidak mampu mengambil kesimpulan dari kasus tersebut. Untuk itu diperlukan penambahan data baru yang dapat dijadikan acuan yang melengkapi *knowledge base*. Jadi, suatu *knowledge base* dapat dikatakan baik apabila memiliki sejumlah *rule* yang mampu digunakan dalam setiap kemungkinan kasus dalam ruang lingkup tertentu. Dalam perancangannya,

perancang harus mampu menggali sebanyak mungkin informasi dari nara sumbernya, seorang ahli dan memindahkannya kedalam *knowledge base*.

Dalam studi kasus pada sistem berbasis pengetahuan terdapat beberapa karakteristik yang dibangun untuk membantu kita di dalam membentuk serangkaian prinsip-prinsip arsitekturnya. Prinsip tersebut meliputi :

a. Pengetahuan merupakan kunci kekuatan sistem pakar.
b. Pengetahuan sering tidak pasti dan tidak lengkap.
c. Pengetahuan sering miskin spesifikasi.
d. Amatir menjadi ahli secara bertahap.
e. Sistem pakar harus fleksibel.
f. Sistem pakar harus transparan.

2.2.8.2 Akuisisi Pengetahuan

Akuisisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan kedalam program komputer. Tahap akuisisi pengetahuan ini biasanya dikerjakan oleh seorang *knowledge engineer*. *Knowledge engineer* adalah seorang yang memiliki latar belakang pengetahuan tentang komputer dan mengerti cara pengembangan sistem pakar serta yang menerjemahkan pengetahuan seorang pakar dalam bentuk deklaratif sehingga dapat digunakan oleh sistem pakar. Dalam tahap ini *knowledge engineer* berusaha menyerap pengetahuan untuk selanjutnya ditransfer kedalam basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, pengalaman pemakai, laporan penelitian dan sebagainya. Dalam

akuisisi pengetahuan (*knowledge acquisition*) ini terdapat empat metode utama, yaitu :

1. Wawancara

   Wawancara adalah metode akuisisi yang paling banyak digunakan. Metode ini melibatkan pembicaraan dengan pakar secara langsung dalam suatu wawancara.

2. Analisis Protokol

   Dalam metode akuisisi ini, pakar diminta untuk melakukan suatu pekerjaan dan mengungkapkan proses pemikirannya dengan menggunakan kata-kata. Pekerjaan tersebut direkam, dituliskan dan dianalisis.

3. Observasi pada pekerjaan pakar

   Dalam metode ini, pekerjaan dalam bidang tertentu yang dilakukan pakar direkam dan diobservasi.

4. Induksi aturan

   Metode ini dibatasi untuk sistem berbasis aturan. Induksi adalah suatu proses penalaran dari khusus ke umum. Suatu sistem induksi aturan diberi contoh-contoh dari suatu masalah yang hasilnya telah diketahui. Setelah diberikan beberapa contoh, sistem induksi aturan tersebut dapat membuat aturan yang benar untuk kasus-kasus contoh. Selanjutnya aturan dapat digunakan untuk menilai kasus lain yang hasilnya tidak diketahui.

2.2.8.3 Mesin Inferensi

Mesin inferensi atau *inference engine* adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace*, dan untuk memformulasikan kesimpulan.

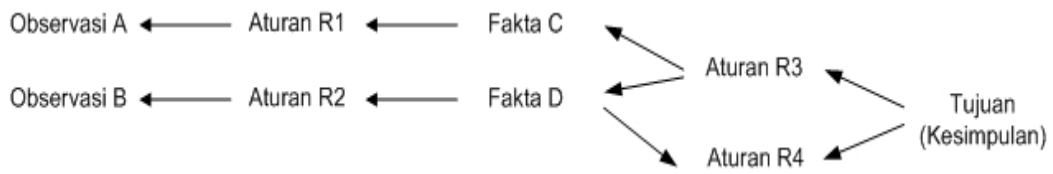
Seperti halnya *knowledge base*, *inference engine* juga terdiri dari *rule-rule* dan fakta-fakta, bedanya apabila didalam knowledge base, *rule-rule* dan fakta-fakta yang ada berisi sebuah domain yang spesifik dari sifat-sifat seorang ahli. Sedangkan *rule-rule* dan fakta-fakta pada *inference engine*, berisi strategi pencarian kesimpulan yang dilakukan oleh *expert system.*

Pemisahan kedua jenis *rule-rule* dan fakta-fakta tersebut memiliki beberapa keuntungan, yang pertama, memudahkan untuk melakukan perubahan dalam *knowledge base* tanpa berimbas banyak pada *inference engine*, dan sebaliknya. Kedua, berguna untuk pengembangan dan komponen-komponen lainnya yang terdapat pada sistem pakar (terkecuali *knowledge base*).

Tujuan utama dari sistem pakar adalah membentuk dan mengeluarkan saran. Untuk menyelesaikan tugas ini, sistem pakar harus melakukan pencarian solusi yang mana adalah tanggung jawab dari *inference engine* untuk melakukannya secara efisien dan efektif. Seringkali dalam proses pencarian, *user* dihadapkan pada sejumlah alternatif dan berbagai jenis batasan.

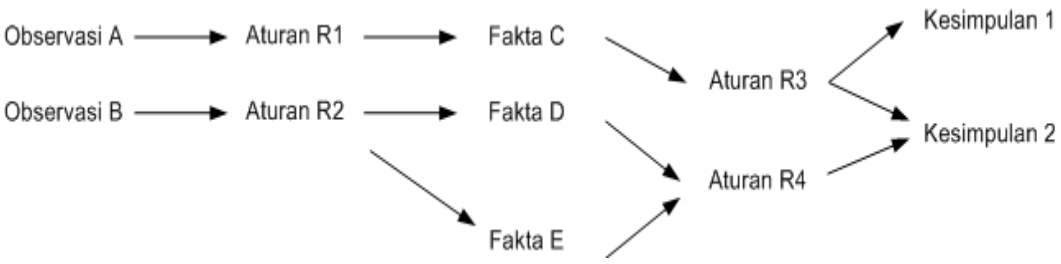
Terdapat dua pendekatan untuk mengontrol inferensi dalam sistem pakar berbasis aturan, yaitu pelacakan kebelakang (*backward chaining*) dan pelacakan kedepan (*forward chaining*). Pelacakan kebelakang adalah pendekatan yang dimotori tujuan (*goal-driven*). Dalam pendekatan ini, pelacakan dimulai dari

tujuan, selanjutnya dicari aturan yang memiliki tujujuan tersebut untuk kesimpulannya. Selanjutnya proses pelacakan menggunakan premis untuk aturan tersebut sebagai tujuan baru dan mencari aturan lain dengan tujuan baru sebagai kesimpulannya. Proses berlanjut sampai semua kemungkinan ditemukan. Gambar 2.2 menunjukan proses pelacakan kebelakang (*backward chaining*).



Gambar 2.2 Proses *Backward Chaining*

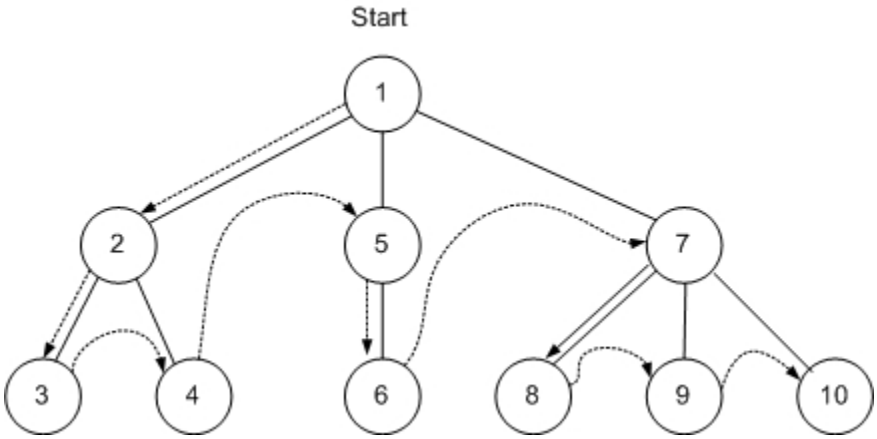
Pelacakan kedepan (*forward chaining*) adalah pendekatan yang dimotori oleh data (*data-driven*). Dalam pendekatan ini pelacakan dimulai dari informasi masukan, dan selanjutnya mencoba menggambarkan kesimpulan. Pelacakan kedepan mencari fakta yang sesuai dengan bagian *IF* dari aturan *IF-THEN*. Gambar 2.3 menunjukkan proses *forward chaining*.



Gambar 2.3 Proses *Forward Chaining*

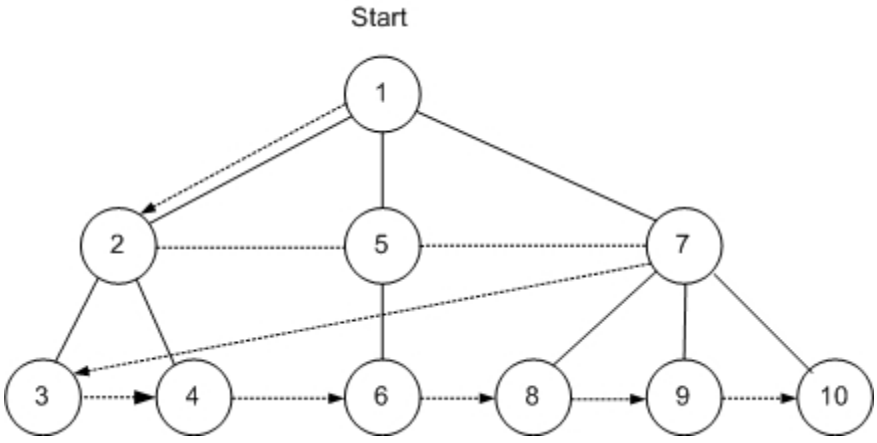
Kedua metode inferensi tersebut dipengaruhi oleh dua macam penelusuran, yaitu *depth first search* dan *breadth first search*.

Depth first search melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ketingkat dalam yang berurutan. Gambar 2.4 dibawah ini menunjukkan metode penelusuran depth first search.



Gambar 2.4 Teknik Penelusuran *Depth First Search*

Breadth first search bergerak dari simpul akar, simpul yang ada pada setiap tingkat diuji sebelum pindah ketingkat selanjutnya. Gambar 2.5 dibawah ini menunjukkan metode penelusuran *breadth first search*.



Gambar 2.5 Teknik Penelusuran *Breadth First Search*

Dalam memilih apakah akan menggunakan pelacakan kedepan atau pelacakan kebelakang, semuanya bergantung masalah yang akan tetap dibuat

sistem pakarnya, dan belum dapat dibuktikan mana yang lebih baik diantara kedua metode inferensi ini.

2.2.8.4 Antarmuka Pengguna

Antamuka pengguna atau *user interface* merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi. *User interface* atau antamuka pengguna menerima informasi dari pemakai dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem. Selain itu, antarmuka pengguna menerima informasi dari sistem dan menyajikannya kedalam bentuk yang dapat dimengerti oleh pemakai. Karena itu, perancangan *user interface* haruslah memenuhi prinsip *user friendly*, yaitu mudah digunakan oleh user dan sejelas mungkin agat tidak terjadi adanya salah interpretasi.

Sistem pakar seringkali interaktif, yaitu saling tukar informasi dan konklusi dengan seorang pemakai dalam suatu dialog. Penampilan kecerdasannya diperbaharui melalui kenyataan bahwa program dapat menerima masukan bentuk bebas dalam kalimat-kalimat *user* yang sederhana dan dapat menyatakan konklusi-konklusinya dengan cara yang sama. Suatu program yang dapat melakukan ini dikatakan memiliki suatu tatap muka bahasa natural.

2.2.8.5 Workplace

Workplace merupakan area dari sekumpulan memori kerja (*working memory*). *Workplace* digunakan untuk merekam hasil-hasil antara dan kesimpulan yang dicapai. Ada tiga tipe keputusan yang dapat direkam, yaitu :

1. Rencana : bagaimana menghadapi masalah.

2. Agenda : aksi-aksi yang potensial yang sedang menunggu untuk dieksekusi.
3. Solusi : calon aksi yang akan dibangkitkan.

2.2.8.6 Fasilitas Penjelasan

Fasilitas penjelasan adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar. Komponen ini menggambarkan penalaran sistem kepada pemakai. Fasilitas penjelasan dapat menjelaskan perilaku sistem pakar dengan menjawab pertanyaan-pertanyaan sebagai berikut :

1. Mengapa pertanyaan tertentu ditanyakan oleh sistem pakar ?
2. Bagaimana kesimpulan tertentu dipeoleh ?
3. Mengapa alternatif tertentu ditolak ?

2.2.8.7 Perbaikan Pengetahuan

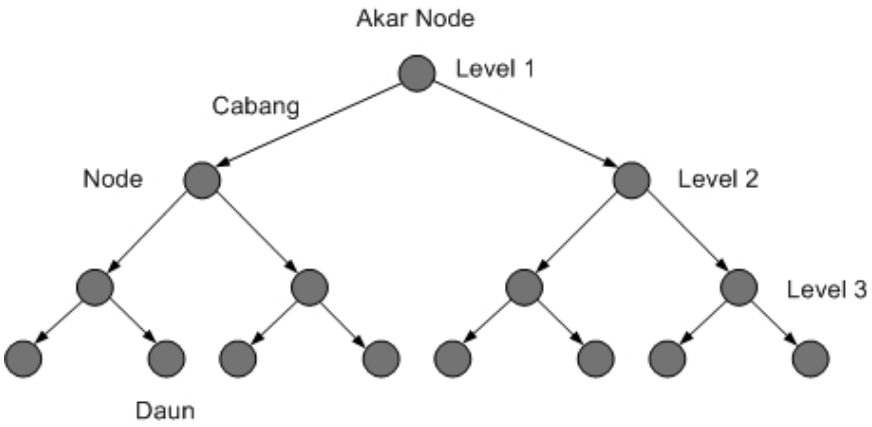
Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerja serta kemampuan untuk belajar dari kinerjanya. Kemampuan tersebut adalah penting dalam pembelajaran terkomputerisasi, sehingga program akan mampu menganalisis penyebab kesuksesan dan kegagalan yang dialaminya.

2.2.9 Pohon

Suatu pohon (*tree*) adalah suatu hierarki struktur yang terdiri dari simpul (*node)* yang menyimpan informasi atau pengetahuan dan cabang yang menghubungkan simpul. Cabang disebut juga *link* atau *edge* dan simpul disebut juga dengan *vertex*. Akar simpul adalah simpul yang tertinggi dalam hierarki dan daun adalah simpul yang paling bawah. *Tree* dapat dianggap sebagai suatu tipe

khusus dari jaringan semantik yang setiap simpulnya, kecuali akar pasti mempunyai satu simpul orang tua dan mempunyai nol atau lebih simpul anak.

Untuk tipe biasa dari *binary tree*, maksimum mempunyai dua anak untuk setiap simpul, dan sisi kiri dan kanan dari simpul anak dibedakan. Jika simpul mempunyai lebih dari satu orang tua maka disebut dengan jaringan. Gambar 2.6 menunjukkan *binary tree* yang mempunyai 0, 1 atau 2 cabang per simpul.



Gambar 2.6 *Binary Tree*

Aplikasi dari *tree* adalah pembuatan keputusan dan biasa disebut dengan istilah *decision tree* (pohon keputusan). Contoh *decision tree* untuk klasifikasi hewan seperti yang ditunjukkan pada gambar 2.7. Contoh ini merupakan permainan klasik dari teka-teki tentang hewan. Simpul berisikan pertanyaan, cabangnya berisi jawaban ya atau tidak untuk menjawab pertanyaan dan daunnya berisikan perkiraan hewan apa yang dijadikan kesimpulan.



Gambar 2.7 *Decision Tree* Klasifikasi Hewan

Struktur keputusan dapat diterjemahkan secara mekanis kedalam kaidah produksi (*production rule*) dengan menggunakan aturan *IF-THEN Rules*.

2.2.10 *IF-THEN Rules*

Dari pohon keputusan (*decision tree*) yang telah tereduksi, dapat dikonversikan menjadi *IF-THEN rule*. Setiap *rule* pada *decision tree* akan membentuk satu set rule final. Struktur dan sintaks penulisan rule adalah sebagai berikut :

RULE Label	:	Label berisi nama *rule* tersebut
IF	:	Sebagai penanda awal kondisi pada sebuah *rule*
THEN	:	Sebagai penanda awal kesimpulan pada sebuah *rule*
ELSE	:	Sebagai penanda awal alternatif kesimpulan pada sebuah *rule*, bersifat opsional, jadi boleh tidak ada.

Sedangkan operator yang dapat digunakan pada *IF-THEN rule* adalah :

AND : Semua kondisi yang dihubungkan oleh operator ini harus bernilai benar, agar kondisi keseluruhan *rule* tersebut bernilai benar. Bila ada satu kondisi yang bernilai salah, maka *rule* tersebut bernilai salah.

OR : Bila semua kondisi yang dihubungkan oleh operator ini harus bernilai salah, maka kondisi keseluruhan rule tersebut bernilai salah. Bila ada satu atau lebih kondisi yang bernilai benar, keseluruhan *rule* tersebut bernilai benar.

Untuk gambar 2.7 diatas, maka *decision tree* dapat diterjemahkan kedalam aturan berikut :

IF pertanyaan = "Apakah dia besar sekali ?"
And jawaban = "Tidak"
THEN pertanyaan = "Apakah dia mencicit"

IF pertanyaan = "Apakah dia besar sekali ?"
And jawaban = "Ya"
THEN pertanyaan = "Apakah dia mempunyai leher yang panjang ?"

IF pertanyaan = "Apakah dia mempunyai leher yang panjang ?"
And jawaban = "Ya"
THEN kesimpulan = "Jerapah"

2.2.11 Ketidakyakinan

Dalam penyusunan *rule*, *user* sering kali dihadapkan pada suatu premis yang meminta suatu kebenaran yang absolut. Sedangkan pada kenyataannya sering kali ditemui fakta yang kebenarannya tidak bersifat absolut. Untuk itu perlu ditentukan tingkat ketidakyakinan atau *uncertainty* terhadap suatu fakta. Misalnya pada sebuah pertanyaan, "Apakah pasien menderita nyeri pada perut?", untuk memasukan suatu *value* kedalam *inference engine*, user terkadang merasa bingung, rasa nyeri yang dimaksudkan adalah rasa nyeri yang bagaimana, apakah nyeri sedikit saja sudah dapat dimasukkan sebagai *value*, atau nyeri yang hebat saja yang dimasukan sebagai *value*. Ataukah dalam *rule* tersebut rasa nyeri dapat diabaikan saha. Bisa saja dibuat pertanyaan pelengkap "Nyatakan intensitas nyeri yang dirasakan dalam skala 0 sampai 10". Tetapi 2 orang yang berbeda dengan kasus yang sama bisa saja memberikan dua jawaban yang berbeda pula. Untuk itu diperlukan sebuah metode ketidakpastian atau *uncertainty* untuk memberikan toleransi jawaban atau pertanyaan yang jawabannya tidak pasti. Tingkat ketidakpastian ditentukan dari validitas *knowledge base rule* dan validitas respon dari *user*. Ada 4 cara pendekatan dalam implementasi teori ketidakpastian yaitu :

1. *Bayesian Probability* : merupakan salah satu cara untuk mengatasi ketidakpastian data dengan menggunakan formula Bayes yang dinyatakan dengan menggunakan rumus :

$$P(H \mid E) = \frac{P(E \mid H)x(P(H)}{P(E)}$$

Dimana :

P(H|E) : Probabilitas hipotesis H jika diberikan *evidence* E.

P(H|E) : Probabilitas munculnya *evidence* E jika diketahui hipetosesis H.

P(H) : Porabilitas hipotesis H tanpa memandang *evidence* apapun.

P(E) : Probabilitas *evidece* E.

2. EXSYS : ada tiga cara untuk menyatakan ketidakpastian yaitu dengan
   a. Sistem 0 atau 1, yaitu hanya ada dua kemungkinan yaitu ya dan tidak.
   b. Sistem 0 sampai 10, yaitu menyatakan ketidakpastian dengan *confidence factor* dari 0 sampai 10. Apabila *confidence factor* suatu *rule* final berasal dari *confidence factor rule-rule* lain, maka kesimpulan *rule* final tersebut memiliki *confidence factor* berupa rata-rata *confidence factor rule-rule* tersebut.
   c. Sistem -100 sampai 100, yaitu menyatakan ketidakpastian dengan *confidence factor* -100 sampai 100. Apabila *confidence factor* bernilai -100, maka *value* suatu premis diyakini pasti salah. Apabila *confidence factor* bernilai 100 berarti kebenaran suatu premis diyakini pasti benar. Apabila *confidence factor* bernilai 0 berarti kebenaran suatu premis dianggap masih ragu-ragu.
3. *Fuzzy Set* : Tidak seperti dua pendekatan diatas, *fuzzy set* beranggapan bahwa ketidakpastian sebagai konsekuensi *input* dari *user*. *Confidence factor* yang digunakan dalam metode ini berkisar antara 0 sampai 1. Untuk menentukan *fuzzy value* dari kesimpulan suatu *rule*, maka diambil nilai

fuzzy value minimum dari semua premis yang ada kemudian dikalikan dengan *confidence factor rule* tersebut apabila premisnya dihubungkan dengan operator AND. Sedangkan apabila premisnya dihubungkan dengan operator OR, maka diambil nilai *fuzzy value* maksimum dari semua premis yang ada kemudian dikalikan dengan *confidence factor rule* tersebut. Sebagai contoh adalah :

Rule 1 :

If disk drive berisik = *yes* (*fuzzy value*=0,8)

And menghasilkan *bad sector* bila memformat disket = *yes* (*fuzzy value* = 0,3)

Then status disk drive = rusak (*confidence factor* = 0,9)

Maka tingkat keyakinan akan hasil kesimpulan *rule* tersebut adalah

min(0,8 ; 0,3) x 0,9

0,3 x 0,9 = 0,27

Rule 2 :

If disk drive berisik = *yes* (*fuzzy value*=0,9)

Or menghasilkan *bad sector* bila memformat disket = *yes* (*fuzzy value* = 0,2)

Then status disk drive = rusak (*confidence factor* = 0,7)

Maka tingkat keyakinan akan hasil kesimpulan rule tersebut adalah

max(0,9 ; 0,2) x 0,7

0,9 x 0,7 = 0,63

4. Metode *Confidence Factor Union*

   Dalam proses pengeksekusian *knowledge base*, nantinya kita akan dihadapkan dalam *multiple value* yang mengacu pada satu kesimpulan yang sama. Misalkan pada penggabungan dua *value*, *value* pertama memiliki *confidence factor* sebesar 0,2 dan *value* kedua memiliki *confidence factor* sebesar 0,5 maka *confidence factor* gabungan kedua *value* tersebut adalah :

   C(cf) = (0,2 + 0,5) – (0,2 x 0,5) = 0,6.

2.3 Konsep Dasar Analisis Sistem

Analisis sistem dapat didefinisikan sebagai penguraian dari suatu sistem informasi dengan maksud untuk mengidentifikasikan dan mengevaluasikan permasalahan-permasalahan, hambatan-hambatan yang terjadi dalam kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

Didalam tahap analisis sistem terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem sebagai berikut :

1. *Identify*, yaitu mengidentifikasi masalah.
2. *Understand*, yaitu memahami kerja dari sistem yang ada.
3. *Analyze*, yaitu menganalisa sistem.
4. *Report*, yaitu membuat laporan hasil analisis.

Model analisis merupakan representasi teknis yang pertama dari sistem, saat ini ada dua yang mendominasi pemodelan analisis, yaitu :

1. Analisis Terstruktur. Dengan menggunakan notasi yang sesuai dengan prinsip analisis operasional, kita menciptakan model yang menggambarkan

muatan dan aliran inforasi, kita membagi sistem secara fungsional dan secara behavior, dan menggambarkan esensi dari apa yang harus dibangun.

2. Analisis Berorientasi Objek, mendefinisikan semua kelas yang relevan terhadap masalah berserta operasi-operasi dan atribut-atribut yang diasosiasikan dengan kelas itu, keterhubungan dikelas-kelas dan perilaku yang dimilikinya..

Alat-alat yang digunakan dalam analisis terstruktur adalah :

1. Diagram E-R (*Entity Relationship Diagram*)
2. Diagram Konteks (*Context Diagram*)
3. Diagram arus data (*Data Flow Diagram*)

2.3.1 Diagram E-R (*Entity Relationship Diagram*)

Diagram E-R adalah diagram grafikal yang menggambarkan keseluruhan struktur *logic* dari sebuah basis data. Pada model ini semua data yang ada pada dunia nyata diterjemahkan atau ditransformasikan dengan memanfaatkan sejumlah perangkat konseptual menjadi sebuah diagram data.

Untuk melambangkan fungsi diatas maka digunakan simbol-simbol yang memiliki arti sebagai berikut :

a. Data Entitas

Entitas merupakan individu yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Data entitas adalah segala sesuatu yang ntaya maupun abstrak yang datanya akan direkam.

b. Atribut

Atribut mendefinisikan karakteristik (*property*) dari entitas.

c. Relasi Data

Relasi menujukkan adanya hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda.

d. Kardinalitas

Kardinalitas adalah penentuan jumlah kejadian dari suatu entitas yang dapat terhubung dengan sejumlah kejadian pada entitas lain, juga menunjukkan jumlah maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain. Terdapat tiga kemungkinan kombinasi kardinalitas :

1. *Relationship one to one* (relasi satu ke satu)

   Yang artinya bahwa setiap entitas pada himpunan entitas yang satu berhubungan dengan paling banyak satu entitas pada himpunan entitas yang lainnya, begitu juga sebaliknya, entitas yang terhubung hanya memiliki satu hubungan dengan entitas yang menghubungi.

2. *Relationship one to many* (relasi satu ke banyak)

   Yang artinya bahwa setiap entitas pada himpunan A dapat berhubungan dengan lebih dari satu entitas pada himpunan entitas B, tetapi tidak berlaku untuk kebalikannya, dimana entitas B hanya dapat berhubungan dengan paling banyak satu entitas pada himpunan A.

3. *Relationship many to many* (relasi banyak ke banyak)

   Yang artinya bahwa setiap entitas pada himpunan A dapat berhubungan dengan lebih dari satu entitas pada himpunan B,

begitu juga sebaliknya dimana entitas B dapat berhubungan dengan lebih dari satu entitas pada himpunan A.

2.3.2 Diagram Konteks

Diagram konteks adalah arus data yang berfungsi untuk menggambarkan keterkaitan aliran-aliran data antara sistem dengan bagian-bagian luar. Kesatuan luar ini merupakan sumber arus data atau tujuan data yang berhubungan dengan sistem informasi tersebut. Diagram konteks memberikan batasan yang jelas mengenai besaran-besaran entitas yang berada diluar sistem yang sedang dibuat, artinya diagram ini menggambarkan secara jelas batasan-batasan dari sebuah sistem yang sedang dibuat.

2.3.3 Diagram Alir Data (*Data Flow Diagram*)

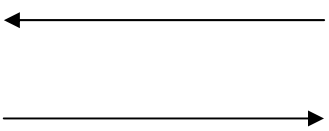
Merupakan representasi grafik dari suatu sistem yang menunjukkan proses atau fungsi, aliran data, tempat penyimpanan data dan entitas eksternal. *Data Flow Diagram* (DFD) juga digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan. Beberapa simbol yang digunakan di DFD adalah :

1. *External Entity*, merupakan kesatuan dilingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lainnya yang berada dilingkungan luarnya yang akan memberikan input atau memeriksa *output* dari sistem. Disimbolkan dengan suatu notasi kotak.



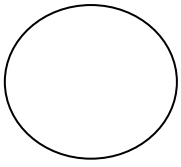
Gambar 2.8 Kesatuan Luar (*External Entity*)

2. *Data Flow* (arus data), di DFD diberi simbol suatu panah. Arus data ini mengalir diantara proses (*process*), simpanan data (*data store*) dan kesatuan luar (*external entity*). Arus data ini menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem.



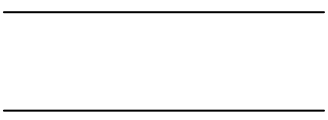
Gambar 2.9 Arus Data (*Data Flow*)

3. *Process* (proses). Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus dara yang masuk kedalam proses untuk dihasilkan arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.



Gambar 2.10 Proses (*Process*)

4. *Data Store* (simpanan data), merupakan simpanan dari data yang dapat berupa suatu file atau *database*, suatu arsip atau catatan manual.



Gambar 2.11 Penyimpanan Data (*Data Storage*)

2.3.4 Kamus Data

Salah satu komponen kunci dalam sistem manajemen *database* adalah *file* khusus yang disebut kamus data (*data dictionary*). Kamus data merupakan katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus data, analisis sistem dapat mendefinisikan data yang mengalir berisi informasi tentang struktur *database*. Untuk setiap elemen data yang disimpan dalam *database* seperti nomor pokok pegawai, diuraikan secara lengkap mulai dari nama, tempat penyimpanan, program komputer yang berhubungan dan lain-lain.

Kamus data biasanya dipelihara secara otomatis oleh sistem manajemen *database.* Cara mendefinisikan kamus data yaitu :

a. Menggambarkan arti aliran data atau penyimpanan yang ditunjuk dalam DFD.
b. Menggabungkan komponen dari kumpulan data yang mengalir yaitu kumpulan komponen yang mungkin bisa dipecah lagi menjadi data elementer.
c. Menggambarkan data yang tersimpan.
d. Menentukan nilai dibagian elementer dari informasi yang relevan di DFD dan data *store*nya.

2.4 Konsep Basis Data

Basis data adalah kumpulan *file* yang saling berinteraksi, relasi tersebut bisa ditunjukan dengan kunci dari tiap *file* yang ada. Satu basis data menunjukan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi.

2.4.1 Pengertian Basis Data

Basis data adalah kumpulan dari data yang saling berhubungan antara yang satu dengan yang lain, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Basis data merupakan salah satu komponen yang penting dalam sistem informasi, karena merupakan basis dalam menyediakan informasi bagi para pemakai.

2.4.2 Desain Basis Data

Penerapan basis data dalam sistem informasi disebut sistem basis data (*database system*). Dengan sistem dasar *database* ini tiap-tiap orang atau bagian dapat memandang *database* dari berbagai sudut pandang yang berbeda. Dalam *database* terdiri dari *entity*, *atribut*, data *value, record* dan *file.*

Perancangan *database* yang digunakan adalah untuk memudahkan dalam mengetahui file-file *database* yang digunakan dalam perancangan sistem, sekaligus untuk mengetahui hubungan antar *file* dari *database* tersebut. Terdapat beberapa kriteria yang harus dipenuhi dalam sistem basis data yang harus ada adalah sebagai berikut :

1. Menyimpan seluruh data dan informasi secara terpusat.
2. Mengurangi redundansi data atau duplikasi data.

3. Melakukan perubahan-perubahan data untuk menyelesaikan dan untuk pengembangan yang akan datang.
4. Menjamin keamanan data.
5. Menjamin integritas data.

2.4.3 *Entity*

Entity adalah orang, tempat kejadian atau konsep yang dapat dibedakan dengan objek yang lain.

2.4.4 Atribut

Setiap *entity* mempunyai atribut atau sebutan untuk mewakili suatu *entity*. Misalnya gedung sekolah atributnya adalah nama gedung sekolah, alamat, kondisi fisik, dan lain-lain.

2.4.5 *Record*

Kumpulan elemen-elemen yang saling berkaitan menginformasikan mengenai suatu *entity* secara lengkap. Suatu *record* mewakili suatu data atau informasi tentang gedung sekolah, misalnya nama gedung sekolah, alamat, kondisi fisik, dan lain-lain.

2.4.6 *File*

File merupakan kumpulan *record-record* sejenis yang mempunyai panjang elemen yang sama, atribut yang sama, *record* yang sama namun berbeda data *value* dan saling berhubungan. *File* dibuat bila diperlukan suatu kumpulan *record*

tertentu. Data dikumpulkan dari sumber dokumen yang baru atau mengambil *record* dari *file* yang sudah ada dan disusun kembali ke dalam *file* baru.

2.5 Sekilas Tentang Borland Delphi

Adanya perkembangan bahasa pemrograman yang semakin pesat dan ilmu pengetahuan serta teknologi yang semakin canggih, banyak perusahaan besar *software* yang mengembangkan bahasa pemrograman menjadi sebuah *software* yaitu pemrograman yang memberikan kemudahan kepada pengguna *software* untuk mendesain sebuah aplikasi yang berorientasi pada *database* maupun *server*. Salah satu bahasa pemrograman yaitu pemrograman Borland Delphi yang telah meluncurkan berbagai versi yang menawarkan kemudahan dengan fasilitas yang lengkap. Karena Delphi merupakan generasi lanjutan dari Turbo Pascal yang telah terkenal sebagai bahasa pemrograman yang terstruktur yang diluncurkan oleh Borland International Incorporation pada tahun 1983. Sesuai dengan sistem operasi yang popular pada saat itu Turbo Pascal memang dirancang untuk dijalankan pada system operasi DOS. Seiring dengan perkembangan zaman dan ilmu pengetahuan Borland International Incorporation meliris Turbo Pascal *for* Windows, dalam versi berjalan pada Windows 3.X.

Penggabungan Turbo Pascal dengan Turbo Pascal *for* Windows pada tahun 1992 melahirkan suatu bahasa pemrograman yang diberi nama Borland Pascal versi 7.0. tetapi masih terdengar banyak keluhannya mengenai cara penggunaan bahasa pemrograman tersebut, sehingga mendorong pihak Borland International Incorporation untuk membuat bahasa pemrograman visual yang berbasis bahasa Pascal, maka lahirlah Borland Delphi versi 2.0.

Pada tahun 1997 lahirlah Borland Delphi versi 3.0 yang mempunyai kemampuan yang semakin bertambah terutama dalam merancang aplikasi untuk *multiuser* dan *database* yang semakin canggih. Dua tahun kemudian diluncurkan kembali Borland Delphi versi 4.0 yang semakin canggih dari versi sebelumnya. Tetapi Borland Delphi 4.0 masih perlu dikembangkan kembali agar mempermudah dalam pengoperasiannya.

Meskipun begitu Borland Delphi memiliki kelebihan dibanding dengan aplikasi pemorgraman visual berbasis *windows* yang lain, diantaranya:

1. Borland Delphi menyediakan fasilitas yang luas mulai dari fungsi membuat form hingga untuk menggunakan format file berbasis data yang popular seperti *dBase* dengan *paradoks*.
2. Dalam Borland Delphi *template* aplikasi dan *form* yang dapat dipakai untuk membuat semua aplikasi dengan lebih cepat.
3. Borland Delphi dapat diatur sesuai dengan kebutuhan dan menunjukan bagaimana memakai Borland Delphi dilingkungan kerja agar pekerjaan lebih produktif.

2.6 Konsep Dasar Penyakit

Penyakit adalah suatu keadaan abnormal dari tubuh atau pikiran yang dapat menyebabkan ketidaknyamanan, disfungsi atau kesukaran terhadap orang yang dipengaruhinya. Penyakit dari segi karakteristiknya dapat dibedakan menjadi dua, yaitu penyakit menular dan penyakit tidak menular.

Penyakit menular merupakan penyakit yang disebabkan oleh kuman yang menjangkiti tubuh manusia. Kuman dapat berupa virus, bakteri atau jamur.

Penyakit tidak menular merupakan penyakit yang tidak disebabkan oleh kuman, tetapi disebabkan karena adanya masalah fisiologis atau metabolisme pada jaringan tubuh manusia.

Untuk mendiagnosa suatu penyakit perlu diketahui dan dipelajari dahulu gejala-gejala yang ditimbulkan. Meskipun dari gejala-gejala yang terlihat langsung maupun yang dirasakan penderita, dapat diambil suatu kesimpulan penyakit yang diderita. Tetapi ada kalanya gejala-gejala yang ada tidak mengarah tepat kepenyakit tertentu, untuk itu perlu dilakukan pemeriksaan lebih mendalam, misalnya tes *urine*, tes darah, USG, atau pemeriksaan sinar X (*rontgen*).

2.6.1 Diagnosa Penyakit oleh Dokter

Dalam menganalisa kondisi pasien, seorang dokter pertama kali melakukan anamnesis atau *medical history talking* atau lazim disebut penelusuran riwayat kesehatan. Anamnesis merupakan cara untuk mendapatkan keterangan dan data-data klinis tentang keadaan penyakit pasien melalui tanya jawab lisan atas keluhan atau gejala-gejala yang dirasakan serta keterangan lain yang dialami oleh pasien tersebut. Sistem anamnesis biasanya terdiri dari 5 tahap, yaitu :

1. Keluhan Utama

   Keluan utama yang menimbulkan perasaan dan pikiran pada penderita sehingga datang untuk meminta pertolongan medis.

2. Keluhan Tambahan

   Keterangan yang menjelaskan keadaan klinis penderita baik yang ada hubungan dengan kelainan utama ataupun perihal lainnya yang menganggu kesehatan penderita saat ini.

3. Riwayat Kesehatan

   Menyangkut riwayat penyakit terdahulu dan kebiasaan yang ada hubungan dengan penyakit.

4. Riwayat Keluarga

   Keterangan atau data mengenai kehidupan bermasyarakat dan riwayat perkawinan untuk mencari apakah ada faktor *heredomafilial* yang mungkin merupakan faktor predisposisi.

5. Riwayat Sosial Ekonomi

   Keterangan atau data mengenai kehidupan bermasyarakat dan riwayat pekerjaan serta status ekonomi penderita, termasuk juga dalam hal ini suasana dan lingkungan hidup dalam rumah.

2.7 Konsep Dasar Pengobatan Tradisional

Sejak lama manusia menggunakan tanaman dan bahan alam lain sebagai obat untuk mengurangi rasa sakit, menyembuhkan dan mencegah penyakit tertentu serta untuk menjaga kondisi badan agar tetap sehat dan bugar. Hingga saat ini penggunaan tumbuhan atau bahan alam sebagai obat tersebut dikenal dengan sebutan obat tradisional.

Menurut definisi dari Departemen Kesehatan RI, yang dimaksud dengan obat tradisional adalah obat jadi atau ramuan bahan alam yang berasal dari tumbuhan, hewan, mineral atau campuran bahan tersebut yang secara tradisional telah digunakan untuk pengobatan berdasarkan pengalaman. Pada dasarnya bahan obat tradisional yang berasal dari tumbuhan porsinya lebih besar dibandingkan dengan yang berasal dari hewan atau mineral, sehingga sebutan untuk obat

tradisional hampir selalu identik dengan tanaman obat karena sebagian besar obat tradisional bahan bakunya berasal dari tanaman obat.

Sedangkan yang dimaksud dengan tanaman obat adalah tanaman yang berkhasiat obat atau tanaman yang mengandung senyawa kimia yang dapat digunakan untuk pengobatan. Setiap tanaman obat memiliki berbagai jenis kandungan senyawa kimia, ada yang berkhasiat untuk pengobatan, yang memiliki efek farmakologis dan adapula yang tidak digunakan untuk pengobatan. Contohnya untuk bahan baku minyak wangi. Adapula yang bersifat racun. Beberapa contoh tanaman yang berkhasiat untuk pengobatan diantaranya adalah :

1. Tanaman Tapak Dara

   Dari akar, batang daun dan biji ditemukan lebih dari 70 macam *alkaloid* yang dapat digunakan untuk menurunkan kadar gula darah dan sebagai obat anti kanker.

2. Tanaman Sesuru

   Batang tanaman ini mengandung asam amino, asam sitrat dan asam malat. Zat asam tersebut beracun, namun batang tanaman ini digunakan untuk bisul atau luka gigitan ular.

3. Rambutan

   Kulit buah berkhasiat sebagai penurun panas, sedangkan biji berkhasiat untuk menurunkan kadar gula darah.

2.7.1 Sifat dan Ciri-Ciri Tanaman Obat

Ada empat macam sifat dan lima macam ciri rasa dari tanaman obat yang merupakan suatu bagian dari cara pengobatan tradisional timur. Keempat macam

sifat dari tanaman obat yaitu dingin, panas, hangat dan sejuk. Tanaman obat yang bersifat panas dan hangat, dipakai untuk pengobatan pada sindroma dingin, misalnya kaki dan tangan dingin, lidah pucat dan lain-lain. Sedangkan sifat sejuk dan dingin dari tanaman obat dipakai untuk pengobatan pada sindroma panas, misalnya demam, rasa haus, lidah merah dan sebagainya.

Lima macam ciri rasa dari tanaman obat yaitu rasa pedas, manis, masam, pahit dan asin, yang penggunaannya mempunyai khasiat berbeda-beda. Rasa pedas bersifat menyebarkan dan berefek merangsang, rasa manis bersifat menguatkan, rasa asam bersifat mengawetkan, rasa pahit bersifat menghilangkan panas dan lembab serta rasa asin bersifat melunakkan dan pencahar.