BAB II
TINJAUAN PUSTAKA

Tinjauan pustaka merupakan kumpulan dasar teori yang digunakan untuk menunjang kelancaran pelaksanaan penelitian. Disamping itu, tinjauan pustaka juga dapat berguna sebagai pedoman dalam penelitian sehingga pelaksanaan penelitian dapat terfokus pada tujuan yang akan dicapai.

2.1 Penelitian Terdahulu

Terdapat beberapa penelitian yang telah dilakukan mengenai pengembangan dan rekayasa sistem basis data yang menjadi referensi dalam melakukan penelitian ini. Berikut adalah beberapa hasil penelitian terdahulu:

1. Radiana (2010), dalam penelitiannya ini melakukan pengembangan sistem pakar *Troubleshooting* kerusakan *hardware* komputer berbasis *web* dalam bentuk *prototype*. *Prototype* yang dikembangkan menggunakan *expert system* dengan metode *forward chaining*, digunakan untuk membantu dalam hal mencari kerusakan *hardware* komputer guna menekan biaya perbaikan komputer. *Software* yang digunakan dalam aplikasi ini adalah *Dreamweaver CS3* dan *MySQL*. Berdasarkan hasil penelitian, diperoleh bahwa dengan adanya aplikasi Sistem Pakar *troubleshooting* kerusakan *hardware* komputer membantu mencari kerusakan *hardware* komputer dengan cepat juga meminimalisasi pengeluaran uang untuk memperbaiki kerusakan *hardware*.
2. Asri (2010), dalam penelitiannya ini melakukan pengembangan integrasi sistem informasi pengendalian kualitas dengan *expert system* untuk menelusuri penyebab cacat produk. *Prototype* yang dikembangkan menggunakan *expert system* dengan *inference engine Visual Basic for Application* sebagai metode untuk menelusuri kecacatan produk yang terjadi. Berdasarkan hasil penelitian, diperoleh bahwa dengan adanya SIPKES sangat membantu divisi produksi dalam perusahaan mendapatkan solusi tentang cacat produk secara cepat dan tepat. Selain itu, sistem juga dapat memberi informasi mengenai proses kritis yang perlu perbaikan dan hubungan sebab alibat cacat produk melalui visualisasi *fishbone diagram*.
3. Andryana (2009), dalam penelitiannya ini melakukan pengembangan deteksi kerusakan *notebook* dengan menggunakan metode sistem pakar. *Prototype* yang

dikembangkan menggunakan *expert system* dengan metode *certainty factor* digunakan untuk membantu mendeteksi kerusakan *notebook* dengan jenis kerusakan LCD, *Motherboard*, *Hardisk*, Fdd, CD/DVD/CDRW/DVDRAM, *Keyboard*, *Modem*, *Ethernet*, *Processor*, *bluetooth*, *mouse*, baterai dan lain sebagainya. Salah satu kelebihan dari sistem pakar diagnosa kerusakan *notebook* tidak membatasi penggunaan sistem, selain itu proses pemeriksaan kerusakan *notebook* mempertimbangkan munculnya gejala khas paska setiap kerusakan, sehingga menyebabkan proses pendiagnosaan memakan waktu yang relatif singkat dan tepat. Berdasarkan hasil penelitian, diperoleh bahwa dengan adanya deteksi kerusakan *notebook* dengan menggunakan metode sistem pakar dapat membantu pengguna dalam mengidentifikasi seluruh kerusakan pada notebooks sejak dini. Ini agar setiap kerusakan yang terjadi pada notebook dapat diselesaikan secara cepat dan tepat. Sehinga mengurangi kesulitan yang ditimbulkan sebagai akibat dari rusaknya notebook.

Tabel 2.1 Penelitian Terdahulu

	Metode	**Hasil Penelitian**
Radiana (2010)	*Expert System* dengan metode *Forward Chaining*	Membantu mencari kerusakan *hardware* komputer dengan cepat juga meminimalisasi pengeluaran uang untuk memperbaiki kerusakan *hardware*
Asri (2010)	*Expert System* dengan *database*	Membantu divisi produksi dalam perusahaan mendapatkan solusi tentang cacat produk secara cepat dan tepat, memberi informasi mengenai proses kritis yang perlu perbaikan dan hubungan sebab alibat cacat produk melalui visualisasi *fishbone diagram*.
Andryana (2009)	*Expert System* dengan metode *Certainty Factor*	Membantu pengguna dalam mengidentifikasi seluruh kerusakan pada notebooks sejak dini

2.2 Maintenance

Perawatan atau lebih dikenal dengan kata *maintenance* dapat didefinisikan sebagai suatu aktivitas yang diperlukan untuk menjaga atau mempertahankan kualitas pemeliharaan suatu fasilitas agar fasilitas tersebut tetap dapat berfungsi dengan baik dalam kondisi siap pakai. Sedangkan fasilitas yang dimaksudkan di sini seperti mesin,

generator, diesel turbin, dan utilitas pabrik lainnya, dan bahkan peralatan kantor. Sedangkan menurut Gasperz (1992), perawatan merupakan suatu kegiatan yang diarahkan pada tujuan untuk menjamin kelangsungan fungsional suatu sistem produksi, sehingga dari sistem diharapkan menghasilkan output sesuai yang dikehendaki. Menurut Handoko (1991) salah satu maksud utama kegiatan pemeliharaan adalah untuk memelihara reliabilitas sistem pengoperasian pada tingkat yang dapat diterima dan tetap memaksimumkan laba atau meminimumkan biaya. Tujuan perawatan pada umumnya adalah sebagai berikut (Mustafa, 1998):

1. Memungkinkan tercapainya mutu produk dan kepuasan pelanggan melalui penyesuaian, pelayanan, dan pengoperasian peralatan secara tepat.
2. Memaksimalkan umur kegunaan dari sistem.
3. Menjaga agar sistem aman dan mencegah berkembangnya gangguan keamanan.
4. Meminimalkan biaya produksi total yang secara langsung dapat dihubungkan dengan servis dan perbaikan.
5. Meminimalkan frekuensi dan kuatnya gangguan-gangguan terhadap proses operasi.
6. Memaksimalkan produksi dari sumbersumber sistem yang ada.
7. Menyiapkan personil, fasilitas, dan metodenya agar mampu mengerjakan tugas-tugas perawatan.

Umumnya strategi perawatan dibagi menjadi 3 strategi utama, yaitu strategi *corrective maintenance, preventive maintenance* dan *predictive maintenance* dengan penjelasan sebagai berikut:

1. *Corrective Maintenance*

   Menurut Assauri (1999), *corrective maintenance* merupakan kegiatan perawatan yang dilakukan setelah mesin atau fasilitas produksi mengalami kerusakan atau gangguan sehingga tidak dapat berfungsi dengan baik. Dalam hal ini, kegiatan *corrective maintenance* sering disebut dengan kegiatan reparasi atau perbaikan. *Corrective maintenance* biasanya tidak dapat direncanakan dahulu karena kegiatan ini menunggu sampai kerusakan mesin terjadi terlebih dahulu, kemudian baru diperbaiki agar dapat beroperasi kembali. Hal ini tentu mengakibatkan tingkat efektifitas pengoperasian mesin pada saat itu sangat rendah. *Corrective maintenance* merupakan studi yang digunakan dalam menentukan tindakan yang

diperlukan untuk mengatasi kerusakan-kerusakan. Tindakan perawatan ini bertujuan untuk mencegah terjadinya kerusakan yang sama. Prosedur ini ditetapkan pada peralatan atau mesin yang sewaktu-waktu dapat terjadi kerusakan. Pada umumnya usaha untuk mengatasi kerusakan dapat dilakukan dengan cara sebagai berikut:

Mencatat data kerusakan, kemudian meng-*improve* peralatan sehingga kerusakan yang sama tidak terjadi lagi.

a. *Improve* peralatan sehingga perawatan menjadi lebih mudah.
b. Merubah proses.
c. Merancang kembali komponen yang gagal
d. Mengganti dengan komponen yang baru
e. Meningkatkan prosedur perawatan preventif
f. Meninjau kembali dan merubah sistem pengoperasian

Tindakan *corrective maintenance* ini memerlukan biaya perawatan yang lebih murah daripada tindakan *preventive maintenance*. Hal ini dapat terjadi apabila kerusakan terjadi pada saat mesin atau fasilitas tidak melakukan proses produksi. Tetapi bila kerusakan terjadi selama proses produksi berlangsung maka biaya perawatan akan mengalami peningkatan yang disebabkan karena terhentinya proses produksi. Dengan demikian dapat ditarik kesimpulan bahwa tindakan *corrective maintenance* memusatkan permasalahan setelah permasalahan itu terjadi, bukan menganalisa masalah untuk mencegah agar tidak terjadi.

2. *Preventive Maintenance*

Sejak tahun 1950-an, muncul suatu metode perawatan mesin yang disebut sebagai *Preventive Maintenance*. Dalam metode ini, tindakan perawatan mesin dilakukan sebagai tindakan pencegahan, dalam arti perawatan mesin dilakukan sebelum mesin mengalami kerusakan. Dalam *preventive maintenance*, perawatan mesin dilakukan secara berkala, dengan memperhitungkan *designed life cycle* (siklus hidup desain) dari suatu peralatan. Perawatan dilakukan untuk memperpanjang siklus hidup dari tiap peralatan hingga maksimum. Sudut pandang dalam memandang *preventive maintenance* sangat mempengaruhi keberhasilan program *preventive maintenance*. *Preventive maintenance* dapat menyediakan keuntungan yang besar, dan jika dapat diaplikasikan dengan baik, dapat mencegah kegagalan,

mengurangi biaya dan *downtime*, dan meningkatkan *uptime*, produktifitas, dan keuntungan.

3. *Predictive Maintenance*

Perawatan *predictive* ini pun merupakan bagian perawatan pencegahan. Perawatan ini dapat diartikan sebagai strategi perawaatan dimana pelaksanaannya didasarkan kondisi mesin itu sendiri. Untuk menentukan kondisi mesin dilakukan tindakan pemeriksaan secara rutin, jika terdapat tanda atau gejala kerusakan segera diambil tindakan perbaikan untuk mencegah kerusakan lebih lanjut, jika tidak terdapat gejala kerusakan segera pula diketahui. Perawatan *predictive* disebut juga perawatan berdasarkan kondisi (*condition based maintenance*) atau juga disebut monitoring kondisi mesin (*machinery condition monitoring*), yang artinya sebagai penentuan kondisi mesin dengan cara memeriksa mesin secara rutin, sehingga dapat diketahui keandalan mesin serta keselamatan kerja terjamin. *Predictive maintenance* adalah pemantauan berkala terhadap kondisi mekanik yang sebenarnya, efisiensi operasi, dan indikator lain dari kondisi operasi mesin dan meminimalkan jumlah dan biaya oleh *failure* mesin. *Predictive maintenance* bertujuan meningkatkan produktivitas, kualitas dan keseluruhan efektivitas dari proses manufaktur. Sebenarnya *predictive maintenance* adalah *preventive maintenance* yang didasarkan oleh kondisi sesungguhnya dari mesin. Metode pelaksanaan pemeliharaan *predictive*:

a. Pemilihan Peralatan

Tidak perlu seluruh peralatan mesin dipelihara secara *predictive*, tetapi langkah yang lebih baik adalah memilih peralatan-peralatan yang kritis atau mahal, juga dipengaruhi oleh fungsi dan kondisi spesifik suatu peralatan misalnya: piston, *turbocharger*, *governor*, *jacket water pump*, *generator*, *compressor* dll.

b. Pengumpulan Data Sejarah Mesin

Riwayat mesin dapat dipakai sebagai pendekatan teknik pemantauan dan analisa pemeliharaan. Data/informasi tersebut dapat berupa: data desain, data sejarah mesin dan data sejarah operasi mesin lain yang sejenis (jika ada).

c. Pemasangan Alat-alat Sensor

Pemasangan alat-alat sensor pada bagian-bagian tertentu untuk dapat memantau kondisi peralatan sangat diperlukan. Pemantauan itu meliputi: vibrasi, temperatur, tekanan, laju aliran, korosi dan lain sebagainya.

d. Pemantauan Rutin

Pemantauan dilaksanakan ketika unit sedang beroperasi atau unit sedang stop, tergantung pada objek yang akan dipantau.

2.3 Troubleshooting

Menurut Mukhibudin (2007), *Troubleshooting* adalah suatu proses sistematik untuk mencari sumber penyebab masalah (kerusakan) sehingga bisa diselesaikan / diperbaiki. *Troubleshooting* seringkali merupakan suatu proses eliminasi berbagai hal yang berpotensi menyebabkan suatu permasalahan (kerusakan) terjadi. *Troubleshooting* diterapkan pada sesuatu yang tiba-tiba berhenti bekerja, untuk mencari penyebab terjadinya perubahan tersebut.

Troubleshooting bisa merupakan sebuah prosedur pengecekan, berbentuk sebuah alur atau sebuah tabel. Pengembangan prosedur tersebut dalam tingkat ahli bisa menghasilkan tahapan-tahapan pengecekan yang efisien dalam menemukan sumber penyebab permasalahan/kerusakan. Tabel prosedur *troubleshooting* bisa dikomputerisasi sehingga semakin memudahkan pemakai.

Ada dua tahapan utama dalam proses *troubleshooting*, yaitu:

1. Mengidentifikasi Masalah
2. Perbaikan

Dalam mengidentifikasi masalah ada beberapa tahapan, antara lain;

1. Pengumpulan Informasi

   Dilakukan dengan melakukan pengamatan pada obyek, dan pengumpulan informasi dari pemakai dengan mengajukan beberapa pertanyaan yang berkaitan. Dalam melakukan dialog dengan pemakai diusahakan menggunakan bahasa yang dimengerti dan mudah dipahami, bila diperlukan berikan penjelasan tentang subyek yang kita tanyakan.
2. Memastikan Permasalahan

   Merupakan tahapan yang penting dalam proses *troubleshooting*, dalam beberapa kasus akan sangat menghemat waktu, karena tidak membuang waktu untuk mengecek permasalahan yang keliru.

3. Memperluas Cakupan Permasalahan (jika diperlukan)

   Jika permasalahan tidak bisa diselesaikan dengan berbagai usaha yang telah dilakukan, bisa jadi perlu untuk memperluas cakupan permasalahan pada bagian yang lain. Lakukan diagnosa menyeluruh pada semua bagian yang berhubungan dengan bagian yang mengalami kerusakan.

Setelah memastikan permasalahan, maka masuk tahapan selanjutnya yaitu perbaikan, ada beberapa tahapan dalam proses perbaikan antara lain:

1. Memperbaiki bagian yang rusak atau melakukan penggantian komponen

   Dalam memperbaiki, harus benar-benar mengetahui apa yang dilakukan, jangan sampai menciptakan permasalahan baru saat perbaikan. Kemudian harus memperhatikan semua batas keamanan pada teknisi saat proses perbaikan.

2. Memastikan perbaikan dengan test menyeluruh

   Memastikan semua permasalahan telah teratasi, obyek bisa berfungsi dengan baik, tidak timbul permasalahan baru karena perbaikan / penggantian komponen, dan semua komponen yang berhubungan dengan obyek yang diperbaiki bisa berfungsi dengan baik.

3. Menginformasikan kepada pemilik, perbaikan apa saja yang telah dilakukan
4. Menyelesaikan administrasi

2.4 Artificial Intelligent

Definisi kecerdasan buatan menurut H. A. Simon [1987] adalah: “Kecerdasan buatan (*artificial intelligent*) merupakan kawasan penelitian, aplikasi dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas”.

Kebanyakan ahli setuju bahwa Kecerdasan Buatan berhubungan dengan 2 ide dasar. Pertama, menyangkut studi proses berfikir manusia, dan kedua, berhubungan dengan merepresentasikan proses tersebut melalui mesin (komputer, robot, dll). Kemampuan untuk *problem solving* adalah salah satu cara untuk mengukur kecerdasan dalam berbagai konteks. Terlihat di sini bahwa mesin cerdas akan diragukan untuk dapat melayani keperluan khusus jika tidak mampu menangani permasalahan remeh atau kecil yang biasa dikerjakan orang secara rutin. Terdapat beberapa alasan untuk memodelkan performa manusia dalam hal ini:

a. Untuk menguji teori psikologis dari performa manusia.

b. Untuk membuat komputer dapat memahami penalaran (*reasoning*) manusia.
c. Untuk membuat manusia dapat memahami penalaran komputer.
d. Untuk mengeksploitasi pengetahuan apa yang dapat diambil dari manusia.

Menurut Winston dan Prendergast (1984), tujuan dari kecerdasan buatan adalah:

a. Membuat mesin menjadi lebih pintar.
b. Memahami apakah kecerdasan (*intelligence*) itu.
c. Membuat mesin menjadi lebih berguna.

2.4.1 Perbandingan AI dengan program komputer konvensional

Program komputer konvensional prosesnya berbasis algoritma, yakni formula matematis atau prosedur yang mengarah kepada suatu solusi. Algoritma tersebut dikonversi ke program komputer yang memberitahu komputer secara pasti instruksi apa yang harus dikerjakan. Algoritma yang dipakai kemudian menggunakan data seperti angka, huruf, atau kata untuk menyelesaikan masalah. Perangkat lunak AI berbasis representasi serta manipulasi simbolik. Di sini simbol tersebut berupa huruf, kata, atau angka yang merepresentasikan obyek, proses dan hubungan keduanya. Sebuah obyek bisa jadi seorang manusia, benda, pikiran, konsep, kejadian, atau pernyataan suatu fakta. Dengan menggunakan simbol, kita dapat menciptakan basis pengetahuan yang berisi fakta, konsep, dan hubungan di antara keduanya. Kemudian beberapa proses dapat digunakan untuk memanipulasi simbol tersebut untuk menghasilkan nasehat atau rekomendasi untuk penyelesaian suatu masalah. Perbedaan dasar antara AI dengan program komputer konvensional diberikan dalam tabel 2.2.

Tabel 2.2. Perbandingan antara AI dan Program Konvensional

Aspek	AI	Program Konvensional
Pemrosesan	Sebagian besar simbolik	Algoritmik
Input	Tidak harus lengkap	Harus lengkap
Pendekatan pencarian	Sebagian besar heuristik	Algoritma
Penjelasan / eksplanasi	Tersedia	Biasanya tidak tersedia
Fokus	Pengetahuan	Data
Pemeliharaan & peningkatan	Relatif mudah	Biasanya sulit
Kemampuan berfikir secara Logis	Ada	Tidak ada

Sumber : Expert System and Artificial Intelligence (Turban, 1992)

2.4.2 Bidang Aplikasi Kecerdasan Buatan

Penerapan kecerdasan buatan yang banyak dikembangkan saat ini adalah (Achmad, 2006):

1. Sistem Pakar (*Expert System*), yaitu program konsultasi (*advisory*) yang mencoba menirukan proses penalaran seorang pakar/ahli dalam memecahkan masalah yang rumit. Sistem pakar merupakan aplikasi AI yang paling banyak. Lebih detail tentang sistem pakar akan diberikan dalam sub bab berikutnya.
2. Pemrosesan Bahasa Alami (*Natural Language Processing*), yang member kemampuan pengguna komputer untuk berkomunikasi dengan komputer dalam bahasa mereka sendiri (bahasa manusia). Sehingga komunikasi dapat dilakukan dengan cara percakapan menggunakan perintah yang biasa digunakan dalam bahasa komputer biasa.
3. Pemahaman Ucapan/Suara (*Speech*/*Voice Understanding*), adalah teknik agar komputer dapat mengenali dan memahami bahasa ucapan. Proses ini mengijinkan seseorang berkomunikasi dengan komputer dengan cara berbicara kepadanya. Istilah pengenalan suara mengandung arti bahwa tujuan utamanya adalah mengenai kata yang diucapkan tanpa harus tahu artinya, di mana bagian itu merupakan tugas pemahaman suara. Secara umum prosesnya adalah usaha untuk menerjemahkan apa yang diucapkan seorang manusia menjadi kata-kata atau kalimat yang dapat dimengerti oleh komputer.
4. Sistem Sensor dan Robotika. Sistem sensor, seperti sistem visi dan pencitraan, serta sistem pengolahan sinyal, merupakan bagian dari robotika. Sebuah robot, yaitu perangkat elektromekanik yang diprogram untuk melakukan tugas manual, tidak semuanya merupakan bagian dari AI. Robot yang hanya melakukan aksi yang telah diprogramkan dikatakan sebagai robot bodoh yang tidak lebih pintar dari lift. Robot yang cerdas biasanya mempunyai perangkat sensor, seperti kamera, yang mengumpulkan informasi mengenai operasi dan lingkungannya. Kemudian bagian AI robot tersebut menerjemahkan informasi tadi dan merespon serta beradaptasi jika terjadi perubahan lingkungan.
5. Komputer Visi, merupakan kombinasi dari pencitraan, pengolahan citra, pengenalan pola serta proses pengambilan keputusan. Tujuan utama dari komputer visi adalah untuk menerjemahkan suatu pemandangan. Komputer visi banyak dipakai dalam kendali kualitas produk industri.

6. *Intellegent Tutoring, Computer-Aided Instruction*, adalah komputer yang mengajari manusia. Belajar melalui komputer sudah lama digunakan, namun dengan menambahkan aspek kecerdasan di dalamnya, dapat tercipta computer guru yang dapat mengatur teknik pengajarannya untuk menyesuaikan dengan kebutuhan murid secara individiual. Sistem ini juga mendukung pembelajaran bagi orang yang mempunyai kekurangan fisik atau kelemahan belajar.
7. Mesin Belajar (*Machine Learning*), yang berhubungan dengan sekumpulan metode untuk mencoba mengajari / melatih komputer untuk memecahkan masalah atau mendukung usaha pemecahan masalah dengan menganalisa kasus-kasus yang telah terjadi.

2.5 Sistem Pakar (*Expert System*)

Sistem pakar pertama kali dikembangkan oleh komunitas AI pada pertengahan tahun 1960. Sistem pakar yang muncul pertama kali adalah *General Purpose Problem Solver* (GPS) yang dikembangkan oleh Newel & Simon (Turban, 1992).

Sistem pakar (*Expert System*) adalah usaha untuk menirukan seorang pakar. Biasanya sistem pakar berupa perangkat lunak pengambil keputusan yang mampu mencapai tingkat performa yang sebanding seorang pakar dalam bidang problem yang khusus dan sempit. Ide dasarnya adalah: kepakaran ditransfer dari seorang pakar (atau sumber kepakaran yang lain) ke komputer, pengetahuan yang ada disimpan dalam komputer, dan pengguna dapat berkonsultasi pada komputer itu untuk suatu nasehat, lalu komputer menyimpulkan dan mendeduksi seperti layaknya seorang pakar, kemudian menjelaskannya ke pengguna sistem tersebut, bila perlu dengan alasannya. Sistem pakar terkadang lebih baik kerjanya daripada seorang pakar manusia.

Definisi sistem pakar menurut John Durkin (1993) adalah suatu program komputer yang dirancang untuk memodelkan kemampuan menyelesaikan masalah dari seorang pakar. Sedang definisi sistem pakar menurut Suyoto (2004) adalah sebagai berikut : Sistem pakar adalah sistem yang didesain dan diimplementasikan dengan bantuan bahasa pemrograman tertentu untuk dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli.

Secara umum sistem pakar adalah sistem yang berusaha mengadopsi kemampuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para pakar. Sistem pakar yang baik dirancang agar dapat menyelesaikan

suatu permasalahan tertentu dengan meniru cara kerja dari para pakar. Sistem pakar dapat mengumpulkan dan menyimpan pengetahuan seorang pakar atau beberapa orang pakar ke dalam komputer.

Pengetahuan tersebut kemudian digunakan oleh siapa saja yang memerlukannya. Tujuan utama sistem pakar bukan untuk menggantikan kedudukan seorang pakar, tetapi hanya untuk memasyarakatkan pengetahuan dan pengalaman para pakar yang keberadaannya cukup jarang. Sistem pakar memungkinkan orang lain dapat meningkatkan produktivitasnya, memperbaiki kualitas pengambilan keputusannya, dan memecahkan masalah rumit lainnya, sekalipun tanpa kehadiran seorang pakar, dengan sistem pakar ini, orang awam sekalipun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli sendiri, sistem pakar ini akan membantu aktivitasnya sebagai asisten yang berpengalaman.

2.5.1 Manfaat dan Keterbatasan Sistem Pakar

Manfaat Sistem Pakar

Sistem pakar menjadi sangat populer disebabkan oleh sangat banyaknya kemampuan dan manfaat yang diberikan oleh sistem pakar, di antaranya (Achmad, 2006):

a. Meningkatkan output dan produktivitas, karena sistem pakar dapat bekerja lebih cepat dari manusia.
b. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
c. Mampu menangkap kepakaran yang sangat terbatas.
d. Dapat beroperasi di lingkungan yang berbahaya.
e. Memudahkan akses ke pengetahuan.
f. Handal. sistem pakar tidak pernah menjadi bosan dan kelelahan atau sakit. Sistem pakar juga secara konsisten melihat semua detil dan tidak akan melewatkan informasi yang relevan dan solusi yang potensial.
g. Meningkatkan kapabilitas sistem terkomputerisasi yang lain. Integrasi sistem pakar dengan sistem komputer lain membuat lebih efektif, dan mencakup lebih banyak aplikasi.

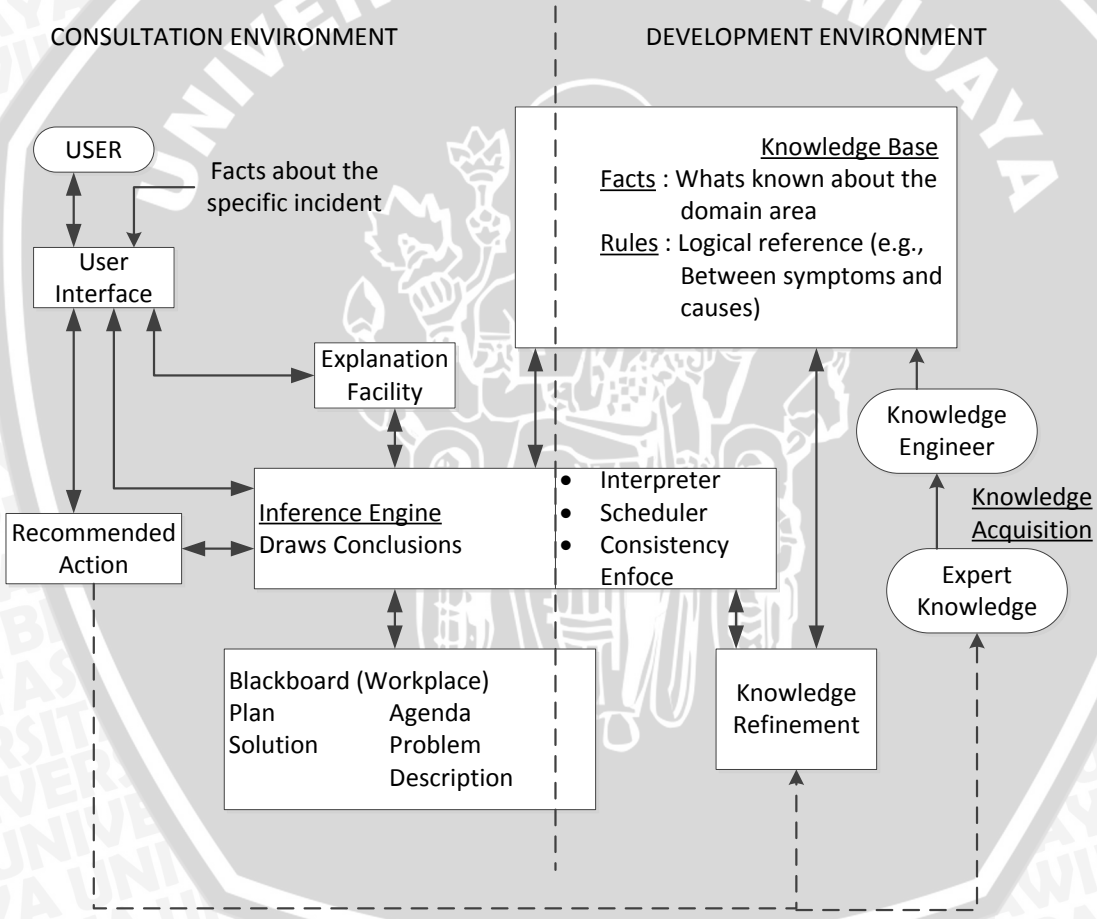
h. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti. Berbeda dengan sistem komputer konvensional, sistem pakar dapat bekerja dengan informasi yang tidak lengkap. Pengguna dapat merespon dengan: "tidak tahu" atau "tidak yakin" pada satu atau lebih pertanyaan selama konsultasi, dan system pakar tetap akan memberikan jawabannya.
i. Mampu menyediakan pelatihan. Pengguna pemula yang bekerja dengan system pakar akan menjadi lebih berpengalaman. Fasilitas penjelas dapat berfungsi sebagai guru.
j. Meningkatkan kemampuan *problem solving*, karena mengambil sumber pengetahuan dari banyak pakar.
k. Meniadakan kebutuhan perangkat yang mahal.
l. Fleksibel.

Keterbatasan Sistem Pakar menurut Achmad (2006) :

a. Metodologi sistem pakar yang ada, tidak selalu mudah, sederhana dan efektif. Berikut adalah keterbatasan yang menghambat perkembangan sistem pakar:
b. Pengetahuan yang hendak diambil tidak selalu tersedia.
c. Kepakaran sangat sulit diungkap dari manusia.
d. Pendekatan oleh setiap pakar untuk suatu situasi atau problem bisa berbeda-beda, meskipun sama-sama benar.
e. Adalah sangat sulit bagi seorang pakar untuk menjelaskan langkah mereka dalam menangani masalah
f. Pengguna sistem pakar mempunyai batas kognitif alami, sehingga mungkin tidak bisa memanfaatkan sistem secara maksimal.
g. Sistem pakar bekerja baik untuk suatu bidang yang sempit.
h. Banyak pakar yang tidak mempunyai jalan untuk mengecek apakah kesimpulan mereka benar dan masuk akal.
i. Istilah yang dipakai oleh pakar dalam mengekspresikan fakta seringkali terbatas dan tidak mudah dimengerti oleh orang lain.
j. Pengembangan sistem pakar seringkali membutuhkan perekayasa pengetahuan (*knowledge engineer*) yang langka dan mahal.
k. Kurangnya rasa percaya pengguna menghalangi pemakaian sistem pakar.
l. Transfer pengetahuan dapat bersifat subyektif.

2.5.2 Komponen Sistem Pakar

Secara umum, sistem pakar biasanya terdiri atas beberapa komponen yang masing-masing berhubungan. *Expert System* terdiri dari dua bagian utama: *Development Environment* dan *Consultation* (runtime) *Environment* (gambar 2.1). *Development Environment* digunakan oleh pembangun *expert system* untuk membangun komponen dan pengetahuan dimasukkan kedalam basis pengetahuan. *Consultation Environment* digunakan oleh orang awam untuk memperoleh pengetahuan dan nasihat ahli (Turban, 1992). Komponen-komponen berikut menurut Turban (1992) mungkin ada dalam sebuah sistem pakar:



Gambar 2.1 Struktur Sistem Pakar
Sumber : Expert System and Artificial Intelligence (Turban, 1992)

a. Antarmuka pengguna (*User Interface*)

User interface merupakan mekanisme yang digunakan untuk pengguna dan sistem pakar untuk berkomunikasi. Antarmuka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu

antarmuka menerima informasi dari sistem dan menyajikannnya dalam bentuk yang dapat dimengerti oleh pemakai. Pada bagian ini terjadi dialog antara program dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan informasi (*input*) dan pemakai juga memberikan informasi (*output*) kepada pemakai.

b. *Knowledge Base* (Basis pengetahuan)

Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang obyek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

c. *Knowledge Acquisition* (Akuisisi pengetahuan)

Akusisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam program komputer. Dalam tahap ini knowledge engineer berusaha menyerap pengetahun untuk selanjutnya ditransfer ke dalam basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, laporan penelitian dan pengalaman pemakai.

Akuisisi pengetahuan dilakukan sepanjang proses pembangunan sistem. Menurut Turban (1992), proses akuisisi pengetahuan dibagi ke dalam enam tahap, yaitu :

1. Tahap identifikasi

Tahap identifikasi meliputi penentuan komponen-komponen kunci dalam sistem yang sedang dibangun. Komponen kunci ini adalah *knowledge engineer*, pakar, karakterisitik masalah, sumber daya, dan tujuan. *Knowledge engineer* dan pakar bekerja bersama untuk menentukan berbagai aspek masalah, seperti lingkup dari proyek, data input yang dimasukkan, bagian-bagian penting dan interaksinya, bentuk dan isi dari penyelesaian, dan kesulitan-kesulitan yang mungkin terjadi dalam pembangunan sistem. Mereka juga harus menentukan sumber pengetahuan seperti basis data, sistem informasi manajemen, buku teks, serta prototipe masalah dan contoh. Selain menentukan sumber pengetahuan, pakar juga mengklarifikasi dan menentukan tujuan-tujuan sistem dalam proses penentuan masalah.

2. Tahap konseptualisasi

   Konsep-konsep kunci dan hubungannya yang telah ditentukan pada tahap pertama dibuat lebih jelas dalam tahap konseptualisasi.

3. Tahap formalisasi

   Tahap ini meliputi pemetaan konsep-konsep kunci, sub-masalah dan bentuk aliran informasi yang telah ditentukan dalam tahap-tahap sebelumnya ke dalam representasi formal yang paling sesuai dengan masalah yang ada.

4. Tahap implementasi

   Tahap ini meliputi pemetaan pengetahuan dari tahap sebelumnya yang telah diformalisasi ke dalam skema representasi pengetahuan yang dipilih.

5. Tahap pengujian

   Setelah prototipe sistem yang dibangun dalam tahap sebelumnya berhasil menangani dua atau tiga contoh, prototipe sistem tersebut harus menjalani serangkaian pengujian dengan teliti menggunakan beragam sampel masalah. Masalah-masalah yang ditemukan dalam pengujian ini biasanya dapat dibagi dalam tiga kategori, yaitu kegagalan input/output, kesalahan logika dan strategi kontrol.

6. Revisi prototipe

   Suatu unsur penting pada semua tahap dalam proses akuisisi pengetahuan adalah kemampuan untuk kembali ke tahap-tahap sebelumnya untuk memperbaiki sistem.

d. *Inference engine* (Mesin inferensi)

   Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace*, dan untuk memformulasikan kesimpulan (Turban, 1992).

e. *Blackboard*

   Blackboard adalah area kerja memori yang disimpan sebagai *database* untuk deskripsi persoalan terbaru yang ditetapkan oleh data input, digunakan juga untuk perekaman hipotesis dan keputusan sementara. Tiga tipe keputusan dapat direkam dalam *blackboard*, yaitu :

   - Rencana : bagaimana mengatasi persoalan

- Agenda : tindakan potensial sebelum eksekusi
- Solusi : hipotesis kandidat dan arahan alternatif yang telah dihasilkan sistem sampai saat ini.

Kemampuan penyelesaian masalah yang efisien

Faktor lain yang membedakan antara pakar dan non-pakar adalah kemampuan (*skill*) dalam menyelesaikan masalah. Seorang pakar seringkali dapat melihat suatu informasi penting dan digunakan sebagai dasar dalam memecahkan masalah secara efisien. Misalnya, pada umumnya, dalam memecahkan suatu masalah diperlukan banyak test dan percobaan untuk memecahkan masalah baru demi menentukan solusi yang tepat. Namun, bagi seorang pakar, ia hanya butuh menggunakan beberapa test yang ia pilih dan mungkin menggunakan pendekatan yang berbeda dari orang lain sehingga bisa menyelesaikan masalah tersebut dengan lebih cepat dan efisien. Contoh kongkrit adalah seorang pakar otomotif, cukup mendengar suara tertentu dari mesin mobil, ia sudah dapat menduga kerusakan yang terjadi, misalnya ring piston mesin sudah longgar.

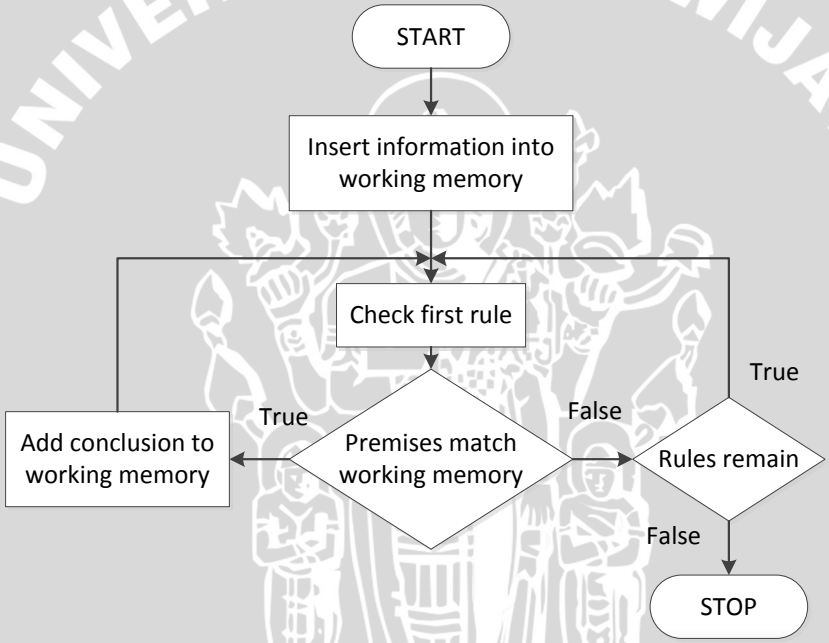
2.5.3 *Inference Engine*

Inference engine adalah salah satu bagian dari sistem pakar. *Inference engine* menerapkan teknik inferensi untuk mengolah basis pengetahuan sehingga sistem sampai pada suatu kesimpulan. Teknik inferensi ada dua, yaitu: *Forward Chaining* (pelacakan ke depan) dan *Backward Chaining* (pelacakan ke belakang).

2.5.3.1 *Forward Chaining*

Forward Chaining atau disebut juga pelacakan ke depan adalah suatu metode pelacakan yang dimulai dengan satu atau beberapa fakta awal sehingga didapatkan hasil atau *goal* yang tidak diketahui sebelumnya. Proses bergerak maju dengan cara mencocokkan fakta tersebut dengan premis atau bagian IF yang ada pada *rule-rule* dalam *knowledge base* (basis pengetahuan). Jika saat dieksekusi sebuah *rule* semua premisnya bernilai benar (*true*) maka bagian THEN akan diambil sebagai fakta baru. Proses pelacakan ini akan dilakukan secara terus-menerus secara berantai sampai ditemukan hasil atau goal yang diinginkan atau sampai tidak terdapat rule lagi.

a. Proses *forward chaining* dapat ditunjukkan pada gambar 2.2. Gambar tersebut dapat dijelaskan sebagai berikut :
b. Masukkan informasi (fakta) ke dalam memori kerja. Untuk mencari hasil dari tujuan yang diinginkan.
c. Cek informasi tersebut pada aturan yang pertama. Aturan tersebut berisikan premis (terdapat pada bagian IF) dan kesimpulan (terdapat pada bagian THEN).
d. Jika fakta tersebut terdapat pada premis. Maka kesimpulan ditambahkan dalam memori kerja. Jika tidak terdapat dalam premis, maka cek aturan berikutnya.
e. Jika masih terdapat aturan, maka cek aturan selanjutnya, tetapi jika aturan tidak terdapat lagi. Maka proses selesai.



Gambar 2.2. Proses *Forward Chaining*
Sumber : Expert System Design and Development (1992)

Keuntungan *Forward Chaining* (Pelacakan ke depan)

- Keuntungan utama pelacakan ke depan adalah ini bekerja baik ketika permasalahan secara normal dimulai dengan pengumpulan informasi dan kemudian melihat apa yang bisa disimpulkan dari data tersebut.
- Pelacakan ke depan mampu menyediakan data yang pantas diperhitungkan dari sedikit data awal.
- Pelacakan ke depan adalah pendekatan yang terbaik untuk tugas penyelesaian masalah tertentu seperti perencanaan, pengamatan, kontrol, dan penafsiran.

Kerugian *Forward Chaining*

- Satu kerugian utama *Forward Chaining* adalah tidak bisa mengerti bahwa beberapa bukti/informasi bisa jadi lebih penting dari pada yang lain. Sistem akan menanyakan semua pertanyaan yang mungkin, meskipun hal itu sebenarnya hanya membutuhkan sedikit pertanyaan untuk mencapai sebuah kesimpulan akhir.
- Sistem juga menanyakan pertanyaan yang tidak berhubungan. Meskipun jawaban untuk pertanyaan tersebut bisa jadi penting, ini membuat bingung pemakai untuk menjawab pertanyaan yang tidak berhubungan dengan permasalahan.

  Q: Do you have high temperatur?
  Q: Have you visited England lately?

2.6 *Decision Table*

Definisi *Decision Table* menurut J. Vanthienen dan G. Wets (1994), *Decision Table* merupakan suatu metode yang digunakan untuk menjelaskan dan menggambarkan aliran data secara logika yang tersimpan didalamnya yang dapat digunakan untuk menyelesaikan sebuah masalah. *Decision table* bekerja dengan cara mengkombinasikan semua kondisi yang ada dimana kondisi ini berisikan aturan-aturan (*rules*) yang disimpan dalam bentuk tabel pada suatu masalah sehingga dapat dipastikan bahwa tidak ada kemungkinan yang terlewati didalam analisa logika terhadap masalah tersebut.

Setiap keputusan sesuai dengan variabel, hubungan atau predikat yang nilai yang mungkin tercantum di antara alternatif kondisi. Setiap tindakan adalah prosedur atau operasi untuk melakukan, dan entri menentukan apakah (atau dalam rangka apa) tindakan yang akan dilakukan untuk set alternatif kondisi masuk sesuai. Banyak tabel keputusan termasuk dalam alternatif kondisi mereka yang tidak peduli simbol, tanda hubung. Menggunakan tidak peduli dapat menyederhanakan tabel keputusan, terutama ketika kondisi tertentu memiliki sedikit pengaruh pada tindakan yang harus dilakukan. Dalam beberapa kasus, kondisi seluruh dianggap penting awalnya ditemukan tidak relevan ketika tidak ada pengaruh kondisi yang tindakan yang dilakukan.

Langkah pembuatan tabel keputusan :

1. Menentukan kondisi yang akan diseleksi
2. Menentukan jumlah kemungkinan kejadian yang akan terjadi
3. Menentukan tindakan yang akan di lakukan

4. Mengisi *condition entry*

Condition Stub	Table Header		1	2	3	4	5	6	7	Condition Entries
	1									
	2									
Action Stub	1									Action Entries
	2									

Gambar 2.3 Contoh Struktur Tabel Keputusan
Sumber: Whiten (1989)

Tiap elemen penyusun *decision table* dapat dijelaskan sebagai berikut :

a. *Tabel header* : nama dari tabel
b. *Condition stub* : menunjukan kondisi
c. *Action stub* : menunjukan sebagai akibat dari action/aksi
d. *Rules column* : sebagai tempat dari pernyataan Y (*yes*) dan N (*no*) untuk mengeset logika dari kondisi.
e. *Condition entries* : merupakan pernyataan Y atau N yang mengikuti status dari setiap kondisi
f. *Action entries* : digunakan untuk setiap *rule* dan berisi satu atau lebih X untuk mengindikasikan bahwa aksi tersebut telah dipilih dan digunakan atau juga tidak berisi apa-apa / *blank* untuk mengindikasikan bahwa tidak ada aksi yang digunakan.

Menentukan struktur dari *decision table*

Jumlah dari kondisi menentukan jumlah dari aturan-aturan (*rules*) yang diperlukan untuk menghasilkan analisis dari semua kemungkinan kombinasi dari kondisi.

Contoh : jika terdapat dua kondisi maka ada 4 *rules* :

Rules 1 : Y untuk kondisi pertama, Y untuk kondisi kedua
Rules 2 : Y untuk kondisi pertama, N untuk kondisi kedua
Rules 3 : N untuk kondisi pertama, Y untuk kondisi kedua
Rules 4 : N untuk kondisi yang pertama, N untuk kondisi yang kedua

Seperti yang ditunjukan di dalam contoh, setiap kondisi dapat bernilai Ya atau Tidak maka dari itu kita memiliki 2 kondisi. Kondisi yang memiliki 2 kondisi menghasilkan 2 kemungkinan. Maka dari itu, kita memerlukan 2 *rule*. Jika terdapat 2

kondisi, kita akan memiliki 2 kesempatan dan 2 kemungkinan sehingga kita memerlukan 4 *rule*, 3 kondisi menghasilkan 2 x 2 x 2 kemungkinan atau 8 *rule.*

Logikanya akan diberikan 2 kondisi Ya/Tidak, dimana N merepresetasikan nomer dari kondisi, 2 pangkat N sama dengan jumlah dari kemngkinan dari kombinasi atau jumlah dari *rules*.

Peta *rules* akan mengilustrasikan semua kombinasi yang mungkin. Peta *rule* terlihat pada table 2.3

Tabel 2.3 Peta *rule*

Rule 1	*Rule* 2	*Rule* 3	*Rule* 4	*Rule* 5	*Rule* 6	*Rule* 7	*Rule* 8
Y	Y	Y	Y	N	N	N	N
Y	Y	N	N	Y	Y	N	N
Y	N	Y	N	Y	N	Y	N

Sumber : Whitten (1989)

Penentuan nilai Y atau N dari rule dilakukan dari kiri ke kanan pada baris yang paling bawah. Dimulai dengan pengisian Y dan N secara bergantian, kemudian dilanjutkan dengan pengisian dua Y dan dua N setelah itu pada baris yang terakhir akan diisi dengan 4 Y dan 4 N, dengan metode seperti ini maka akan dapat dipastikan semua kombinasi yang mungkin telah digunakan dan tidak ada duplikasi.

a. Jika kode produk sama dengan A dan produk ditangan lebih dari 500 maka tidak ada order yang dibutuhkan
b. Jika kode sama dengan A dan produk ditangan kurang dari 500 maka order dibutuhkan
c. Jika kode tidak sama dengan A maka perlu dicek dengan sales sebelum mengorder

Tabel 2.4 Contoh *rule inventory*

No	Inventory	1	2	3	4
1	Code = A	Y	Y	N	N
2	On Hand > 500	Y	N	Y	N
3	No Order needed	X			
4	Order Product		X		
5	Check with sales			X	X

Sumber : Whitten (1989)

Selain dari struktur empat kuadran dasar, tabel keputusan sangat bervariasi di jalan alternatif kondisi dan entri tindakan yang diwakili. Beberapa tabel keputusan menggunakan *true* / nilai-nilai palsu sederhana untuk mewakili alternatif untuk suatu kondisi (mirip dengan *if-then-else*), tabel lain dapat menggunakan nomor alternatif (mirip dengan *switch-case*), dan beberapa meja bahkan menggunakan logika *fuzzy* atau representasi probabilistik alternatif kondisi. Dalam cara yang sama, entri tindakan hanya dapat mewakili apakah suatu tindakan yang akan dilakukan (periksa tindakan untuk melakukan), atau lebih tabel keputusan maju, urutan tindakan untuk melakukan (nomor tindakan untuk melakukan).

Decision Table, ketika digabungkan dengan penggunaan bahasa domain-spesifik , memungkinkan pengembang dan ahli kebijakan untuk bekerja dari informasi yang sama, tabel keputusan sendiri. Alat untuk membuat bersarang jika pernyataan dari bahasa pemrograman tradisional menjadi tabel keputusan juga dapat digunakan sebagai alat *debugging*. Tabel keputusan telah terbukti menjadi lebih mudah untuk memahami dan mengkaji dari kode, dan telah digunakan secara luas dan berhasil untuk menghasilkan spesifikasi untuk sistem yang kompleks.

2.7 *Microsoft Acces*

2.7.1 Pengenalan *Micosoft Access*

Microsoft Access adalah salah satu program pengelola basis data yang berdaya guna dan fleksibel. *Access* bukanlah suatu program sederhana meskipun *Access* mempunyai beberapa fitur untuk membantu para pemula. *Access* memungkinkan penggunanya untuk mengumpulkan, menyimpan, dan mengatur informasi seperti halnya membuat laporan yang mengarah kepada kesimpulan akhir.

2.7.2 Penyimpanan Data oleh *Microsoft Access*

Pada *Access*, pertama-tama perlu dibuat satu *file* basis data. *File* tersebut menyimpan segala sesuatu yang dibuat untuk basis data, tidak hanya semua data, tapi juga *form-form*, laporan-laporan, dan indeks.

- Tabel

  Tabel adalah pusat basis data. Tabel sangat mirip dengan *spreadsheet*. *Access* menyimpan setiap entri basis data pada barisnya sendiri.

- *Form*

  Semua data yang dimasukkan ke basis data akan disimpan dalam tabel melalui *form*.

- Laporan

  Form dirancang untuk pemakai pada layar, sedangkan laporan (*report*) dirancang untuk dicetak. Laporan adalah kumpulan data yang diformat scara khusus dan dikelola menurut spesifikasi *user*.

- *Query*

  Query adalah suatu cara untuk membuang informasi yang tidak perlu dilihat sehingga user hanya melihat informasi yang diperlukan saja.

2.8 Integrasi *Expert System* dan *Decision Table*

Untuk membangun integrasi antara *Expert System* dan *decision table*, perlu ditentukan terlebih dahulu peran *decision table* pada suatu *Expert System*, dalam hal ini *decision table* dibangun dengan sistem basis data. Menurut Harmon (1990), terdapat beberapa cara kerja basis data dalam integrasinya dengan *Expert System*, yaitu:

1. Suatu *Expert System* dapat digunakan sebagai *front end* bagi basis data (*Expert System* ditempatkan di depan basis data). Dalam hal ini, *Expert System* memberikan pertanyaan kepada pengguna pada saat melakukan konsultasi dan selanjutnya memberikan kesimpulan akhir berdasarkan aturan-aturan yang ada untuk inisialisasi *query* basis data.
2. Suatu *Expert System* dapat digunakan sebagai *back end* bagi basis data (*Expert System* ditempatkan di belakang basis data). Dalam hal ini, *Expert System* mengambil hasil *query* dari basis data dan menganalisanya menggunakan *rule-rule* yang ada untuk kemudian dibuat suatu rekomendasi.
3. Atribut dan nilai dari *rule-rule* pada *Expert System* dapat disimpan pada suatu basis data. Secara umum, cara ini digunakan untuk membuat *Expert System* dengan pendekatan *decision table*.

4. Basis data dapat digunakan untuk menyimpan kasus-kasus yang sebelumnya telah dijalankan pada *Expert System* dengan tujuan untuk mencatat alasan serta rekomendasi yang telah diberikan.

Pada perancangan penelitian ini, cara integrasi yang digunakan adalah cara yang kedua, yaitu *Expert System* berperan sebagai *front-end* bagi basis data.