BAB 2

TINJAUAN PUSTAKA

2.1 Sejarah Berdirinya Polda Metro Jaya

Cikal bakal Kepolisian Jakarta di bentuk oleh penjajah Belanda. Ini terjadi sejak penduduk Belanda terhadap bangsa Indonesia, jauh sebelum Proklamasi Kemerdekaan Republik Indonesia tanggal 17 Agustus 1945. Setelah Kemerdekaan Rl, pembentukan Kepolisian Kota Jakarta belum sepenuhnya dapat dilaksanakan Bangsa Indonesia, sebab saat itu jawatan Kepolisian Negara masih sangat sederhana. Akibatnya, Kepolisian Kota Jakarta masih tetap melanjutkan sistem Kepolisian yang dibentuk pada masa pendudukkan Jepang. Inilah yang menyebabkan penulisan sejarah hari jadi Polda Metro Jaya diawali dari sejarah Kepolisian Batavia di tahun 1936 (sesuai Regeerings Almanak Halaman 287 Voor Nederlandsch Indie 1941 Tweede Gedeelte yang disusun Belanda selama berada di Indonesia). [22]

2.1.1 Logo Polda Metro Jaya

Dalam pengabdiannya kepada negara dan pelayanannya kepada masyarakat, Polda Metro Jaya memiliki atribut badge sebagai simbol dari satuan unit Polda Metro Jaya itu sendiri. Adapun makna badge dari Satuan Kepolisian Polda Metro Jaya (Gambar 2.1) adalah sebagai berikut:

a. Bentuk; Oval.

b. Wujud; Tugu Nasional, berwarna putih. Untaian padi berwarna kuning. Untaian kapas berwarna hijau serta putih, Pita berwarna merah putih.

c. Artinya; Kekhususan kota Jakarta sebagai kota Revolusi dan kota Proklamasi dilambangkan dengan Tugu Nasional yang melambangkan kemegahan, daya juang, cipta bangsa dan rakyat Indonesia yang tak kunjung padam. Untaian Padi dan Kapas yang melingkari Tugu Nasional melambangkan cita-cita perjuangan bangsa Indonesia yang bertujuan mewujudkan masyarakat adil dan makmur. Pita Merah Putih yang melingkari dasar adalah Tugu Nasional, melambangkan persatuan bangsa Indonesia yang kokoh dan sentosa.

Tata Warna dan Artinya:

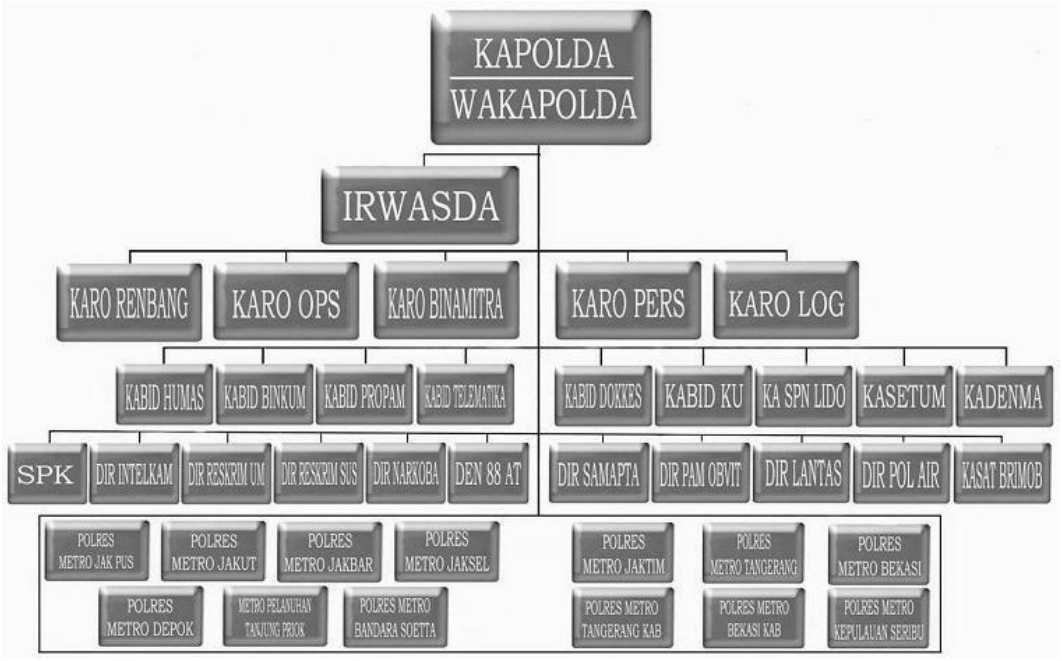
1. Putih pada Tugu Nasional, berarti kemegahan kreasi mulia.
2. Kuning padi dan hijau kapas, berarti kemakmuran dan keadilan.
3. Pita Merah Putih, berarti kedaulatan, kehormatan dan persatuan bangsa Indonesia.



Gambar 2.1 Badge Polda Metro Jaya

2.1.2 Struktur Organisasi Polda Metro Jaya

Dalam organisasinya Polda Metro Jaya Dipimpin oleh seorang Kapolda dengan Wakapolda sebagai wakilnya, dan membawahi beberapa bidang, diantaranya yaitu Karo Renbang, Karo RPS, Karo Binamitra, Karo Pers, dan Karo Log. Berikut adalah struktur organisasi Polda Metro Jaya.



Gambar 2.2 Struktur Organisasi Polda Metro Jaya

2.1.3 Visi dan Misi Polda Metro Jaya

Polda Metro Jaya memiliki visi dan misi dalam mengemban tugasnya sehari-hari demi pengabdiaannya kepada negara dan pelayanannya kepada masyarakat.

2.1.3.1 Visi

Tergelarnya polisi yang dipercaya masyarakat disemua titik dan lini pelayanan masyarakat disepanjang waktu dalam mewujudkan keamanan diwilayah hukum Polda Metro Jaya dan tegaknya hukum sebagai sinergi pencapaian hasil pembangunan yang berwawasan keamanan.

2.1.3.2 Misi

a. Perkuat dan tingkatkan kemampuan intelijen keamanan Polda Metro Jaya guna menjaring informasi untuk mencegah gangguan keamanan dan pengungkapan kasus yang secara sistematis dan tuntas.
b. Kembangkan pelayanan publik di setiap lini berbasis pelayanan prima.
c. Menggelar polisi sebanyak-banyaknya ditengah-tengah masyarakat dalam memberikan perlindungan, pengayoman dan pelayanan masyarakat.
d. Kembangkan falsafah dan strategi perpolisian masyarakat (*polmas*) dalam membangun hubungan polisi dan masyarakat yang lebih dekat dan interaktif dalam upaya mewujudkan masyarakat yang patuh hukum.
e. Berdayakan seluruh kekuatan dan kemampuan organisasi pengemban fungsi lidik dan sidik dalam mewujudkan POLRI sebagai penegak hukum yang terdepan.
f. Tingkatkan kinerja Polda Metro Jaya secara profesional, transparan dan akuntabel guna mendukung *tupoksi* POLRI.

2.1.4 Satwal Patwal Polda Metro Jaya

Secara baku, Sat Patwal didefinisikan sebagai unsur pelaksana pada Direktorat Lalu Lintas Polda Metro Jaya, yang hanya bertugas untuk menyelenggarakan fungsi pengawalan dan patroli di jalan non tol, bersifat terpusat/regional. Sat Patwal (Gambar 2.3) memberikan bimbingan teknis dan mengendalikan pelaksanaan patroli dan pengawalan oleh satuan kewilayahan di lingkungan Polda Metro Jaya.

Penekanan pada kalimat “Patroli di jalan non tol” menjadi penting untuk menghindari terjadinya tumpang tindih dengan satuan lain di lingkungan Polda Metro Jaya (PMJ). Pengaturan lalu lintas di lingkungan PMJ melibatkan tiga satuan utama yaitu PJR (Patroli Jalan Raya), Gatur (Penjagaan dan Pengaturan), dan Patwal (Patroli dan Pengawalan).



Gambar 2.3 Logo Sat Patwal Polda Metro Jaya

Utamanya Patwal memiliki tugas dan tanggung jawab yang cukup kompleks dan boleh dikatakan *elit*. Memang tugas yang dilakukan tidak bersentuhan langsung dengan keamanan Presiden seperti yang dipahami Paspamres (Pasukan Pengamanan Presiden). Namun dengan kesigapan petugas

Patwal serta Polisi lainnya dalam mengatur lalu lintas, setiap perjalanan Kepala Negara bisa dilakukan dengan baik.

Secara struktural Sat Patwal berada dibawah kendali Direktur Lantas Polda Metro. Jabatan Kasat Patwal dipegang perwira berpangkat AKBP (Ajun Komisaris Besar Polisi) dan Direktur Lantas berpangkat KBP (Komisaris Besar Polisi).

Sebagai Ibukota Negara, Jakarta merupakan pusat aktifitas kenegaraan dan ekonomi. Selain tempat bermukimnya Kepala Negara, Jakarta juga dihuni banyak tamu-tamu negara, para duta besar, pejabat tinggi negara, Ketua DPR/MPR dan eksekutif lainnya. Dalam kegiatan profesionalnya, tokoh-tokoh ini sering melakukan perjalanan dengan intensitas tinggi. Dengan demikian, aspek kelancaran dan kenyamanan mereka dijalan raya menjadi perhatian besar petinggi POLRI sejak dulu. Khusus pengawalan Kepala Negara, Polisi sudah mengambil peran ini sejak masa awal kemerdekaan. Sampai hari ini, peran itu tetap dijalankan.

Selain aktifitas kenegaraan, di Jakarta juga berlangsung interaksi sosial dan ekonomi yang sangat cepat. Lingkungan eksternal ini akhirnya membuat penduduk Jakarta hidup dalam irama yang cepat dan dinamis. Hanya saja dinamika sosial yang kadang tidak terkendali itu membawa dampak negatif di jalan raya. Penambahan jumlah kendaraan tidak seimbang dengan penambahan panjang dan jumlah jalan baru, yang terjadi lalu kemacetan yang kian hari makin parah.

Sejak pemikiran itu muncul, tugas dan tanggung jawab Patwal mengalami penyesuaian. Tak lagi sekedar mengawal Kepala Negara dan pejabat tinggi, Patwal menawarkan jasa pengawalan kepada masyarakat. Alhasil bagi siapa saja yang membutuhkan pengawalan di jalan raya, Sat Patwal PMJ akan mengirimkan petugasnya untuk melayani Masyarakat. Di daerah, tugas sejenis bisaanya berada di bawah komando Sat PJR.

Menangani hal-hal tersebut dalam mendukung upaya dari kinerja Satwal Patwal Polda Metro Jaya tentunya didukung oleh teknisi-teknisi yang mengemban tugas dalam menjaga kinerja dan performa dari kendaraan-kendaraan yang digunakan oleh kepolisian setempat. Bpk. Herianto bersama dengan 9 rekannya menjadi bagian dari teknisi dalam mengemban hal tersebut. Dengan kurun waktu kerja lebih dari 15 tahun bekerja di Polda Metro Jaya, beliau adalah bagian dari para teknisi yang mampu menangani kerusakan-kerusakan yang dialami oleh kendaraan-kendaraan yang digunakan oleh kepolisian tersebut, terutama dibagian Satwal Patwal Polda Metro Jaya. [23]

2.2 Definisi dan Konsep Dasar Sistem Pakar

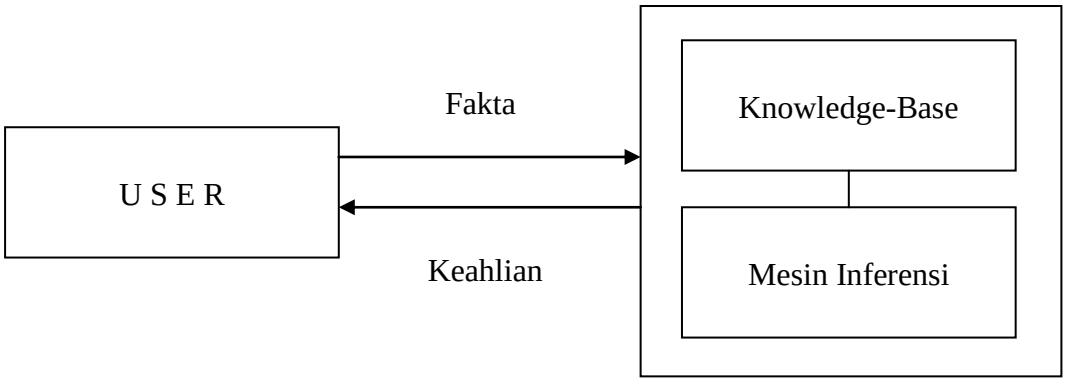
Sistem Pakar menurut Professor Edward Feigenbaum adalah suatu program komputer cerdas yang menggunakan *knowledge* (pengetahuan) dan prosedur inferensi untuk menyelesaikan masalah yang cukup sulit sehingga membutuhkan seorang yang ahli untuk menyelesaikannya. Suatu sistem pakar adalah suatu sistem komputer yang menyamai (*emulates*) kemampuan pengambilan keputusan dari seorang pakar. Istilah *emulates* berarti bahwa sistem

pakar diharapkan dapat bekerja dalam semua hal seperti seorang pakar. Suatu emulasi jauh lebih kuat daripada suatu simulasi yang hanya membutuhkan sesuatu yang bersifat nyata dalam beberapa bidang atau hal. [6]

Sistem pakar merupakan salah satu cabang dari *Artificial Intellegence* yang membuat penggunaan secara luas mengenai *knowledge* yang khusus untuk menyelesaikan masalah pada tingkat manusia. Seorang pakar adalah orang yang mempunyai keahlian dalam bidang tertentu dalam, yaitu pakar yang mempunyai *knowledge* atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya. Implementasinya digunakan pada berbagai macam sistem yang menggunakan teknologi sistem pakar. Teknologi sistem pakar ini meliputi bahasa sistem pakar, program dan perangkat keras yang dirancang untuk membantu pengembangan dan pembuatan sistem pakar (Gambar 2.4). [2]

Knowledge dalam sistem pakar mungkin saja seorang ahli, atau *knowledge* yang umumnya terdapat dalam buku, majalah, artikel-artikel dan atau orang yang memiliki pengetahuan tentang suatu bidang.

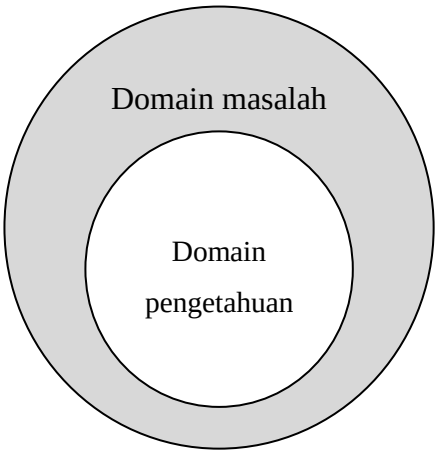
Implementasinya pengguna menyampaikan fakta atau informasi untuk sistem pakar dan kemudian menerima saran dari pakar atau jawaban dari ahlinya. Bagian dalam sistem pakar terdiri dari 2 komponen utama, yaitu *knowledge base* yang berisi *knowledge* dan *mesin inferensi* yang menggambarkan kesimpulan. Kesimpulan tersebut merupakan respon dari sistem pakar atas permintaan pengguna.



Gambar 2.4 Konsep Dasar Fungsi Sistem Pakar

Suatu *knowledge* dari sistem pakar bersifat khusus untuk suatu domain masalah. Domain masalah adalah bidang atau ruang lingkup yang khusus, seperti kedokteran, keuangan, bisnis, ilmu pengetahuan dan teknik. Sistem pakar yang menyerupai kepakaran manusia yang secara umum dirancang untuk menjadi pakar dalam satu domain masalah saja.

Knowledge dari sistem pakar tentang penyelesaian masalah yang khusus disebut dengan *domain knowledge* dari suatu pakar (Gambar 2.5). Pada studi kasus ini, yaitu sistem pakar mekanik yang dirancang untuk mendiagnosis kerusakan pada mesin-mesin akan mempunyai suatu uraian *knowledge* tentang gejala-gejala yang disebabkan oleh berbagai macam hal. *Domain knowledge*-nya adalah bidang mekanik yang terdiri dari *knowledge* tentang mesin-mesin, gejala, dan cara penanggulangnnya.



Gambar 2.5 Hubungan Antara Domain Knowledge dan Domain Masalah

2.2.1 Kelebihan dan Kelemahan Sistem Pakar

Seorang pakar dengan sistem pakar mempunyai perbedaan. Pada Tabel 2.1 merupakan perbandingan konkret antara kemampuan sorang pakar dengan sistem pakar:

Tabel 2.1 Perbandingan Kemampuan Seorang Pakar dengan Sistem Pakar

Factor	Human Expert	Expert System
Time avability	*Hari kerja*	*Setiap saat*
Geografis	*Lokal/tertentu*	*Dimana saja*
Keamanan	*Tidak tergantikan*	*Dapat diganti*
Perishable/dapat habis	*Ya*	*Tidak*
Performansi	*Variable*	*Konsisten*
Kecepatan	*Variable*	*Konsisten*
Biaya	*Tinggi*	*Terjangkau*

Berikut adalah penjelasan rinci mengenai Tabel 2.1, yaitu:

1. Sistem pakar bisa digunakan setiap hari menyerupai sebuah mesin sedangkan seorang pakar tidak mungkin bekerja terus-menerus setiap hari tanpa istirahat.
2. Sistem pakar merupakan suatu *software* yang dapat diperbanyak dan kemudian dibagikan ke berbagai lokasi maupun tempat yang berbeda-beda untuk digunakan, sedangkan seorang pakar hanya bekerja 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/ancaman, sedangkan seorang pakar bisa saja mendapat ancaman atau tekanan pada saat menyelesaikan permasalahan.
4. Pengetahuan yang disimpan pada sistem pakar tidak akan dapat hilang/lupa, yang dalam hal ini harus didukung oleh *maintenance* yang baik, sedangkan pengetahuan seorang pakar manusia lambat laun akan hilang karena usia yang semakin tua, menderita suatu penyakit, maupun meninggal. Walaupun pengetahuan yang dimilikinya dalam waktu yang singkat tidak akan hilang, akan tetapi bisa saja seorang pakar mengundurkan diri dari pekerjaannya, pindah tugas atau bahkan dipecat dari pekerjaannya sehingga organisasi yang mempekerjakannya akan kehilangan seorang pakar yang berbakat.
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 dapat 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.
7. Biaya menggaji seorang pakar lebih mahal bila dibandingkan dengan penggunaan program sistem pakar (dengan asumsi bahwa sistem pakar itu sudah ada).
8. Secara otomatis mengerjakan tugas-tugas rutin yang membutuhkan seorang pakar.

Sistem pakar merupakan paket perangkat lunak atau paket program komputer yang ditujukan sebagai penyedia nasihat dan sarana bantu dalam memecahkan masalah dibidang-bidang spesialisasi tertentu seperti sains, perekayasaan, matematika, kedokteran, pendidikan, dan lain-lain. Ada beberapa keunggulan sistem pakar, diantaranya yaitu 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 dan tanpa jemu mencari kembali data yang tersimpan dengan kecepatan tinggi.

Sementara kemampuan sistem pakar diantaranya adalah:

1. Menjawab berbagai pertanyaan yang menyangkut bidang keahliannya.
2. Bila diperlukan dapat menyajikan asumsi dan alur penalaran yang digunakan untuk sampai kepada jawaban yang dikehendaki.
3. Menambah fakta kaidah dan alur penalaran sahih yang baru kedalam programnya.

Selanjutnya keuntungan dalam penggunaan sistem pakar adalah sebagai berikut:

1. Menjadikan pengetahuan dan nasihat lebih mudah didapat.
2. Meningkatkan output dan produktivitas.
3. Menyimpan kemampuan dan keahlian pakar.
4. Meningkatkan penyelesaian masalah–meneruskan panduan pakar, penerangan, sistem pakar yang khas.
5. Meningkatkan *reliabilitas*.
6. Memberikasn *respon* (jawaban) yang cepat.
7. Merupakan panduan yang *intelligence*.
8. Dapat bekerja dengan informasi yang kurang lengkap dan yang mengandung ketidakpastian.

Selain pendeskripsian keuntungan-kentungan, sistem pakar seperti halnya sistem lain juga memiliki beberapa kelemahan, yaitu:

1. Masalah dalam mendapatkan pengetahuan dimana pengetahuan tidak selalu bisa didapatkan dengan mudah, karena kadangkala pakar dari masalah yang dibuat tidak ada, dan jikapun 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 pemeliharaannya.
3. Boleh jadi sistem tidak dapat membuat keputusan.

Sistem pakar tidaklah 100% menguntungkan, walaupun seorang tidak sempurna atau tidak selalu benar. Oleh karena itu perlu kajian ulang secara teliti sebelum digunakan. Dalam hal ini peran pakar (manusia) tetap merupakan faktor dominan. [2]

2.2.2 Elemen Sistem Pakar

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

Keahlian merupakan suatu penguasaan pengetahuan dibidang tertentu yang didapatkan dari pelatihan, membaca atau pengalaman. Contoh bentuk pengetahuan yang merupakan keahlian adalah:

1. Fakta-fakta pada lingkup permasalahan tertentu.
2. Teori-teori pada lingkup permasalahan tertentu.
3. Prosedur-prosedur dan aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
4. Strategi-strategi global untuk menyelesaikan masalah.
5. *Meta-Knowledge* (pengetahuan tentang pengetahuan).

Bentuk- bentuk tersebut memungkinkan para ahli untuk dapat mengambil keputusan lebih cepat dan lebih baik dari seorang yang bukan ahli. Seorang ahli adalah seorang yang mempunyai pengetahuan tertentu dan mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan (domain), menyusun kembali pengetahuan jika dipandang perlu, memilah aturan jika dibutuhkan, dan menentukan relevansi 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 4 aktivitas, yaitu tambahan pengetahuan (dari para ahli atau sumber lainnya), representasi pengetahuan (kedalam komputer), inferensi pengetahuan dan pengalihan pengetahuan ke pengguna. Pengetahuan yang disimpan dalam komputer dinamakan dengan nama basis pengetahuan (*knowledge base*). Terdapat dua tipe pengetahuan yaitu, fakta dan prosedur. 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*). [7]

2.2.3 Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*) (Lihat Gambar 2.6). Lingkungan pengembangan

sistem pakar digunakan untuk memasukan pengetahuan pakar ke dalam lingkungan sistem pakar, sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar.

Terdapat tiga orang yang terlibat dalam lingkungan sistem pakar, yaitu:

1. Pakar.

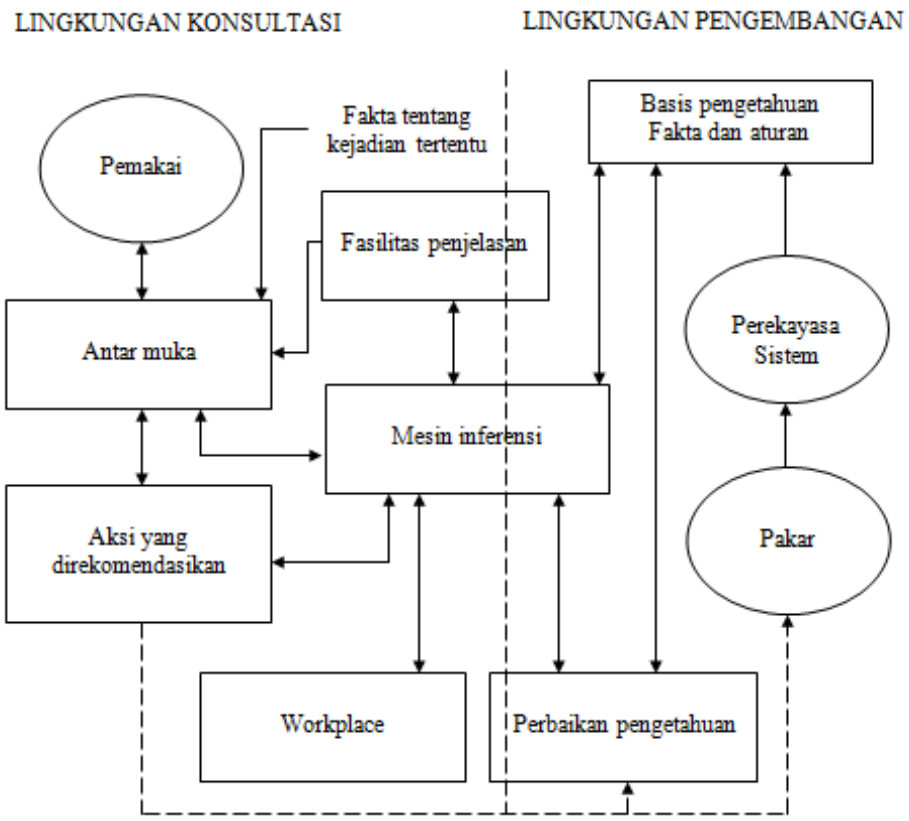
   Pakar adalah orang yang memiliki pengetahuan khusus, pendapat, pengalaman dan metode, serta kemampuan untuk mengaplikasikan keahliaannya tersebut guna menyelesaikan masalah.

2. Perekayasa Sistem.

   Perekayasa sistem adalah orang yang membantu pakar dalam menyusun area permasalahan dengan menginterpretasikan dan mengintegrasikan jawaban-jawaban pakar atas pertanyaan yang diajukan, menggambarkan analogi, mengajukan *counter example* dan menerangkan kesulitan-kesulitan konseptual.

3. Pemakai.

   Sistem pakar memiliki beberapa pemakai, yaitu: pemakai bukan pakar, pelajar, pembangun sistem pakar yang ingin meningkatkan dan menambah basis pengetahuan, dan pakar.



Gambar 2.6 Arsitektur Lingkungan Sistem Pakar

2.2.3.1 Akusisi Pengetahuan

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

1. Wawancara.

   Metode ini melibatkan pembicaraan dengan pakar secara langsung dalam suatu wawancara. Terdapat beberapa bentuk wawancara dengan tujuan yang berbeda, yaitu:

   a. Contoh kasus.

      Pakar akan dihadapkan pada sebuah kasus nyata.

   b. Wawancara klasifikasi.

      Ditujukan untuk memperoleh wawasan pakar untuk domain permasalahan tertentu.

   c. Wawancara terarah.

      Metode ini merupakan pelengkap bagi metode wawancara dengan klasifikasi menggunakan contoh kasus dan wawancara klasifikasi. Dalam bentuk wawancara ini, pakar dan perekayasa sistem mendiskusikan domain dan cara penyelesaian kasus dalam tingkat yang lebih umum dari dua metode sebelumnya.

2. Diskusi kasus dalam konteks dari sebuah prototipe sistem.

   Seorang pakar dihadapkan dengan sebuah contoh kasus dari prototipe sistem. Metode ini digunakan untuk melihat apa yang pakar pikirkan tentang prototipe sistem.

3. Analisis protokol.

   Seorang pakar diminta untuk melakukan suatu pekerjaan dan megungkapkan proses pemikirannya dengan menggunakan kata-kata. Pekerjaan tersebut direkam, dituliskan, dan dianalisis.

4. Observasi pada pekerjaaan pakar.

   Pekerjaan dalam bidang tertentu yang dilakukan pakar direkam dan diobservasi.

5. Induksi aturan dari contoh.

   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 contoh kasus-kasus. Selanjutnya aturan dapat digunakan untuk menilai kasus orang lain yang hasilnya tidak diketahui.

2.2.4 Aturan Kaidah Produksi

Pengetahuan dalam sistem produksi boleh juga direpresentasikan oleh himpunan kaidah dalam bentuk:

IF [kondisi] **THEN** [aksi]

Dengan sebuah kontrol sistem dan basis data. Kontrol sistem memberikan aturan penerjemahan dan pengurutan. Basis data beraksi sebagai konteks cadangan untuk record yang kondisinya dievaluasi oleh kaidah dan informasi dimana kaidah akan beraksi. Berikutnya untuk sintaks **IF-THEN**, kaidah produksi juga sering digambarkan sebagai pasangan-pasangan berikut kondisi-aksi, anteceden-konsequent, pola-aksi, situasi-responsi. Contoh-contoh produksi dari kasus, yaitu:

…

IF [mesin tidak mau hidup/sulit dinyalakan]

THEN [kerusakan pada selang ventilasi udara]

IF [mesin berbunyi tidak normal **AND** busi menjadi cepat tertutup arang]

THEN [kerusakan pada crankshaft]

…

Struktur intuitif secara alami untuk representasi pengetahuan yang telah dihasilkan oleh sistem produksi merupakan metode yang banyak diimplementasikan dalam sistem pakar, termasuk perintis program DENDRAL dan MYCIN. Sebagai contoh, kaidah produksi dari sistem tersebut seperti yang ditunjukan dibawah ini.

…

DENDRAL

IF [the spectrum of the molecule has two peaks at masses x_1 and x_2 such that]

a) $x_1 + x_2 = M + 28$ **AND**

b) $x_1 - 28$ is a high peak **AND**

c) $x_2 - 28$ is a high peak **AND**

d) at least one of x_1 or x_2 is high

THEN [the molecule containts a ketone group]

…

MYCIN

IF [

a) the stain of the organism is gramneg **AND**

b) the morphology of the organism is rod **AND**

c) the patient is a compromised host

]

THEN [there is suggestive evidence 90,60 that the identity of the organism is pseudomonas]

…

Dari kesamaan perumusan kedua kaidah produksi itu, didapat suatu gambaran atraktif pertama dari sistem produksi secara formal: pemisahan secara rapi dari struktur program dari data. Ada tiga elemen utama dari semua sistem produksi, yaitu:

1. Database Global

   Database global merupakan struktur data utama dari sistem produksi. Database mungkin mempunyai jangkauan dari sebuah daftar sederhana atau matriks kecil hingga kekompleksan, relasi, dan struktur indeks. Hal ini merupakan struktur dasar dimana kaidah produksi dapat beroperasi. Hal iini juga merupakan struktur dinamis dan berubah-ubah secara *continue* sebagai hasil dari operasi kaidah produksi. Database global juga menjadi acuan untuk konteks, memori cadangan jangka pendek, atau memori kerja.

2. Kaidah Produksi

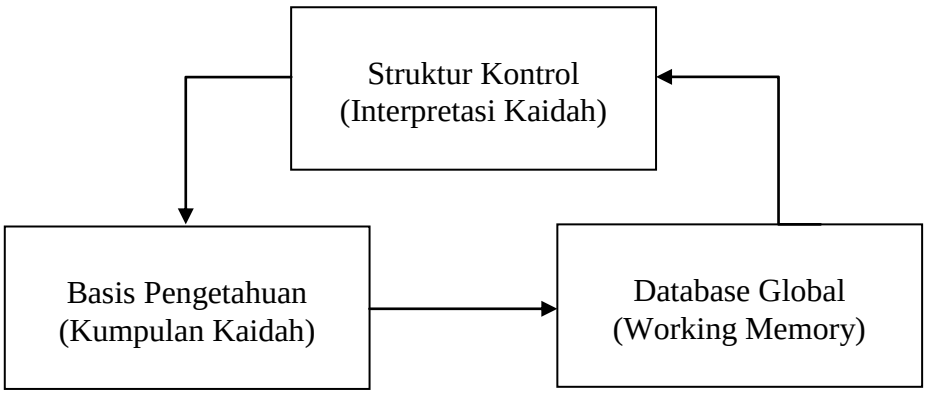
   Sebagaimana yang diindikasikan oleh contoh diatas, kaidah produksi mempunyai bagian kondisi (IF) yang disebut bagian kanan dan aksi (THEN) disebut bagian kiri. Jika sisi kiri kadang-kadang dinamakan kondisi atau

premis yang dipenuhi oleh database, maka kaidah-kaidah dapat diterapkan dan subjek menjadi pemicu bagi sistem kontrol.

3. Sistem Kontrol

   Sistem kontrol merupakan program penterjemah yang esensial untuk mengontrol urutan dimana kaidah-kaidah produksi dipicu dan menyelesaikan konflik jika lebih dari satu kaidah yang diaplikasikan. Sistem kontrol secara berulang-ulang mengaplikasikan kaidah-kaidah untuk database hingga sebuah gambaran dari tujuan yang dihasilkan. Kemudian mendeteksi kejadian seperti tujuan dan record kaidah yang telah diaplikasikan untuk mencapainya bagi referensi sebelumnya.

Masalah yang dihasilkan oleh sistem nyata untuk tiga elemen ini dinamakan masalah representasi (*representasi problem*). Hal ini merupakan masalah utama dalam rancangan sistem produksi sembarang. (Lihat Gambar 2.7)



Gambar 2.7 Komponen Sistem Produksi

2.2.5 Kaidah Produksi, Pengetahuan, dan Kaidah Inferensi

Kaidah produksi dituliskan dalam bentuk pernyataan **IF-THEN** (**Jika-Maka**). Pernyataan ini menghubungkan bagian premis (**IF**) dan bagian kesimpulan (**THEN**) yang dituliskan dalam bentuk:

IF [premis] **THEN** [konklusi]

Apabila bagian premis dipenuhi maka bagian konklusi akan bernilai benar. Sebuah kaidah terdiri dari klausa-klausa. Sebuah klausa mirip dengan sebuah kalimat dengan subjek, kata kerja dan objek yang menyatakan suatu fakta. Suatu aturan juga dapat terdiri dari beberapa premis dan lebih dari satu konklusi. Untuk merumuskan beberapa domain pengetahuan secara akurat diperlukan banyak kaidah produksi. Aturan-aturan ini menyediakan rincian objek, karakteristik, dan tindakan-tindakan yang harus diambil. Untuk dapat meliputi suatu objek diperlukan banyak rincian aturan. Aturan-aturan ini biasanya saling berkaitan dan saling mengacu. Aturan ini membentuk pangkalan pengetahuan yang kemudian menjadi bagian sistem produksi. Biasanya pengetahuan diturunkan dalam bentuk pernyataan linguistic dari pakar. Sebagai contoh:

…

Bercak pada daun itu sangat banyak

Ujung daun kering itu banyak

…

Dalam contoh pertama, variabel linguistik “Bercak pada daun” memiliki nilai linguistik “sangat banyak”, sedangkan pada contoh kedua, variabel linguistik “Ujung daun kering” memiliki nilai linguistik “banyak”.

Bagian premis dalam aturan produksi dapat memiliki lebih dari satu sisi proposisi. Proposisi-proposisi tersebut dihubungkan dengan menggunakan operator logika AND atau OR. Sebagai contoh:

…

Bercak pada daun itu sangat banyak

AND ujung daun kering itu banyak

…

Aturan produksi digunakan untuk menyatakan hubungan antara dua bentuk pernyataan linguistik yang masing-masing merupakan bagian premis dan konklusi/kesimpulan. Sebagi contoh:

…

IF bercak pada daun itu sangat banyak

AND ujung daun kering itu banyak

THEN penyakit bercak ungu itu puso

…

Kaidah menurut Turban dapat dibuat dalam beberapa bentuk yang berbeda, yaitu:

1. **IF** Premis **THEN** Konklusi. Contohnya, IF pendapatan anda tinggi THEN kesempatan anda untuk diaudit oleh IRS adalah tinggi.
2. Konklusi, **IF** premis, kesempatan anda diaudit adalah tinggi IF pendapatan anda tinggi.

3. Inklusi dari **ELSE**, IF pendapatan anda tinggi OR deduksi anda luar biasa THEN kesempatan anda diaudit tinggi OR ELSE kesempatan anda diaudit rendah.
4. Kaidah yang lebih kompleks, IF tingkat kredit adalah tinggi AND gaji lebih dari $30,000 OR asset lebih dari $75,000 AND histori pelunasan lancer THEN disetujui peminjaman hingga $10,000 dan daftar peminjamannya termasuk dalam kategori B. [6]

Ada dua tipe kaidah yang umum dalam AI, yaitu pengetahuan dan inferensi. Kaidah pengetahuan atau kaidah deklaratif menyatakan semua fakta dan hubungan tentang suatu permasalahan. Kaidah inferensi atau kaidah prosedural pada sisi lain merupakan nasihat atau saran tentang bagaimana menyelesaikan suatu masalah yang diberikan dengan fakta tertentu yang diketahui.

Adapun kaidah inferensi yang berisi kaidah tentang kaidah dinamakan *metarule* yaitu menyinggung kaidah lain. Perekayasa pengetahuan memisahkan kaidah pengetahuan menjadi basis pengetahuan dan kaidah inferensi menjadi bagian dari mesin inferensi.

Representasi kaidah terutama dapat diaplikasikan bila dibutuhkan untuk merekomendasikan suatu bagian aksi berdasarkan kejadian yang dapat diobservasi. Ada beberapa keuntungan penggunaan kaidah, yaitu:

1. Kaidah mudah mengerti. Mudah disampaikan karena merupakan bentuk alami dari pengetahuan.
2. Inferensi dan penjelasan mudah diperoleh atau diturunkan.
3. Modifikasi dan perawatan relatif lebih mudah.

4. Ketidakpastian lebih mudah dikombinasikan dengan kaidah.
5. Setiap kaidah sering saling *independent* dari semua kaidah.

Selain itu terdapat pula keterbatasan dari representasi kaidah, yaitu:

1. Pengetahuan yang kompleks membutuhkan beribu-ribu kaidah, yang mungkin agak sukar membuatnya, baik untuk menggunakan sistem maupun untuk perawatannya.
2. Pembangun menyukai kaidah; oleh karenanya mereka mencoba kekuatan semua pengetahuan kedalam kaidah dibandingkan mencari representasi yang lebih sesuai.
3. Sistem dengan banyak kaidah mungkin mempunyai batasan pencarian dalam kontrol program. Beberapa program mempunyai kesulitan dalam mengevaluasi sistem berbasis kaidah dan membuat inferensi.

Pada tabel 2.2 merupakan penjelasan dari perbedaan-perbedaan karakteristik dari representasi kaidah yang terdapat dalam artificial intelligence.

Tabel 2.2 Karakteristik dari Representasi Kaidah

	Bagian Pertama	Bagian Kedua
Nama	Premis Antecedent Situasi IF	Konklusi Konsekuen Aksi THEN
Alami	Kondisi, sama dengan pengetahuan deklaratif	Resolusi, sama dengan pengetahuan prosedural
Ukuran	Dapat mempunyai banyak IF	Biasanya hanya mempunyai satu konklusi
Pernyataan	Pernyataan AND Pernyataan OR	Semua kondisi harus benar untuk konklusi benar Jika ada kondisi pernyataan OR benar maka konklusinya benar

2.2.6 Forward Chaining

Mesin inferensi merupakan 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.

Terdapat dua pendekatan untuk mengontrol inferensi dalam sistem pakar berbasis aturan, yaitu pelacakan ke belakang (*backward chaining*) dan pelacakan ke depan (*forward chaining*). Pelacakan ke belakang adalah pendekatan yang dimotori tujuan (*goal driven*). Dalam pendekatan ini pelacakan dimulai dari tujuan, selanjutnya dicari aturan yang memiliki tujuan 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.

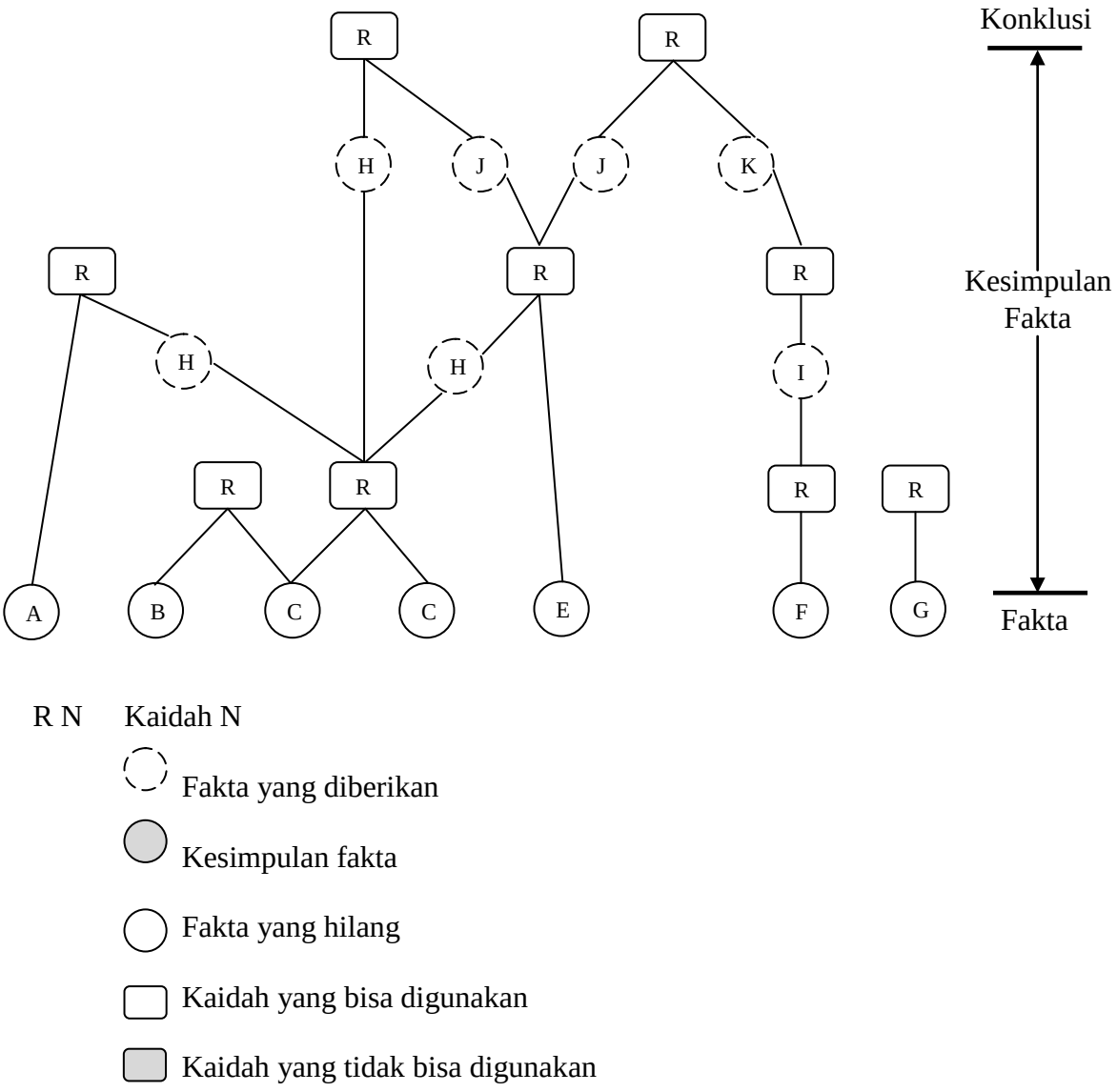
Suatu perkalian inferensi yang menhubungkan suatu permasalahan dengan solusinya disebut dengan rantai (*chain*). Suatu rantai yang dicari atau dlewati/dilintasi dari suatu permasalahan untuk memperoleh solusinya disebut *forward chaining*. Cara lain menggambarkan forward chaining adalah dengan penalaran dari fakta menuju konklusi yang terdapat dari fakta. Sedangkan suatu rantai yang dilintasi dari suatu hipotesa kembali ke fakta yang mendukung hipotesa tersebut adalah *backward chaining*.

Forward Chaining disebut juga penalaran dari bawah ke atas karena penalaran dari *evidence* (fakta) pada level bawah menuju konklusi pada level atas

didasarkan pada fakta. Penalaran dari bawah ke atas dalam suatu sistem pakar dapat disamakan untuk pemrograman konvensional dari bawah ke atas. Fakta merupakan satuan dasar dari paradigma berbasis pengetahuan karena mereka tidak dapat diuraikan ke dalam satuan paling kecil yang mempunyai makna. [5]

Berikut adalah karakteristik metode inferensi forward chaining :

1. Perencanaan, monitoring, dan kontrol.
2. Disajikan untuk masa depan.
3. Antecedent ke konsekuen.
4. Antecedent menentukan pencarian.
5. Data memandu, penalaran dari bawah ke atas.
6. Bekerja ke depan untuk mendapatkan solusi apa yang mengikuti fakta.
7. Breadth first search dimudahkan.
8. Penjelasan tidak difasilitasi.



Gambar 2.8 Forward Chaining

2.2.7 Best First Search

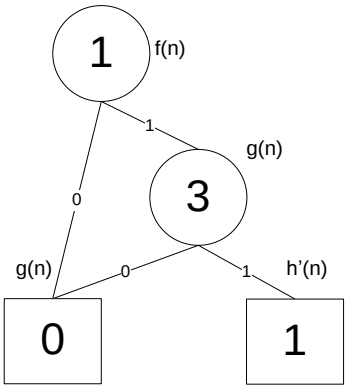
Sesuai dengan namanya, *Best First Search* membangkitkan simpul berikutnya dari sebuah simpul (yang sejauh ini) terbaik diantara semua *leaf nodes* yang pernah dibangkitkan. Penentuan simpul terbaik dapat dilakukan

menggunakan informasi berupa biaya perkiraan dari suatu simpul menuju ke *goal* atau gabungan biaya sebenarnya dan biaya perkiraan tersebut. Biaya perkiraan dapat diperoleh menggunakan suatu fungsi yang disebut fungsi heuristik. Oleh karena itu algoritma *Best First Search* ini digolongkan sebagai *heuristic search* atau *informed search* (pencarian dengan berbekal informasi yang berupa biaya perkiraan).

Terdapat dua jenis algoritma *Best First Search,* yaitu: 1) algoritma *Graf Or* yang hanya memperhitungkan biaya perkiraan saja; dan 2) algoritma A* yang memperhitungkan gabungan dua biaya, biaya sebenarnya dan biaya perkiraan.

2.2.8 Graf Or Algorithm

Graf or merupakan *function boolean* pada sebuah penyelesaian masalah dalam penelusuran, pencarian, dan pelacakan sebuah kasus. Fungsi ini hanya memiliki 2 nilai, yaitu true dan false (1 dan 0). Dalam penggambaran grafik, penelusuran akan selalu berawal dari *root* (level 1) dan kemudian akan berlanjut ke *leaf* hingga akhirnya menuju *goal* (level n+1) (Lihat Gambar 2.9).



Gambar 2.9 Contoh Graf Or

Pada Graf Or yang berawal dari *root*, alur graf berisi vertek ‘V’ dimana terdapat 2 tipe node, yaitu node dengan *value* 1 (true) dan node dengan *value* 0 (false). Tiap node akan dievaluasi sesuai dengan nilai evaluasi heuristiknya. Node dengan nilai heuristik terbaik akan menjadi *terminal vertek* untuk melanjutkan pencarian selanjutnya dengan 2 *leaf*. Demikian seterusnya sampai ditemukannya *goal*. Pada gambar 2.3 membutuhkan 2 antrian yang berisi node-node sebagai berikut:

1. OPEN, yang berisi node-node yang sudah dibangkitkan, sudah memiliki fungsi heuristik namun belum diuji. Umumnya berupa antrian berprioritas yang berisi elemen-elrmen dengan nilai heuristik tertinggi.
2. CLOSED, berisi node-node yang sudah diuji.

Disamping 2 antrian tersebut, terdapat fungsi heuristik yang akan mengestimasi seberapa baik dibangkitkannya setiap node. Fungsi f’(n) merupakan pendekatan dari fungsi f(n) yang merupakan fungsi evaluasi yang sebenarnya terhadap node n. biasanya fungsi f’(n) merupakan kombinasi dari 2 komponen, yaitu g(n) dan h’(n). fungsi g(n) merupakan biaya yang dikeluarkan dari keadaan awal sampai ke node n. nilai yang diperoleh g(n) bukan merupakan hasil estimasi, namun merupakan nilai eksak yang diperoleh dengan cara menjumlahkan total biaya yang telah dikeluarkan dari keadaan awal sampai ke node n. Apabila ada beberapa lintasan dari keadaan awal ke node g tersebut, maka diambil lintasan terbaiknya. Fungsi h’(n) merupakan estimasi tambahan biaya yang harus dikeluarkan dari node n sampai mendapatkan tujuan. [5]

Secara informal, *pseudocode* algoritma *Graf Or* diilustrasikan sebagai berikut:

1. *OPEN* berisi *initial state* dan *CLOSED* masih kosong.
2. Ulangi sampai *goal* ditemukan atau sampai tidak ada *nodes* didalam *OPEN*:
    a) Ambil simpul terbaik yang ada di *OPEN*.
    b) Jika simpul tersebut sama dengan *goal*, maka sukses.
    c) Jika tidak, masukan simpul tersebut kedalam *CLOSED*.
    d) Bangkitkan semua suksesor dari simpul tersebut.
    e) Untuk setiap suksesor kerjakan:
        i. Jika suksesor tersebut belum pernah dibangkikan, evaluasi suksesor tersebut, tambahkan ke *OPEN*, dan catat *parent*-nya.
        ii. Jika suksesor tersebut sudah pernah dibangkitkan, ubah *parent*-nya jika jalur melalui *parent* ini lebih baik daripada jalur melalui *parent* yang sebelumnya. Selanjutnya, perbaharui biaya untuk suksesor tersebut dan *nodes* lain yang berada dilevel bawahnya.

Pada algoritma tersebut diatas, *OPEN* adalah senarai (list) yang digunakan untuk menyimpan simpul-simpul yang pernah dibangkitkan dan nilai heuristiknya telah dihitung tetapi belum terpilih sebagai simpul terbaik (*best node*). Dengan kata lain, *OPEN* berisi simpul-simpul yang masih memiliki peluang (peluangnya masih terbuka) untuk terpilih sebagai simpul terbaik. Sedangkan *CLOSED* adalah senarai untuk menyimpan simpul-simpul yang pernah dibangkitkan dan sudah pernah terpilih sebagai simpul terbaik. Artinya, *CLOSED*

berisi simpul yang tidak mungkin terpilih sebagai simpul terbaik (peluang untuk terpilih sudah tertutup). [9]

2.2.8.1 Mencari Nilai Evaluasi Terbaik dengan Teorema Probabilitas

Nilai evaluasi terbaik didapat dari probabilitas, yang didefinisikan sebagai peluang P(A) dengan n adalah banyaknya kejadian, nA merupakan banyaknya hasil mendapatkan A. frekuensi relatif terjadinya A adalah ***n*(*A*)/n**, maka:

$$\mathbf{P(A) = \lim_{n \to \sim} = \frac{n(A)}{n}} \text{ (}\textit{posteriori probability}\text{)}$$

P(A) sendiri adalah frekuensi relatif jangkan panjang terjadinya A. Atau:

P(kejadian) = *jumlah kejadian* / *total jumlah percobaan* (*posteriori probability)*

Sebagai contoh pada kasus dengan 1-67 (67 gejala) dan A-AP (42 kerusakan). Untuk mendapat probabilitas gejala 1 dilihat dari banyaknya gejala yang keluar pada kerusakan A-AP, maka didapat:

P(Gejala 1) = 10/42 = 0,2381

A = Gejala 1

n(A) = jumlah gejala yang timbul pada kerusakan

n = total jumlah kerusakan

Maka didapat probabilitas dari Gejala 1 adalah 0,2381 dan Gejala 1 ditetapkan menjadi *root* karena memiliki nilai probabilitas evaluasi terbaik dalam melakukan penelusuran nantinya.

2.3 Software Pendukung

Pembangunan aplikasi didukung oleh beberapa software sebagai media pembangunan aplikasi. Diantaranya Turbo Prolog 2.0 dengan bahasa pemrograman Prolog, Delphi 7 sebagai media interface update gejala, diagnosis, dan solusi, dan menyertakan MySQL sebagai media databasenya.

2.3.1 Prolog

Prolog singkatan dari Programming in Logic yang dikembangkan oleh Alain Colmenraurer dan P.Roussel di Universitas Marseilles Perancis, tahun1972. Prolog populer di Eropa untuk aplikasi *artificial intelligence*, sedangkan di Amerika peneliti mengembangkan aplikasi yang sama, yaitu LISP. Adapun perbedaan bahasa Prolog dengan Bahasa Pemrograman Umum lainnya (seperti : Basic, Pascal, C, Fortran), yaitu:

1. Tidak diperlukan algoritma/prosedur untuk memecahkan masalah (*procedural language*).
2. Program tidak menjalankan prosedur yang sama berulang-ulang dengan data masukan yang berbeda-beda.
3. Prosedur dan pengendalian program tidak ditentukan oleh programer dan perhitungan tidak dilakukan sesuai dengan prosedur yang telah dibuat.

Karakteristik Bahasa Pemrograman Prolog adalah:

1. Predikat.

   Prolog membutuhkan fakta-fakta yang terungkap dalam relasi dan sifat untuk mencari jawaban.

2. Basis data.

   Sebagai bahasa yang berorientasi pada fakta dan untuk menunjang basis pengetahuan, prolog menyediakan fasilitas basis data.

3. Deduksi.

   Prolog dapat menarik kesimpulan.

4. Pemadanan dan Unifikasi.

   Tujuan melakukan pemadanan adalah untuk melakukan Unifikasi, yaitu penyatuan argumen dalam suatu predikat dengan predikat yang lain.

5. Pembuktian Mundur.

   Pemilihan anggapan suatu fakta terlebih dulu yang kemudian dilakukan pembuktian. Cara ini disebut pembuktian mundur (*backward chaining*) atau juga disebut metode *top-down* karena berawal dari kenyataan pokok menuju masalah rinci.

6. Lacak Balik (*backtracking*).

   Prolog melacak suatu fakta atau aturan dengan melakukan pemadanan (*matching*) berturut-turut. Jika menemui kegagalan langkah berikutnya adalah mencari anggapan baru dengan definisi baru.

7. Rekursi.

   Pemecahan masalah dengan menguraikan masalah tadi menjadi bagian yang lebih kecil tetapi memiliki struktur atau model yang sama.

8. Pengolahan dan pemanipulaisan simbol.

9. Meta programming.

   Program dapat diubah prosedurnya atau dalam hal ini logikanya pada saat program dijalankan sekalipun.

10. Pararel.

    Prolog merupakan bahasa yang berstruktur pararel dari sifatnya. Ini karena beberapa klausa yang sama dapat dieksekusi secara bersamaan (*concurrent*).

Adapun bagian-bagian dalam pemrograman prolog, yaitu:

1. Fakta dan Relasi

   Prolog terdiri dari kumpulan data-data objek yang merupakan suatu fakta. Fakta itu sendiri dibedakan menjadi 2 macam, yaitu fakta yang menunjukan relasi dan fakta yang menunjukan milik/sifat. Dalam hal penulisannya selalu diakhiri dengan tanda titik ".".

2. Aturan (Rules)

   Aturan adalah suatu pernyataan yang menunjukkan bagaimana fakta-fakta berinteraksi satu dengan yang lain untuk membentuk suatu kesimpulan. Sebuah aturan dinyatakakan sebagai suatu kalimat bersyarat. Kata *if* adalah kata yang dikenal Prolog untuk menyatakan kalimat bersyarat atau disimbolkan dengan ":-". Setiap aturan terdiri dari kesimpulan (kepala) dan tubuh. Tubuh dapat terdiri dari 1 atau lebih pernyataan atau aturan yang lain yan disebut *subgoal*

dan dihubungkan dengan logika *and* dan aturan juga memiliki sifat *then/if conditional.*

3. Pertanyaan (Query)

   Setelah memberikan data-data berupa fakta dan aturan, selanjutnya dapat mengajukan pertanyaan berdasarkan fakta dan aturan yang ada. Penulisannya diawali simbol "?-" dan diakhiri tanda ".".

4. Predikat

   Predikat adalah nama simbolik untuk relasi. Sebuah predikat dapat tidak memiliki atau memiliki argumen dengan jumlah bebas. Jumlah argumen suatu predikat disebut aritas (*arity*). Beberapa syarat dalam penulisan nama predikat, yaitu:

   a. Harus diawali dengan huruf kecil dan dapat diikuti dengan huruf, bilangan atau garis bawah.
   b. Panjang nama predikat maksimum 250 karakter.
   c. Tidak diperbolehkan menggunakan spasi, tanda minus, tanda bintang dan garis miring.

5. Variabel

   Variabel adalah besaran yang nilainya dapat berubah-ubah. Tata cara penulisan variabel:

   a. Nama variabel harus diawali huruf besar atau garis bawah (_).
   b. Nama variabel dapat terdiri dari huruf, bilangan, atau simbol dan merupakan kesatuan dengan panjang maksimum 250 karakter.

c. Nama variabel hendaknya mengandung makna yang berkaitan dengan data yang dinyatakannya. [1]

2.3.1.1 Struktur Program Turbo Prolog 2.0

Bahasa Pemrograman prolog dengan toolsnya yaitu Turbo Prolog 2.0 (Gambar 2.10) terbentuk atas bagian-bagian yang membentuknya, yaitu:

1. Domains.

   Domains dalam Turbo Prolog seperti tipe dalam *pascal*. Dibagian domains, menyatakan jenis suatu variabel atau argumen.

2. Komentar.

   Turbo Prolog mengenal komentar untuk mempermudah penelaahan program.

3. Predicates.

   Mendeklarasikan predikat yang ada pada bagian *predicates* dan mendefinisikannya dalam bagian *clauses*. Deklarasi predikat dalam bagian *predicates* harus disertai dengan argumen jika memang berargumen. Jumlah argumen dalam predikat disebut aritas (*arity*). Satu predikat boleh memiliki aritas yang berbeda dengan syarat harus dikelompokan dalam bagian klausanya.

4. Clauses.

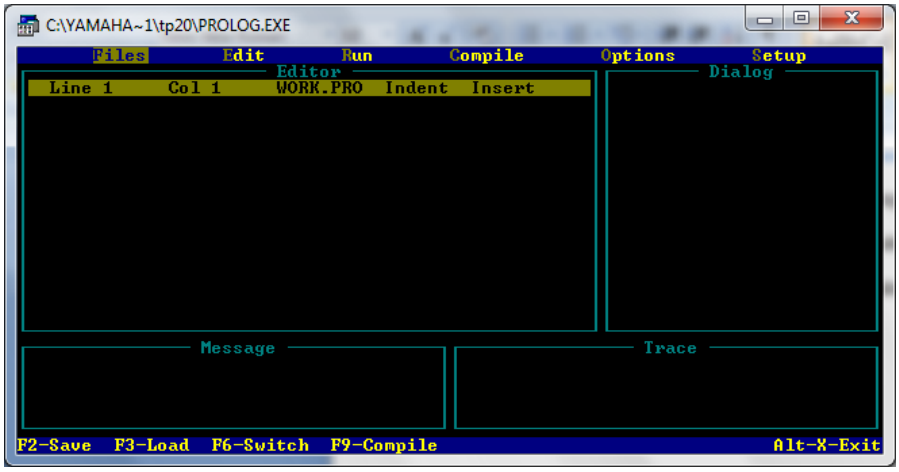
   Sekumpulan klausa dari predikat yang sama harus disatukan atau dikelompokan dalam bagian *clauses* yang disebut dengan prosedur. Bagian *clauses* merupakan inti program Turbo Prolog.

5. Goal.

   Goal dalam Turbo Prolog ada 2 macam, yaitu:

   1. Goal eksternal.
   2. Goal internal.

6. Database.
7. Constans.
8. Global.
9. Compiler.



Gambar 2.10 Turbo Prolog 2.0

2.3.2 Delphi

Delphi adalah sebuah bahasa pemrograman dan lingkungan pengembangan perangkat lunak. Produk ini dikembangkan oleh Borland. Bahasa Delphi, yang sebelumnya dikenal sebagai object pascal (pascal dengan ekstensi pemrograman berorientasi objek (PBO/OOP)) pada mulanya ditujukan hanya untuk Microsoft Windows, namun saat ini telah mampu digunakan untuk mengembangkan aplikasi untuk Linux dan Microsoft .NET framework .Dengan

menggunakan Free Pascal yang merupakan proyek *open source*, bahasa ini dapat pula digunakan untuk membuat program yang berjalan di sistem operasi Mac OS X dan Windows CE.

Pada lingkungan pengembangan Delphi, umumnya banyak digunakan untuk pengembangan aplikasi desktop dan enterprise berbasis database, tapi sebagai perangkat pengembangan yang bersifat *general-purpose*, ia juga mampu dan digunakan dalam berbagai jenis proyek pengembangan software. Ia juga yang dikenal sebagai salah satu yang membawa istilah RAD tool (Rapid Application Development), saat dirilis tahun 1995 untuk windows 16-bit. Pada tahun 2001 sebuah versi linux yang dikenal sebagai Kylix tersedia. Pada tahun 2002 dukungan untuk Linux (melalui Kylix dan CLX component library) ditambahkan dan tahun 2003 .NET dan mulai didukung dengan munculnya Delphi.Net (Delphi 8). Yang terbaru adalah versi .NET yaitu RAD Studio 2007 dan Turbo Delphi .NET dengan versi desktopnya yaitu Delphi 2007 for win 32 dan Turbo Delphi. Delphi 2005 (nama lain dari Delphi 9) yang mendukung *code generation* baik untuk win32 maupun .NET, dan seperti yang telah dikenal, fitur-fitur manipulasi data secara *live* dari database secara *design-time*. Dalam hal ini, Delphi juga membawa banyak pembaruan pada IDE secara signifikan. [4]

Perbedaan fitur yang utama antara Delphi, Kylix dengan IDE-IDE yang lain adalah keberadaan bahasanya (Bahasa pemrograman delphi), VCL/CLX (Visual Component Library), Penekanan konektifitas database yang sangat baik, dan banyaknya komponen-komponen pihak ketiga yang mendukungnya. Aspek penting yang perlu dicatat tentang Bahasa pemrograman Delphi termasuk, yaitu:

1. Penanganan objek sebagai reference/pointer secara transparan.
2. Properti sebagai bagian dari bahasa tersebut, sebagai getter dan setter (atau accessor and mutator) yang secara transparan mengenkapsulasi akses pada field-field anggota dalam kelas tersebut.
3. Properti index dan default yang menyediakan akses pada data kolektif.
4. Pendelegasian (type safe method pointer) yang digunakan untuk memproses kejadian yang dipicu oleh komponen.
5. Pendelegasian implementasi interface pada Field ataupun properti dari Class.
6. Implementasi penanganan windows message dengan cara membuat metode dalam Class dengan nomor/nama dari windows message yang akan ditangani.
7. COM bersifat sebagai interface yang independen dengan implementasi Class sebagai reference counted.
8. Kompilasi yang dapat menghasilkan kode yang berjalan secara native x86 ataupun *managed code* pada arsitektur framework .NET.

Berikut adalah karakteristik bahasa pemrograman Delphi, yaitu:

1. Tidak case sensitive

   Pemrograman delphi tidak case sensitive, artinya delphi tidak membedakan huruf besar dan huruf kecil.

2. Object based

   Delphi merupakan pemrograman berorientasi objek, artinya hampir seluruhnya merupakan objek. Seperti button yang merupakan komponen turunan dari objek dengan nama TObject. Objek ini merupakan objek utama dalam delphi. Hampir semua komponen diturunkan dari objek ini. Dari turunan objek pun

kita bisa menurunkan objek lain, misalkan kita membuat komponen yang diturunkan langsung dari button, maka komponen buatan kita memiliki properti yang sama dengan objek diatasnya yaitu button.

3. Pengembangan dari pemrograman bahasa pascal

   Sehingga bahasanya hampir mirip, tetapi memiliki kelebihan yang sangat banyak, seperti tipe data yang lebih fleksibel dan besar.

4. Modularitas

   Setiap aplikasi yang dibuat dengan delphi akan memiliki banyak sekali file modul yang terpisah. Terdapat pada clausa uses pada setiap unit.

2.3.2.1 Interface Delphi 7

Interface pemrograman Delphi (toolsnya yaitu Delphi 7) dibagi menjadi beberapa bagan, yaitu :

1. Palete Komponen

   Palete (toolbar) ini merupakan tempat untuk meletakkan komponen-kompoenen dalam Delphi.

2. Object Tree View

   Window ini digunakan untuk melihat komponen apa saja yang digunakan dalam form. Setiap komponen yang berada dalam form akan terlihat.

3. Object Inspector

   Window ini digunakan untuk menampilkan properti dari object. Setiap object dalam delphi memiliki properti.

4. Code Editor

Disinilah kita akan dituliskan kode program. Secara otomatis delphi akan membuat struktur dari program unit ini, seperti penulisan clausa uses, unit, type dll.

5. Form Designer

Jika code editor digunakkan untuk ‘mendesain’ code yang diketikkan, maka form designer digunakan untuk mendesain tampilan dari aplikasi yang akan dibuat. Ketika delphi di load (dibuka) maka akan diberikan sebuah form kosong. Terlihat dalam contoh sebuah form dengan sebuah komponen tombol

Peran Delphi dalam kaitannya dengan prolog adalah sebagai media interface dalam melakukan update gejala, kerusakan, dan solusi sebagaimana prolog dalam tampilan Under DOS dapat ditangani dengan menggunakan Delphi yang telah terintegrasi dengan mysql sebagai databasenya. Sistem update disini, merupakan update query pada file prolog. Update yang tersimpan pada mysql akan diinsert pada baris file .PRO ketika melakukan update pada Delphi.

2.3.3 MySQL

MySQL bukan merupakan lingkungan penegmbangan aplikasi basis data, tetapi satu DBMS yang kompak. MySQL adalah DBMS kecil, kompak dan mudah digunakan. MySQL ideal untuk aplikasi berukuran kecil dan menengah, namun telah menjanjikan untuk penggunaan besar. [10]

Adapun kelebihan dari MySQL untuk memilih MySQL dalam penggunaan database, yaitu:

1. Kecepatan
2. Kemudahan penggunaan
3. Ongkos
4. Dukungan bahasa query
5. Kapabilitas
6. Konektivitas dari keamanan
7. Portabilitas
8. Distribusi terbuka

Beberapa karakteristik MySQL adalah sebagai berikut:

1. Standar

   Mendukung *entry-level* ANSI SQL92 ODBC level 0-2.

2. Character Set

   Secara *default* menggunakan ISO-8859-I (Latin I) character set untuk data pengurutan.

3. Bahasa Pemrograman

   Mendukung pemrograman aplikasi dalam bahasa Java, Delphi, PHP, dll.

4. Tabel Besar

   Menyimpan tiap relasi (tabel) pada file terpisah didirektori basis data. Ukuran maksimum tabel dibatasi kemampuan sistem operasi menangani ukuran file

5. Kecepatan dan Kemudahan Pemakaian

   Lebih cepat dibanding basis data komersial dan mudah untuk dikelola.

6. MySQL

   Merupakan *open source relational* DBMS.