BAB II

LANDASAN TEORI

2.1 *Intelligence*

Menurut Suparman (1991:2) Intelligence atau intelegensia adalah kemampuan manusia untuk memperoleh pengetahuan dan pandai melaksanakannya dalam praktek. Hal ini berarti kemampuan berfikir dan menalar. Pada batas-batas tertentu *Artificial Intelligence* (AI) memungkinkan komputer bisa menerima pengetahuan melalui *input* manusia menggunakan pengetahuannya itu melalui simulasi proses penalaran dan berfikir manusia untuk memecahkan berbagai masalah.

Bagian utama untuk *Artificial Intelligence* adalah pengetahuan (*knowledge*), suatu pengertian tentang beberapa wilayah subjek yang diperoleh melalui pendidikan dan pengalaman. Pengetahuan terdiri dari fakta, pemikiran, teori, prosedur dan hubungan satu sama lain. Pengetahuan juga merupakan informasi terorganisasi dan teranalisa agar bisa lebih diterapkan pada pemecahan masalah dan pengambilan keputusan.

Dengan basis pengetahuan (*Knowledge Base*) dan kemampuan untuk menarik kesimpulan melalui pengalaman (*inferensi*), komputer dapat disejajarkan sebagai alat bantu yang bisa digunakan secara praktis dalam memecahkan masalah dalam pengambilan keputusan.

2.2 Sistem Pakar

Menurut Arhami (2005:3) sistem pakar adalah suatu cabang dari AI yang membuat penggunaan secara luas *knowledge* yang khusus untuk penyelesaian

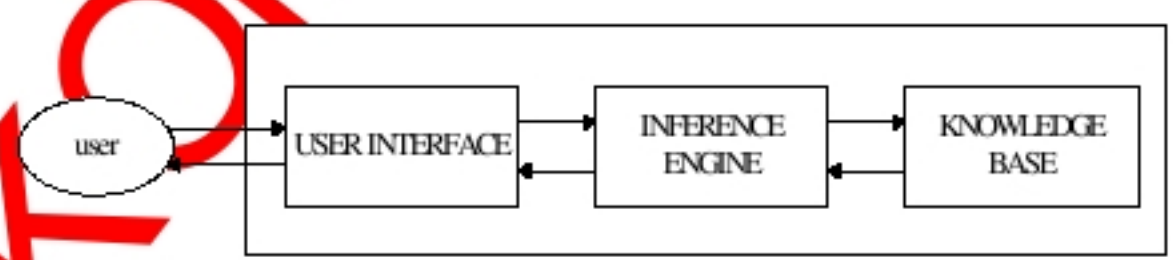
STIKOMPS SURABAYA

masalah tingkat manusia yang pakar. Sistem pakar bertindak sebagai penasehat atau konsultan pintar dengan mengambil pengetahuan yang di simpan dalam *Knowledge Base*. Seorang pemakai yang tidak berpengalaman sekalipun asalkan mengetahui secara umum tentang cara kerja peralatan yang didiagnosa bisa memecahkan suatu masalah yang rumit dan bisa mengambil keputusan yang tepat dan akurat layaknya yang dilakukan seorang pakar.

Menurut Suparman (1991:13) tujuan utama sistem pakar bukan untuk mengganti kedudukan seorang ahli atau seorang pakar, tetapi hanya untuk memasyarakatkan pengetahuan dan pengalaman pakar-pakar yang sangat langka itu. Sistem pakar memungkinkan orang lain bisa meningkatkan produktivitasnya dan memperbaiki kualitas keputusannya. Pada dasarnya sistem pakar bisa memecahkan masalah yang rumit, sekalipun tidak ada seorang ahli.

2.3 Komponen Sistem Pakar

Sistem pakar terdiri dari tiga komponen utama, yaitu *Knowledge Base*, *User Interface,* dan *Inference Engine*. Komponen sistem pakar dapat digambarkan pada Gambar 2.1



Gambar 2.1 Diagram Blok Umum Sistem Pakar

1. *Knowledge Base*

Knowledge Base adalah data atau pengetahuan yang diperlukan untuk membuat sesuatu keputusan dimana di dalam *Knowledge Base* tersebut terdapat fakta

STIKOMP SURABAYA

dan juga teknik dalam menerangkan masalah yang menjelaskan bagaimana fakta tersebut cocok dengan yang lainnya dalam urutan yang logis.

1. Fakta: adalah suatu kenyataan atau kebenaran yang diketahui. Fakta menyatakan hubungan antara dua objek atau lebih. Fakta dapat pula menunjukkan sifat atau objek.
2. *Rule* atau aturan: Salah satu skema representasi *knowledge* yang sangat populer adalah *rule* dengan format IF-THEN. *Rule* pada bagian IF disebut kondisi (disebut juga premis) yang mungkin benar atau tidak, dan bagian THEN disebut aksi (disebut juga konklusi). *Rule* pada bagian THEN akan menggantikan kondisi bagian IF bila ternyata sudah sesuai. Statement THEN akan bernilai benar jika statement IF bernilai benar.

2. *User Interface*

Merupakan bagian *software* yang menyediakan sarana untuk *user* agar bisa berkomunikasi dengan sistem. *User Interface* menyediakan bagi *user* kemudahan dalam berkomunikasi dengan program pintar. *User Interface* akan mengajukan pertanyaan, selain juga menyediakan menu pilihan untuk memasukkan informasi awal dalam *Knowledge Base*. Kemudian ia juga menyediakan sarana komunikasi jawaban atau solusi bila masalahnya sudah ditemukan. Setiap komunikasi antara dan selama proses pemecahan masalah dikendalikan oleh *User Interface*.

3. *Inference Engine*

Inference Engine merupakan alat operasi pelacakan dan pencocokan pola dengan cara menguji *rule* dalam urutan tertentu untuk mencari yang sesuai dengan kondisi awal dan kondisi berjalan yang sudah dimasukkan ke dalam database. Begitu *rule* cocok dengan kondisi yang ditemukan, maka *rule* terstimulasi, dan kemudian mulailah langkah selanjutnya ditentukan.

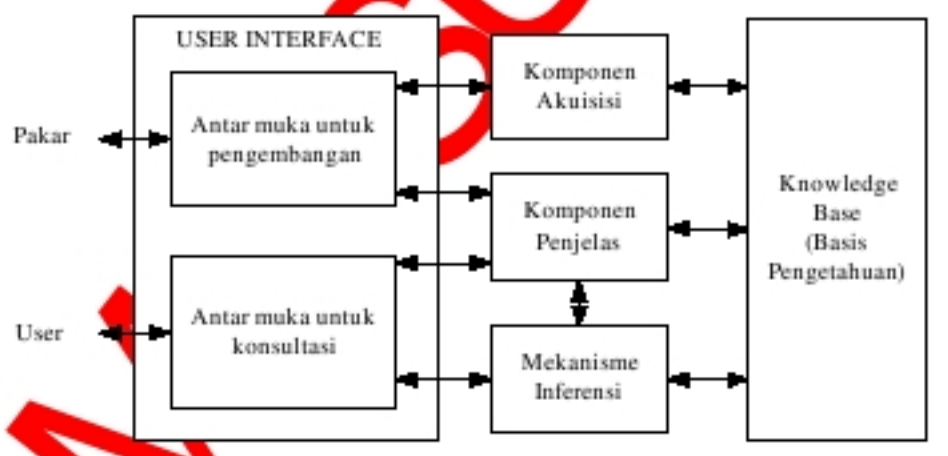
STIKOMPS SURABAYA

2.4 Sistem Berbasis Aturan

Sistem berbasis aturan merupakan suatu sistem pakar yang menggunakan aturan-aturan untuk menyajikan pengetahuannya. Dengan demikian sistem berbasis aturan dapat dikatakan suatu perangkat lunak yang menyajikan keahlian pakar dalam bentuk aturan-aturan pada suatu domain tertentu untuk menyelesaikan suatu permasalahan.

2.5 Komponen Sistem Berbasis Aturan

Ada beberapa komponen yang terdapat pada sistem berbasis aturan. Komponen-komponen tersebut adalah basis pengetahuan, mekanisme inferensi, komponen penjelas, *User Interface* dan komponen akuisisi. Komponen sistem berbasis aturan dapat digambarkan pada Gambar 2.2.



Gambar 2.2 Komponen Sistem Berbasis Aturan

1. Basis Pengetahuan

Basis pengetahuan yaitu pengetahuan yang menjadi dasar bagi pembuatan aturan-aturan dalam sistem, yang mencakup aturan-aturan itu sendiri, fakta-fakta yang terkait serta atribut-atributnya. Perbedaan antara fakta dan aturan dapat dilihat pada contoh berikut:

R1: IF p AND q THEN r → aturan (*rule*)

STIKOM SURABAYA

p, q, r	→	fakta
p dan q	→	premis
r	→	konklusi

2. Mekanisme Inferensi

Mekanisme inferensi adalah bagian dari sistem pakar yang melakukan penalaran dengan menggunakan isi daftar aturan berdasarkan urutan dan pola tertentu. Selama proses konsultasi antar sistem dan pemakai, mekanisme inferensi menguji aturan satu demi satu sampai kondisi aturan itu benar.

Secara umum ada dua teknik utama yang digunakan dalam mekanisme inferensi untuk pengujian aturan, yaitu penalaran maju (*Forward Chaining*) dan penalaran mundur (*Backward Chaining*).

3. Komponen Penjelas

Komponen penjelas merupakan bagian dari sistem pakar yang memberikan penjelasan tentang bagaimana program dijalankan, apa yang harus dijelaskan kepada pemakai tentang suatu masalah, memberikan rekomendasi kepada pemakai, mengakomodasi kesalahan pemakai dan menjelaskan bagaimana suatu masalah terjadi.

4. *User Interface*

User Interface yaitu bagian program yang berhubungan langsung dengan pemakai, baik selama konsultasi maupun untuk pengembangan sistem. Oleh karena itu sistem haruslah mengunakan bahasa dan sistem pengoperasian yang mudah dimengerti.

STIKOMPS SURABAYA

5. Komponen Akuisisi

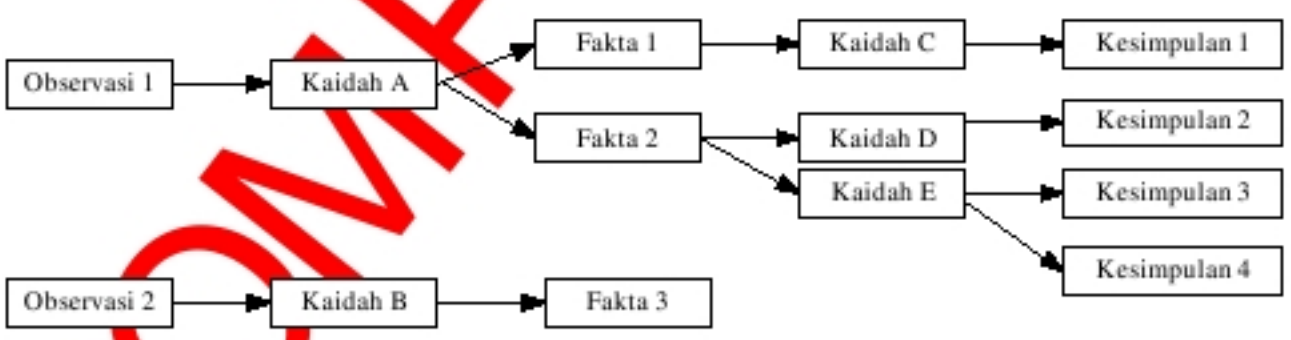
Komponen akuisisi berfungsi untuk menyusun dan mengimplementasikan pengetahuan dalam basis pengetahuan. Komponen ini mempunyai karakteristik antara lain:

a. Pengetahuan yang terdiri dari aturan dan fakta yang harus mudah untuk dimasukkan.

b. Metode penyajian informasi dalam basis pengetahuan harus mudah dimengerti.

2.6 *Forward Chaining*

Menurut Arhami (2005:20) *Forward Chaining* adalah pendekatan yang dimotori data (*data-driven*). Dalam pendekatan ini pelacakan dimulai dari informasi masuken, dan selanjutnya mencoba menggambarkan kesimpulan. Pelacakan ke depan mencari fakta yang sesuai dengan bagian IF dari aturan IF-Then. Untuk lebih jelasnya dapat dilihat alur dari metode *Forward Chaining* pada Gambar 2.3



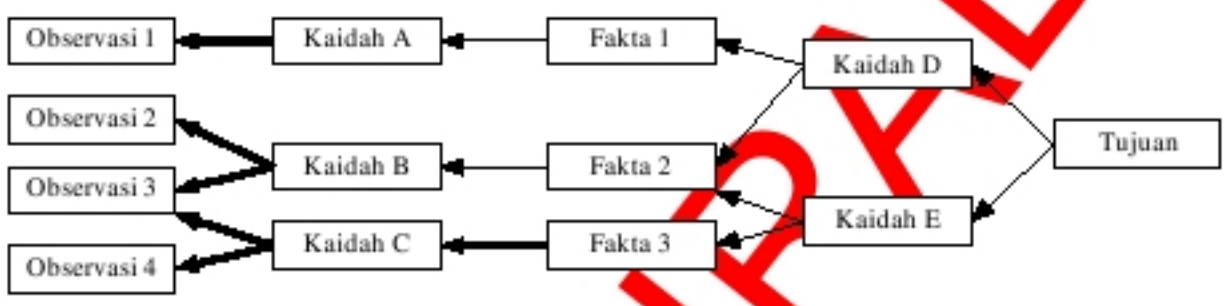
Gambar 2.3 Metode *Forward Chaining*

2.7 *Backward Chaining*

Untuk *Backward Chaining* (penalaran mundur) merupakan kebalikan dari *Forward Chaining* (penalaran maju), *Backward Chaining* akan diuji dari hipotesis akhir sampai pada pembuktiannya.

STIKOMPS SURABAYA

Menurut Arhami (2005:19) *Backward Chaininng* adalah pendekatan yang dimotori tujuan. 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 ditentukan. Untuk lebih jelasnya dapat dilihat alur dari metode *Backward Chaining* pada Gambar 2.4.



Gambar 2.4 Metode *Backward Chaining*

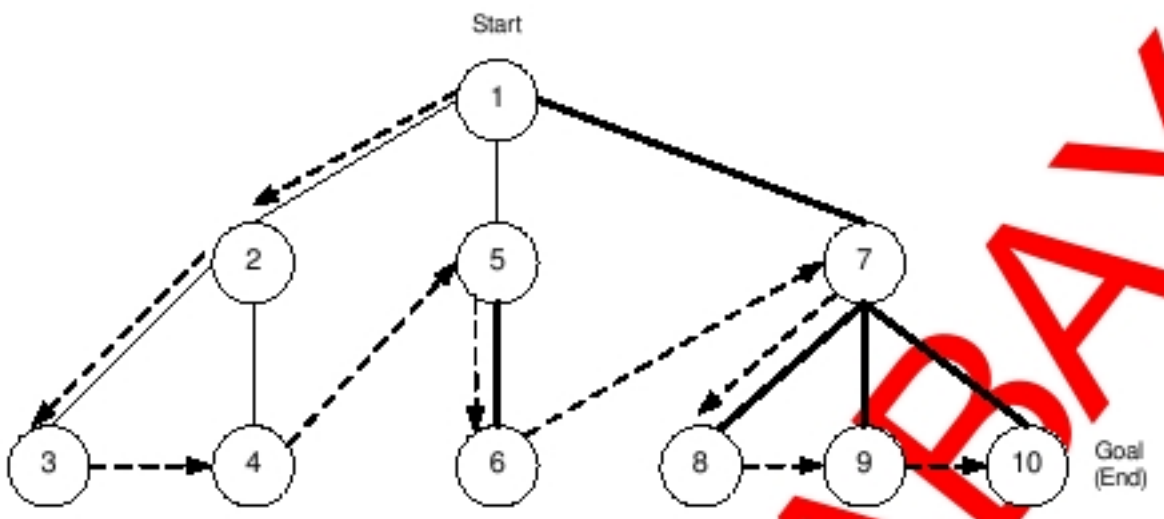
2.8 *Depth First Search*

Menurut Schildt (1987:29) *Depth First Search* adalah teknik melacak tiap *path* yang mungkin untuk kesimpulan (atau goal) sebelum melacak path yang lain. Pelacakan ini bermula dari node akar dan bergerak ke bawah (menurun) ke tingkat dalam yang berurutan. Suatu operator diterapkan pada *node* yang menimbulkan *node* berikutnya dalam satu urutan. Dengan lain perkataan, *node* anak atau *node* keturunannya langsung terus sampai solusi diketemukan atau jika menemui jalan buntu ia melacak ke belakang (*backtracking*).

Keuntungan dengan menggunakan teknik ini bahwa penelusuran masalah dapat digali secara mendalam sampai ditemukannya kepastian suatu solusi yang optimal. Kekurangan dalam teknik penelusuran ini adalah membutuhkan waktu yang

STIKOMP SURABAYA

sangat lama untuk ruang lingkup masalah yang besar. Model *Depth First Search* dapat digambarkan pada Gambar 2.5.



Gambar 2.5 Diagram Alir *Depth First Search*

2.9 *Breadth First Search*

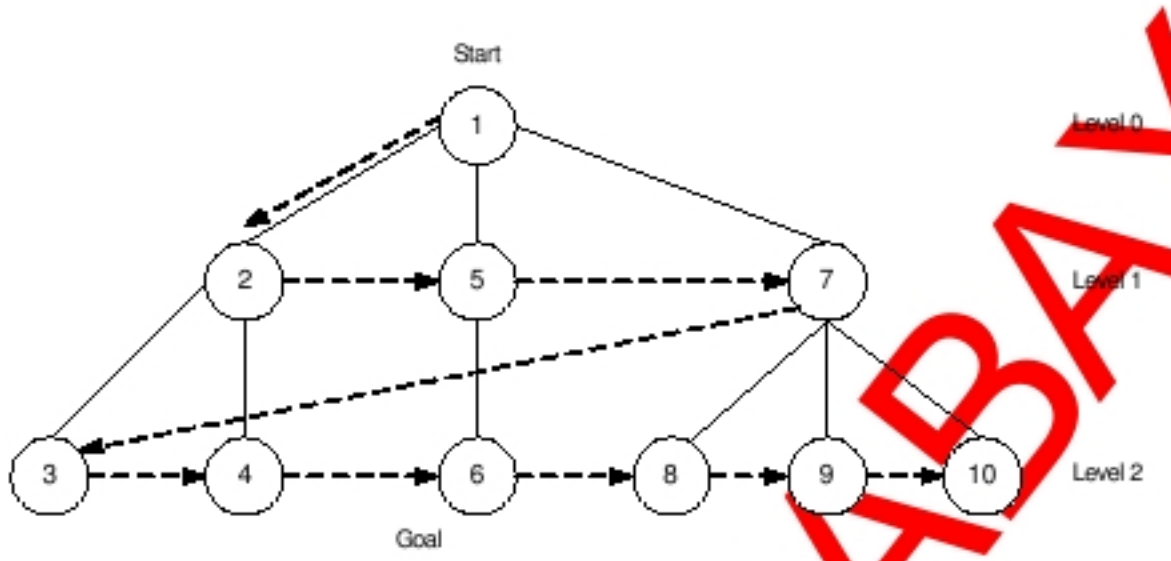
Menurut Schildt (1987:40) *Breadth First Search* adalah kebalikan dari *Depth First Search*. *Breadth First Search* memeriksa setiap node di level yang sama sebelum diproses ke level berikutnya.

Keuntungan pencarian dengan teknik ini adalah sama dengan *Depth First Search*, hanya saja penelusuran dengan teknik ini ruang keadaan akan menemukan jalan terpendek antara keadaan awal dan keadaan tujuan dengan jumlah langkah yang sedikit, di mana semua *node* akan dicek secara menyeluruh pada setiap tingkatan *node*.

Kekurangan teknik penelusuran ini terletak pada waktu yang dibutuhkan yang sangat lama apabila solusi berada dalam posisi *node* terakhir sehingga menjadi tidak efisien. Kekurangan dalam implementasi juga dipertimbangkan, misalnya teknik penelusuran menjadi tidak interaktif antara pemakai dan sistem karena menyebabkan tidak adanya relasi antara satu topik dangan topik yang lain atau harus melompat dari

STIKOMPS SURABAYA

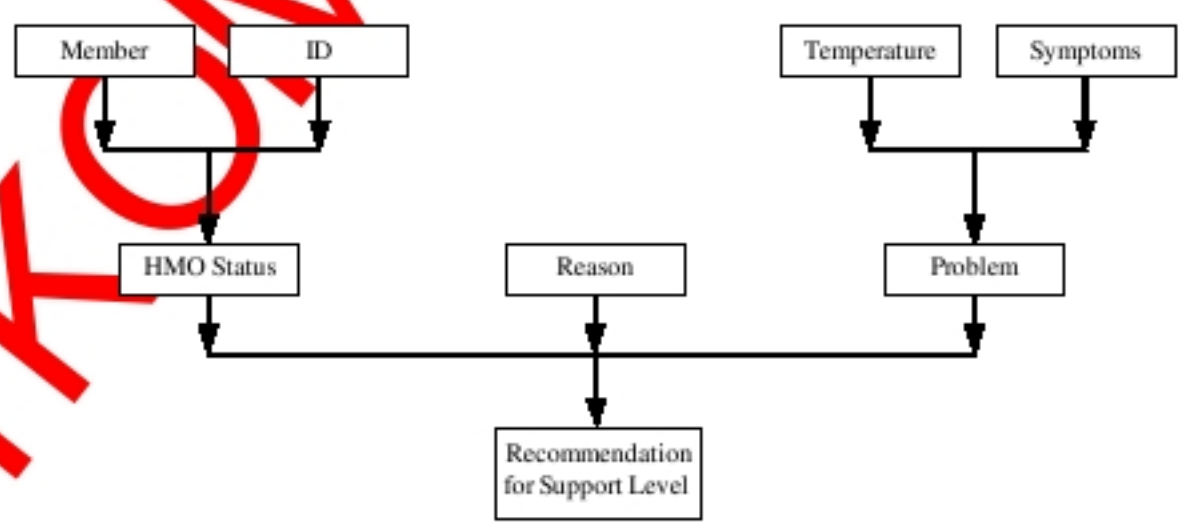
satu topik ke topik yang lain sebelum topik tersebut selesai ditelusuri. Model *Breadth First Search* dapat digambarkan pada Gambar 2.6.



Gambar 2.6 Diagram Alir *Breadth First Search*

2.10 *Block Diagram*

Block diagram (Diagram Blok) adalah struktur yang menggambarkan suatu permasalahan dalam suatu KBS secara hirarki. Menurut Dologite (1993:20) *block diagram* ini berisi faktor-faktor penting untuk membuat suatu rekomendasi dari KBS. Contoh *block diagram* dapat dilihat pada Gambar 2.7.

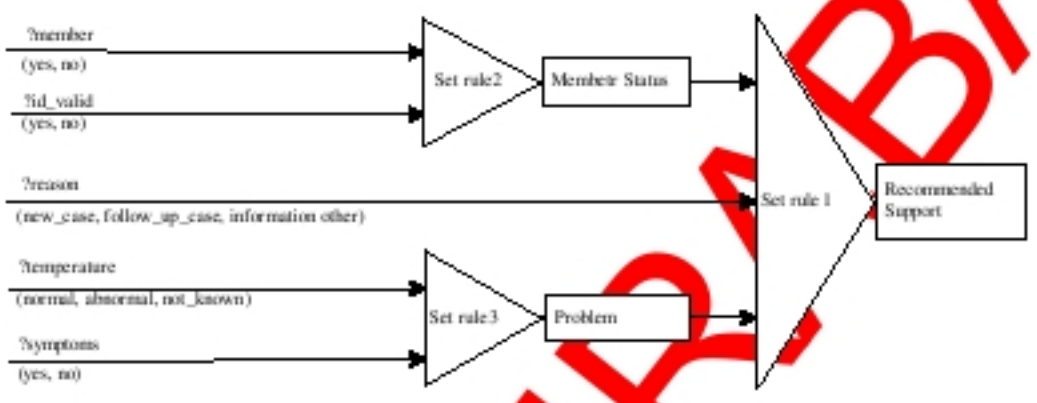


Gambar 2.7 Contoh *Block Diagram*

STIKOM SURABAYA

2.11 *Depedency Diagram*

Menurut Dologite (1993:22) *dependency diagram* diindikasikan sebagai hubungan antara faktor-faktor penting, memasukan pertanyaan-pertanyaan, aturan-aturan, nilai-nilai, dan rekomendasi yang dibuat prototipe KBS. *Dependency diagram* adalah hasil *convert* dari *block diagram*. Contoh *dependency diagram* dapat dilihat pada Gambar 2.8.



Gambar 2.8 Contoh *Dependency Diagram*

2.12 Verifikasi

Verifikasi merupakan sekumpulan aktifitas yang memastikan suatu sistem apakah telah berlaku dalam kondisi yang diterapkan. Verifikasi terdiri dari 2 proses yaitu:

1. Memeriksa pelaksanaan suatu sistem secara spesifik
2. Memeriksa konsistensi dan kelengkapan dari basis pengetahuan

Verifikasi dijalankan ketika ada penambahan atau perubahan pada *rule*, karena *rule* tersebut sudah ada pada sistem. Sedangkan tujuan verifikasi adalah untuk memastikan adanya kecocokan antara sistem dengan apa yang sistem kerjakan (*rule base*) dan juga untuk memastikan bahwa sistem itu terbebas dari *error*.

Berikut ini adalah beberapa metode pemeriksaan *rule* dalam suatu basis pengetahuan :

STIKOMP SURABAYA

1. *Redudant rules*

Redudant rules terjadi jika ada dua *rule* atau lebih mempunyai *premise* dan *conclussion* yang sama.

Contoh:

Rule 1 : If Batas atau level oli mesin rendah And
Kebocoran mesin bagian luar
Then Pemipaan bagian luar, saluran pembuangan oli, penyangga peralatan bagian luar, kebocoran pada kotak mesin dan kotak oli

Rule 2 : If Batas atau level oli mesin rendah And
Kebocoran mesin bagian luar
Then Pemipaan bagian luar, saluran pembuangan oli, penyangga peralatan bagian luar, kebocoran pada kotak mesin dan kotak oli

2. *Conflicting rules*

Conflicting rules terjadi ketika dua buah *rule* atau lebih mempunyai *premise* yang sama, tetapi mempunyai *conclusion* yang berlawanan.

Contoh:

Rule 1 : If Batas atau level oli mesin rendah And
Kebocoran mesin bagian luar
Then Pemipaan bagian luar, saluran pembuangan oli, penyangga peralatan bagian luar, kebocoran pada kotak mesin dan kotak oli

Rule 2 : If Batas atau level oli mesin rendah And
Kebocoran mesin bagian luar

Then Pengarah katub, tabung silinder, ring kontrol oli dan ring kompresi

3. *Subsumed rules*

Subsumed rules terjadi jika dua *rule* tersebut mempunyai *consraints* yang lebih atau kurang tetapi mempunyai *conclusion* yang sama.

STIKOMP SURABAYA

Contoh:

Rule 1 : If Batas atau level oli mesin rendah And
Kebocoran mesin bagian luar
Then Pemipaan bagian luar, saluran pembuangan oli, penyangga peralatan bagian luar, kebocoran pada kotak mesin dan kotak oli
Rule 2 : If Batas atau level oli mesin rendah
Then Pemipaan bagian luar, saluran pembuangan oli, penyangga peralatan bagian luar, kebocoran pada kotak mesin dan kotak oli

4. *Circular rules*

Circular rules adalah suatu keadaan dimana terjadinya proses perulangan dari suatu *rule*. Ini dikarenakan suatu *premise* dari salah satu *rule* merupakan *conclusion* dari *rule* yang lain, atau kebalikannya.

Contoh:

Rule 1 : If Rara dan Riri saudara kembar
Then Rara dan Riri mempunyai orang tua yang sama
Rule 2 : If Rara dan Riri mempunyai orang tua yang sama
Then Rara dan Riri saudara kembar

5. *Unnecessary IF conditions*

Unnecessary IF condition terjadi ketika dua *rule* atau lebih mempunyai *conclusion* yang sama, tetapi salah satu dari *rule* tersebut mempunyai *premise* yang tidak perlu dikondisikan dalam *rule* karena tidak mempunyai pengaruh apapun.

Contoh:

Rule 1 : If mesin tidak dapat di-*start* And motor starter tidak bekerja
And klakson tidak berbunyi
Then baterai perlu di setrum
Rule 2 : If mesin tidak dapat di-*start* And motor starter tidak bekerja
And klakson berbunyi
Then baterai perlu di setrum

STIKOMPS SURABAYA

6. *Dead-end rules*

Dead-end rules adalah suatu *rule* yang konklusinya tidak diperlukan oleh rule-rule lainnya.

Contoh:

Rule 1 : If mesin tidak dapat di-*start* And motor starter tidak bekerja
Then nonton televisi

2.13 *Decision Table*

Menurut Dologite (1993:23) *decision table* untuk menunjukkan hubungan suatu nilai-nilai yang menghasilkan beberapa tahap menengah atau rekomendasi akhir dari KBS. Sebagai contoh dari pembuatan *decision table* dapat dilihat pada Tabel 2.1. Tabel 2.1 merupakan *decision table* untuk parameter kebocoran bahan bakar.

Tabel 2.1 *Decision Table*

Rule	Saluran bahan bakar tekanan tinggi bocor	Kerusakan alat pengamat kebocoran bahan bakar	Kebocoran bahan bakar
A1	Y	Y	-
A2	Y	T	Saluran bahan bakar tekanan tinggi bocor
A3	T	Y	Kerusakan alat pengamat kebocoran bahan bakar
A4	T	T	-

2.14 *Reduced Decision Table*

Setelah didapatkan nilai dari *decision table* akan direduksi untuk mendapatkan nilai dari kondisi terakhir. *Reduced decision table* adalah pembuatan tabel yang nilai-nilainya didapat dari mereduksi *decision table*. Sebagai contoh dari pembuatan *reduced decision table* dapat dilihat pada Tabel 2.2. Tabel 2.2 merupakan *reduced decision table* untuk parameter kebocoran bahan bakar.

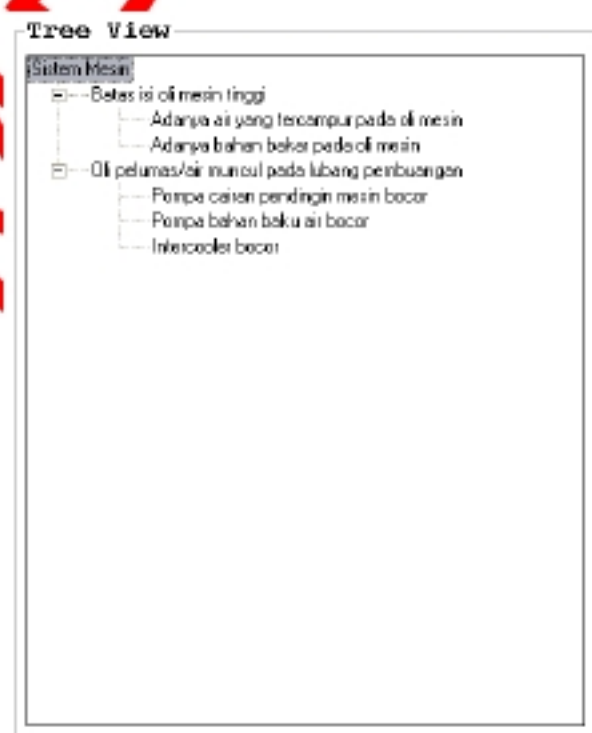
STIKOMP SURABAYA

Tabel 2.2 *Reduced Decision Table*

Rule	Saluran bahan bakar tekanan tinggi bocor	Kerusakan alat pengamat kebocoran bahan bakar	Kebocoran bahan bakar
B1	Y	T	Saluran bahan bakar tekanan tinggi bocor
B2	T	Y	Kerusakan alat pengamat kebocoran bahan bakar

2.15 Penyajian Aturan (*Rule*) dengan *Tree View*

Tree View adalah sebuah fasilitas yang disediakan *Visual Basic* 6.0 untuk penyusunan aturan-aturan. Dalam sebuah *tree view* ada beberapa fungsi dan prosedur yang bisa membantu menyusun aturan-aturan dan memanfaatkannya sebagai mekanisme inferensi ketika sistem dijalankan. Dengan *tree view* langkah-langkah untuk mengubah diagram ketergantungan (*dependency diagram*) menjadi *rule* tidak diperlukan karena diagram ketergantungan dapat langsung diaplikasikan dalam *tree view*. *Tree view* menyediakan fasilitas untuk menambah, menyisipkan, atau menghapus *node* yang ada. Contoh tampilan *tree view* dapat dilihat pada Gambar 2.9.



Gambar 2.9 Contoh *Tree View*

STIKOM SURABAYA

2.16 Definisi Kapal Minehunter

Merupakan jenis kapal perang, yang memiliki dua buah mesin utama yang terletak di dalam bagian belakang kapal. Data kapal terlihat pada Tabel 2.3.

Tabel 2.3. Data Teknik Kapal Minehunter

Dimensi	Jumlah
Panjang	51.6 m
Lebar	8.99 m

2.17 Mesin Disel 12 V

Berfungsi sebagai tenaga pendorong. Jenis mesin disel 12 V 396-TC-82-DB-51-L. Secara umum mesin terdiri dari komponen-komponen sebagai berikut:

a. Piston atau torak: Bagian yang bergerak naik atau turun selama proses pembakaran.

b. Batang piston: Bagian penyalur atau penghubung piston dengan kruk as.

c. Kruk as (*crankshaft*): Bagian pengubah tenaga ulang-alik (translasi) menjadi tenaga putaran (rotasi).

d. Kopling: Bagian pemindah daya dari kruk as ke gigi transmisi.

e. Gigi transmisi: Pengatur atau pemindah tenaga dari mesin ke penggerak akhir yaitu baling-baling.

f. Pompa injeksi: Penyuplai bahan bakar ke ruang bakar.

g. Noken as atau *camshaft* dan klep: Pengatur buka-tutup klep supply udara masuk ke ruang bakar dan penyaluran gas buang.

h. Knalpot: Penyalur pembuangan sisa pembakaran.

STIKOMPS SURABAYA

2.18 Cara Kerja Mesin Disel

Menurut Mans (1993:Bab 2-1) cara kerja mesin jenis ini memiliki empat langkah atau tahapan kerja sesuai urut-urutannya, yaitu:

a. Langkah hisap

Piston bergerak turun dari Titik Mati Atas (TMA) menuju ke Titik Mati Bawah (TMB) dan membuat ruang vakum antara kepala piston dan kepala silinder. Klep hisap membuka sehingga udara terhisap masuk mengisi ruang vakum, bersamaan dengan ini bahan bakar diinjeksikan ke ruang bakar berbentuk kabut .

b. Langkah kompresi

Pada akhir langkah hisap, klep hisap mulai menutup, campuran bahan bakar dan udara mulai dipadatkan oleh gerakan naik piston dari TMB menuju TMA.

c. Langkah tenaga atau usaha

Karena adanya perbandingan kompresi yang tinggi, maka campuran bahan bakar dan udara tersebut meledak dengan sendirinya. Energi yang timbul akibat ledakan tersebut mendorong piston menuju TMB. Gerakan piston dari TMA menuju TMB itu disebut langkah usaha.

Menjelang piston mencapai TMB, klep pembuangan mulai terbuka dan gas sisa pembakaran pun mengalir keluar.

d. Langkah buang

Proses pembuangan gas sisa pembakaran terjadi saat piston berada di TMB dan mulai bergerak menuju TMA. Pada saat klep pembuangan terbuka dan gas sisa

STIKOMP SURABAYA

pembakaran di dorong keluar dari ruang bakar. Pada saat mendekati TMA baker, aliran ini mendorong gas sisa yang masih tertinggal di ruang bakar. Setelah piston mencapai TMA, maka proses langkah hisap dimulai lagi.

STIKOMP SURABAYA