BAB II
TINJAUAN PUSTAKA

2.1 Teori Dasar

Secara umum teori merupakan suatu konseptualisasi yang umum. Konseptualisasi atau sistem pengertian diperoleh melalui jalan sistematis. Suatu teori harys dapat diuji kebenarannya. Bila tidak itu bukan teori. Teori mempuyai dasar empiris. Teori dapat memandang gejala yang dihadapi dari sudut pandang yang berbeda (Sudaryono, 2015: 13).

Pada bab ini akan dijelaskan tentang teori dasar yang meliputi kecederdasan buatan atau *Artificial Intelligence* (AI) dan bagian dari kecerdasan buatan seperti jaringan saraf tiruan (*artificial neural network*), logika *fuzzy* (*fuzzy logic*), dan sistem pakar (*expert system*); *web,*validitas data, dan basis data.

2.1.1 Kecerdasan Buatan

Kecerdasan buatan berasal dari kata bahasa inggris "*Artificial Intellegence*" atau singkatan dari AI, yaitu *Intelligence* adalah kata sifat yang berarti cerdas, sedangkan *Artificial* artinya buatan. Kecerdasan buatan yang dimaksud disini merujuk pada mesin yang akan diambil, dan mampu mengambil keputusan seperti yang dilakukan oleh manusia (Sutojo, dkk., 2011: 1).

Cerdas adalah memiliki pengetahuan, pengalaman, dan penalaran untuk membuat keputusan dan mengambil tindakan. Jadi, agar mesin bisa cerdas (bertindak seperti manusia) maka harus diberi bekal pengetahuan dan diberi kemampuan untuk menalar. (Sutojo, dkk., 2011: 3).

Kecerdasan buatan memiliki cakupan yang besar. Mulai ilmu paling umum hingga paling khusus. Dari *learnig* atau *perception* hingga pada permainan catur, pembuktian teori matematika, menulis puisi, mengemudikan mobil, dan melakukan diagnosis penyakit. Kecerdasan buatan relevan dengan macam *task* kecerdasan, kecerdasan buatan merupakan sebuah ilmu yang universal (Budiharto, dkk., 2014:132).

Implentasinya kecerdasan buatan merupakan salah satu bidang ilmu komputer yang mendayagunakan komputer sehingga dapat berprilaku cerdas ssperti manusia. Ilmu komputer tersebut mengembangkan perangkat lunak dan perangkat keras untuk menirukan tindakan manusia. Aktivitas manusia yang ditirukan seperti penaralan, penglihatan, pembelajaran, pemecahan masalah, pemahaman bahasa alami dan sebagainya. Sesuai dengan definisi tersebut, maka teknologi kecerdasan buatan dipelajari dalam bidang-bidang seperti : Robotika (*Robotics*), Penglihatan Komputer (*Computer Vision*), Pengolahan Bahasa Alami (*Natural Language Processing*), Pengenalan Pola (*Pattren Recognition*), Sistem Syaraf Buatan (*Artificial Neural System*), Pengenalan Suara (*Speech Recognition*), dan Sistem Pakar (*Expert System*). (Hartati dan Iswanti, 2008: 1)

2.1.1.1 Jaringan saraf tiruan (artificial neural network)

Jaringan saraf tiruan merupakan salah satu upaya manusia memodelkan cara kerja atau fungsi sistem syaraf manusia dalam melaksanakan tugas tertentu.pemodelan ini didasarkan oleh kemampuan otak manusia dalam mengorganisasikan sel-sel penyusunnya yang disebut *neuron*, sehingga mampu melaksanakan tugas-tugas tertentu, khususnya pengenalan pola yang sangat tinggi. Alexander dan Morton prosesor mendefinisikan bahwa jaringan saraf tiruan merupakan prosesor terbesar paralel (*paralel distributed processor*) yang sangat besar yang memiliki kecenderungan untuk menyimpan pengetahuan yang bersifat pengalaman dan membuatnya siap untuk digunakan. Jaringan saraf tiruan merupakan otak manusia dalam dua hal, yaitu : Pengetahuan diperoleh jaringan melalui proses belajar. Kekuatan hubungan antara sel syaraf (*neuron*) yang dikenal sebagai bobot-bobot sinaptik digunakan untuk menyimpan pengetahuan. Cara kerja jaringan saraf tiruan sama dengan cara kerja manusia, yaitu belajar melalui contoh. Jaringan saraf tiruan bisa digunakan untuk menyelesaikan berbagai masalah, melalui dari klasifikasi, optimasi, kompresi, peramalan (*forecasting*), sistem kontrol, sistem klarifikasi, kecurangan (*instruction detection systems*), dan sebagainnya. Pada umumnya, jaringan saraf tiruan sangat sesuai dengan permasalahan yang bernilai kontinyu, seperti pengenalan tulisan tangan, pengenalan wajah, pengenalan suara, dan sebagainya (Suyanto, 2014: 169-189).

Jaringan saraf tiruan memiliki kelebihan-kelebihan (Sutojo, dkk, 2011: 284), antara lain :

1. Belajar Adaptive, ialah kemampuan untuk mempelajari bagaimana melakukan pekerjaan berdasarkan data yang diberikan untuk pelatihan atau pengalaman awal.
2. *Self-Organisation,* ialah kemampuan untuk membuat organisasi sendiri atau representasi dari informasi yang diterimanya selama waktu belajar.
3. *Real Time Operation,* ialah perhitungan jaringan saraf tiruan yang dapat dilakukan secara parallel sehingga perangkat keras yang dirancang dan diproduksi secara khusus dapat mengambil keuntungan dari kemampuan ini.

Kelebihan-kelebihan yang dimiliki oleh jaringan saraf, yaitu (Sutojo, dkk, 2011: 284-285):

1. Tidak efektif jika digunakan untuk melakukan operasi-operasi numerik dengan persis tinggi.
2. Tidak efisien jika digunakan untuk melakukn operasi algoritma aritmatik, operasi logika, dan simbolis.
3. Untuk beroperasi jaringan saraf tiruan butuh pelatihan sehingga bila jumlah datanya besar, waktu yang digunakan untuk proses pelatihan sangat lama.

Suatu model jarigan saraf tiruan dapat disebut baik atau ditentukan oleh hubungan antara *neuron* atau yang biasa disebut arsitektur jaringan (Sutojo, dkk., 2011: 292).

Neuron-neuron tersebut terkumpul dalam lapisan-lapisan yang disebut *layers.* Terdapat 3 lapisan penyusun jaringan saraf tiruan, antaralain:

1. Lapisan input (*Input Layer*)

   Unit-unit dalam lapisan input disebut unit-unit yang bertugas menerima pola inputan dari luar yang menggambarkan suatu permasalahan.

2. Lapisan Tersembunyi (*Hidden Layer*)

   Unit-unit dalam lapisan tersembunyi disebut unit-unit tersembunyi, yang mana nilai output-nya tidak dapat diamati secara langsung.

3. Lapisan Output (*Output Layaer*)

   Unit-unit dalam lapisan output disebut unit-unit output, yang merupakan solusi jaringan saraf tiruan terhadap suatu permasalahan.

Menurut (suyanto, 2014: 174-178) *neuron-neuron* pada jaringan saraf tiruan disusun berhubungan erat dengan lagoritma beajar yang digunakan untuk melatih jaringan. Secara umum, kita bisa membagi arsitektur jaringan menjadi empat, yaitu:

1. *Single-layer Feedforward Network*

   Suatu jaringan saraf tiruan berlapis adalah jaringan neuron yang diorganisasikan dalam bentuk lapisan-lapisan. Pada bentuk jaringan yang paling sederahana, hanya terdapat *input layer* dengan node sumber proyeksi kedalam *output layer* dari *neuron* (*computation nodes*), tetapi tidak sebaliknya. Dengan kata lain, jarigan ini adalah jenis *feeedforward* yang tepat.

2. *Multi-Layer feedforward Networks*

   Kelas ke dua dari *feedforward neural network* adalah jaringan dengan satu atau lebih lapisan tersembunyi (*hidden layer*), dengan *computation nodes* yang berhubungan disebut *hidden neurons* atau *hidden units.*

3. *Recurrent Networks*

   Recurrent neural network adalah jaringan yang mempunyai minimal satu *feedback loop.* Sebagai contoh, suatu *recurrent network* bisa terdiri dari satu lapisan *neuron* tunggal dengan masing-masing *neuron* memberikan kembali *output* nya sebagai *input* semua *neuron* lain.

4. *Lattice Structure*

   sebuah *Lattice* (kisi-kisi) terdiri dari satu dimensi, dua dimensi, atau lebih array *neuron* dengan himpunan *node* sumber yang bersesuaian yang memberikan sinya *input* ke array, dimensi *lattice* mengacu pada jumlah ruang dimensi ruang dimana *graph* berada.

Menuurut (Suyanto, 2014: 178-179) belajar adalah merupakan suatu proses dimana parameter-parameter bebas jaringan saraf tiruan diadaptasikan melalui suatu proses perangsangan berkelanjutan oleh lingkungan dimana jaringan berada. Jenis berlajar ditentukan oleh pola dimana pengubahan parameter dilakukan. Sehingga dalam proses belajar terdapat kejadian-kejadian seperti jaringan saraf tiruan dirangsang oleh lingkungan, jaringan saraf tiruan mengubah dirinya sebagai hasil rangsangan ini, dan jaringan saraaf tiruan memberikan respon dengan cara yang baru kepada lingkungan, disebabkan perubahan-perubahan yang terjadi dalam bentuk sruktu internalnya sendiri.

Menurut (Suyanto, 2014: 179-180) pelatihan jaringan saraf tiruan terbagi menjadi dua, yaitu

1. *Supervised Learning* (belajar dengan pengawasan)

   Supervised atau *active learning* yaitu suatu proses belajar membutuhkan guru. Yang dimaksud guru disini adalah suatu pemilik yang memiliki pengetahuan tentang lingkungan. Guru bisa dipersentasikan sebagai sekumpualan sampel *input-output.* Pembangunan pengetahuan dilakukan oleh guru dengan memberikan respon yang diinginkan kepada jaringan saraf tiruan. Respon yang diinginkan tersebut mempresentasikan aksi optimum yang dilakukan oleh jaringan saraf tiruan. Parameter-parameter jaringan berubah-ubah beradasarkan vector latih dan sinyal kesalahan (sinyal kesalahan adalah perbedaan antara keluaran jaringan saraf tiruan dan respon yang diinginkan). Proses perubahan ini dilakukan secara berulang-ulang, selangkah demi selangkah, dengan tujuan agar jaringan saraf tiruan bisa memiliki kemampuan yang mirip dengan gurunya. Dengan kata lain, jaringan saraf tiruan dilatih untuk dapat memetakan sekumpulan sampel *input-output* dengan akurasi yang tinggi.

2. *Unsupervised Learning* (belajar tanpa pengawasan)

   unsupervised atau *self-organized* tidak membutuhkan guru untuk memantau proses belajar. Dengan kata lain, tidak ada sekumpulan sampel *input-output* atau fungsi tertentu untuk dipelajari oleh jaringan.

   Salah satu contoh *unsupervised learning* adalah *competitive learning.* Sebagai contoh, kita bisa menggunakan jaringan saraf tiruan yang terdiri dari

dua lapisan, satu lapisan masukan dan satu lapisan kompetitif. Lapisan masukan menerima data yang disediakan. Lapisan kompetitif dari *neuron-neuron* yang saling bersaing untuk meraih "kesempatan" memberikan respon ke ciri khas berisi data masukan. Dalam bentuk paling sederhana, jaringan beroperasi berdasarkan strategi "*winner-takes-all*".

2.1.1.2 Logika fuzzy (fuzzy logic)

Logika *fuzzy* adalah metodologi sistem kontrol pemecahan masalah, yang cocok untuk diimplementasikan pada sistem, mulai dari sistem yang sederhana, sistem kecil, *embedded system,* jaringan PC, multi channel atau workstation berbasis akuisisi data, dan sistem kontrol. Metodologi ini dapat diterapkan pada perangkat keras, perangkat lunak, aatau kombinasi keduannya. Dalam logika klasik dinyatakan bahwa segala sesuatu bersifat biner, yang artinya "Ya dan Tidak", "Benar atau Salah". Namun bersarnya nilainya tergantung pada bobot keanggotaan yang dimilikinya. Logika *fuzzy* dapat dugunakan di berbagai bidang, seperti pada sistem diagnosis penyakit dalam (dalam bidang kedokteran); pemodelan sistem pemasaran, riset operasi (dalam bidang ekonomi); kendali kualitas air, prediksi adanya gempa bumi, klasifikasi dan pencocokan pola (dalam bidang teknik) (Sutojo, dkk., 2011: 211-212).

Memahami logika *fuzzy,* sebelum memahami logika *fuzzy* terlebih dahulu perhatikan konsep himpunan *fuzzy*. Himpunan *fuzzy* memiliki 2 atribut, yaitu (Sutojo, dkk., 2011: 212):

1. *Linguistik*, yaitu nama suatu kelompok memwakili suatu keadaan tertentu menggunakan bahasa alami, misalnya DINGIN, SEJUK, PANAS, mewakili variabel temperature. Contoh lain misalnya MUDA, PAROBAYA, TUA, mewakili variabel umur.
2. *Numeris*, yaitu suatu nilai yang menunjukkan ukuran dari suatu variabel, misalnya 10, 35, 40, dan sebagainya.

Disamping itu, ada beberapa hal yang harus dipahami dalam memahami logika *fuzzy* , yaitu (Sutojo, dkk., 2011: 211-212):

1. Varaibel *fuzzy*, yaitu variable yang akan dibahas dalam suatu sistem *fuzzy*. Contoh: penghasilan, temperature, permintaan, umur, dan sebagainya.
2. Himpunan *fuzzy*, yaitu suatu kelompok yang mewakili suatu keadaan tertentu dalam suatu variabel *fuzzy*.
3. Semesta pembicara, yaitu seluruh nilai yang diijinkan untuk dioperasikan dalam suatu variabek *fuzzy*.
4. Domain himpunan *fuzzy*, yaitu seluruh nilai yang diijinkan dalam semesta pembicara dan boleh dioperasikan dalam suatu himpunan *fuzzy*.

Struktur sistem inferensi *fuzzy*, terbentuk dari empat elemen anatara lain (Sutojo, dkk., 2011: 211-232):

1. Basis pengetahuan *fuzzy*, yaitu kumpulan aturan *fuzzy* dalam bentuk pernyataan IF…THEN.
2. Fuzzifikasi, yaitu proses untuk mengubah input sistem yang mempunyai nilai tegas menjadi variabel linguistik menggunakan fungsi keanggotaan yang disimpan dalam basis pengetahuan *fuzzy*.

3. Mesin inferensi, yaitu proses untuk mengubah *input fuzzy* menjadi output *fuzzy* dengan cara mengikuti aturan-aturan yang telah ditetapkan pada basis pengetahuan *fuzzy*.
4. Defuzzifikasi, yaitu mengubah *output fuzzy* yang diperoleh dari mesin inferensi menjadi nilai tegas menggunakan fungsi keanggotaan yang sesuai dengan saat dilakukan fuzzifikasi.

Terdapat beberapa metode yang digunakan dalam sistem inferensi *fuzzy*, yaitu (Sutojo, dkk., 2011: 233-237):

1. Metode Tsukamoto

Dalam inferensinya, metode Tsukamoto menggunakan tahapan berikut:

a. Fuzifikasi
b. Pembentukan basis data pengetahuan *fuzzy* (rule dalam bentuk IF…THEN)
c. Mesin inferensi menggunkan nilai *MIN* (minimum)
d. Defuzzyfikasi menggunakan nilai *Average* (rata-rata)

2. Metode Mamdani

Metode Mamdani paling sering digunakan dalam aplikasi-apalikasi karena strukturnya sederhana, yaitu menggunakan operasi MIN-MAX atau MAX-PRODUCT. Untuk mendapatkan *output,* diperlukan 4 tahapan berikut:

a. Fuzzyfikasi
b. Pembentukan basis pengetahuan fuzzy (dalam bentuk IF…THEN)

c. Aplikasi fungsi implikasi menggunakan fungsi *MIN* dan komposisi antara-rule menggunakan fungsi *MAX* (menghasilkan himpunan fuzzy baru)

d. Defuzzyfikasi menggunakan meode *Centroid*

3. Metode Sugeno

Dalam metode Sugeno, *output* sistem berupa konstanta atau persamaan linier. Metode ini diperkenalkan oleh Takagi-Sugeno Kang pada 1985. Dalam inferensinya metode Sugeno menggunakan tahapan:

a. Fuzzyfikasi

b. Pembentuukan basis pengetahuan *fuzzy* (rule dalam bentuk IF…THEN)

c. Mesin inferensi menggunakan fungsi implikasi *MIN*

d. Defuzzyfikasi menggunakan *Average* (rata-rata)

2.1.1.3 Sistem pakar (*expert system*)

Sistem pakar merupakan cabang dari *Artificial Intelligence* (AI) yang cukup tua karena sistem ini mulai dikembangkan pada pertengahan 1960. Sistem pakar yang pertama kali adalah *General-purpose problem solver* (GPS) yang dikembangkan oleh newel dan simon. Sampai saat ini sudah banyak sistem pakar yang dibuat, seperti MYCIN untuk diagnosis penyakit, DENDRAL untuk mengidentifikasi struktur molekul campuran yang tak dikenal, XCON dan XSEL untuk membantu konifgurasi sistem komputer besar, SOPHIE untuk analisis sirkuit elektronik, Prospector digunakan di bidang geologi untuk membantu mencari dan menemukan deposit, FOLIO digunakan untuk membantu

memberikan keputusan bagi seorang manajer dalam masalah stok dan investasi, DELTA dipakai untuk pemeliharaan lokomotif listrik diesel, dan sebagainya (Sutojo, dkk., 2011: 159-160).

Sistem pakar merupakan sebuah program komputer yang menyimpulasi penilaian dan prilaku manusia atau organisasi yang mempunyai pengetahuan dan pengalaman ahli dalam bidang tertentu. Sistem pakar yang canggih dapat ditingkatkan dengan penambahan basis pengetahuan atau set aturan (Budiharto, dkk., 2014: 132).

Menurut beberapa pakar dalam Merlina dan Hidayat (2012: 1) definisi tentang sistem pakar, antara lain:

1. Menurut Durkin, yaitu sistem pakar adalah suatu program komputer yang dirancang untuk memodelkan kemampuan penyelesaian masalah yang dilakukan seorang pakar.
2. Menurut Ignizio, yaitu sistem pakar adalah suatu model dan prosedur yang berkaitan, dalam suatu domain tertentu, yang mana tingkat keahliannya dapat dibandingkan dengan keahlian seorang pakar.
3. Menurut Giarranto dan Riley, yaitu sistem pakar adalah suatu komputer yang bisa menyamai atau meniru kemampuan seorang pakar.

Inferensi adalah prosedur (program) yang mempunyai kemampuan dalam melakukan penalaran. Inferensi ini ditampilkan pada suatu komponen yang disebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah. Semua pengetahuan yang dimiliki oleh seorang pakar disimpan pada basis pengetahuan oleh sistem pakar. Tugas mesin inferensi adalah mengambil

kesimpulan berdasarkan basis pengetahuan yang dimiliki (Sutojo, dkk., 2011: 164-165).

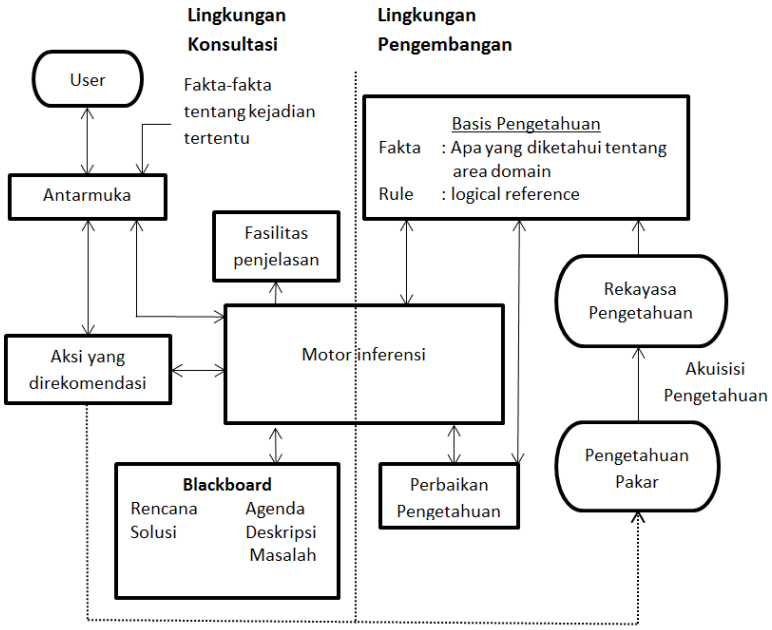
Mesin inferensi merupakan otak dari sistem pakar. Komponen ini sebenarnya adalah program komputer yang menyediakan metodologi untuk *reasoning* (pertimbangan) mengenai informasi dalam basis pengetahuan dan dalam "*workplace*"dan digunakan untuk merumuskan kesimpulan (Merlina dan Hidayat, 2014: 6).

Mesin inferensi mempunyai 3 elemen utama, yaitu:

1. *Interpreter*, adalah elemen yang mengeksekusi item agenda yang dipilih dengan mengaplikasikannya pada basis pengetahuan *rule* yang berhubungan.
2. *Scheduler*, adalah elemen yang menjaga kontrol di sepanjang agenda. Memperkirakan akibat dari pengaplikasian *rule inferensia* yang menampakkan prioritas item atau kriteria lain pada agenda.
3. *Consistency enforcer*, adalah elemen yang mencoba menjaga konsistensi representasi solusi yang muncul.

Ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan (development environment) dan lingkungan konsultasi (consultation environment). Lingkungan pengembangan digunakan oleh pembuat sistem pakar untuk membangun komponen-komponennya dan memperkenalkan pengetahuan ke dalam knowledge base (basis pengetahuan). Lingkungan konsultasi digunakan oleh pengguna untuk berkonsultasi sehingga pengguna mendapatkan pengetahuan dan nasihat dari sistem pakar layaknya berkonsultasi dengan seorang pakar

(Sutojo, dkk., 2011: 166). Gambar 2.1 menunjukkan komponen-komponen yang penting dalam sebuah sistem pakar.



Gambar 2.1 Komponen-komponen sistem pakar
Sumber: (Sutojo, dkk., 2011: 166)

Keterangan:

1. Akuisisi Pengetahuan

   Subsistem ini digunakan untuk memasukkan pengetahuan dari seorang pakar dengan cara merekayasa pengetahuan agar bisa di proses oleh computer dan menaruhnya ke dalam basis pengetahuan dengan format tertentu (dalam bentuk representasi pengetahuan). Sumber-sumber pengetahuan bisa diperoleh dari pakar, buku, dokumen multimedia, basis data, laporan riset khusus, dan informasi yang terdapat di Web.

2. Basis Pengetahuan (*Knowledge Base*)

Basis pengetahuan mengandung pengetahuan yang diperlukan untuk memahami, memformulasikan, dan menyelesaikan masalah. Basis pengetahuan terdiri dari dua elemen dasar, yaitu:

a. Fakta, misalnya situasi, kondisi atau permasalahan yang ada.
b. Rule (Aturan), untuk mengarahkan penggunaan pengetahuan dalam memecahkan masalah.

3. Mesin Inferensi (*Inference Engine*)

Mesin inferensi adalah sebuah program yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada, memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan.

4. Daerah Kerja (*Blackboard*)

Untuk merekam hasil sementara yang akan dijadikan sebagai keputusan dan untuk menjelaskan sebuah masalah yang sedang terjadi, sistem pakar membutuhkan Blackboard, yaitu area pada memori yang berfungsi sebagai basis data. Tiga tipe keputusan yang dapat direkam pada blackboard, yaitu:

a. Rencana: bagaimana menghadapi masalah.
b. Agenda: aksi-aksi potensial yang sedang menunggu untuk dieksekusi.
c. Solusi: calon aksi yang akan dibangkitkan.

5. Antarmuka Pengguna (*User Interface*)

Digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi ini paling bagus bila disajikan dalam bahasa alami (natural

language) dan dilengkapi dengan grafik, menu, dan formulir elektronik. Pada bagian ini akan terjadi dialog antara sistem pakar dan pengguna.

6. Subsistem Penjelasan (*Explanation Subsystem / Justifier*)

   Berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil. Kemampuan seperti ini sangat penting bagi pengguna untuk mengetahui proses pemindahan keahlian pakar maupun dalam pemecahan masalah.

7. Sistem Perbaikan Pengetahuan (*knowledge refining system*)

   Berfungsi memperbaiki pengetahuan (knowledge refining system) dari seorang pakar diperlukan untuk menganalisis pengetahuan, belajar dari kesalahan masa lalu, kemudian memperbaiki pengetahuannya sehingga dapat dipakai pada masa mendatang. Kemampuan evaluasi diri seperti itu diperlukan oleh program agar dapat menganalisa alasan-alasan kesuksesan dan kegagalannya dalam mengambil kesimpulan. Dengan cara ini basis pengetahuan yang lebih baik dan penalaran yang lebih efektif akan dihasilkan.

8. Pengguna (*User*)

   Pada umumnya pengguna sistem pakar bukanlah seorang pakar (non-expert) yang membutuhkan solusi, saran, atau pelatihan (training) dari berbagai permasalahan yang ada.

Representasi pengetahuan merupakan merupakan metode yang digunakan untuk mengkodekan pengetahuan dalam sebuah sistem pakar. Representasi

dimaksudkan untuk menangkap sifat-sifat penting masalah dan membuat iinformasi itu dapat diakses oleh prosedur pemecahan masalah (kusrini, 2008: 6).

Adapun karakteristik dari metode representasi pengetahuan adalah

1. Harus bisa diprogram bahasa pemograman atau dengan *shells* dan hasilnya disimpan dalam memori
2. Dirancang sedemikian sehingga isinya dapat digunakan untuk proses penalaran.
3. Model representasi pengatahuan merupakan sebuah stuktur data yang dapat dimanipulasi oleh mesin inferensi dan pencarian untuk aktivasi pencocokan pola.

Dalam melakukan suatu inferensi, sistem pakar memerlukan adanya proses pengujian kaidah-kaidah dalam urutan tertentu untuk mencari suatu kondisi yang sesuai dengan kondisi awal atau untuk memastikan kondisi yang berjalan sudah dimasukkan ke dalam database. Proses tersebut diistilahkan dengan perunutan atau penalaran, yaitu proses pencocokan fakta atau kondisi tertentu yang tersimpan dalam basis pengetahuan maupun pada memori kerja dengan kondisi yang dinyatakan dalam premis atau bagian kondisi pada suatu kaidah atau aturan (Hartati dan Iswanti, 2008: 45).

Ada beberapa konsep penalaran yang dapat digunakan oleh mesin inferensi yaitu:

a. Runut maju (*forward chaining*).

Konsep ini merupakan proses perunutan diawali dengan menampilakan kumpulan data atau fakta yang pasti menuju konklusi akhir. Runut maju disebut

juga sebagai pencarian yang dimotori data (*data drive search*). Proses perunutan (penalaran) dimulai dari premis-premis atau informasi masukan (IF) terlebih dahulu kemudian menuju konklusi atau derived information (THEN). Konsep ini dapat dimodelkan sebagai berikut:

IF (informasi masukan)

THEN (konklusi)

Informasi masukan dapat berupa suatu bakti dari hasil pengamatan sedangkan konklusi dapat berupa diagnosa sehingga dapat dikatakan jalannya penalaran runut maju dimulai dari pengamatan menuju diagnosa. Pada metode ini, sistem tidak melakukan praduga terhadap konklusi, sistem hanya akan menerima semua gejala yang dimasukan pengguna pada sistem dan kemudian sistem akan melakukan pemeriksaan gejala-gejala tersebut dan selanjutnya mencocokkan dengan konklusi yang sesuai (Hartati dan Iswanti, 2008: 45-47).

b. Runut mundur (*backward chaining*)

Konsep ini merupakan proses perunutan yang meruapakan kebalikan dari runut maju. Jadi umumnya runut balik diaplikasikan ketika tujuan ditentukan sebagai kondisi atau keadaan awal. Konsep ini disebut juga sebagai *goal-driven reasoning*, merupakan cara yang efesien untuk memcahkan masalah yang dimodelkan sebagai masalah pemilihan terstruktur.

Konsep ini dapat dimodelkan sebagai berikut:

Tujuan,

IF (kondisi). Penalaran dimulai dengan tujuan kemudian merunut balik ke jalur yang akan mengarahkan ketujuan tersebut. Pada konsep ini proses internal selalu

memeriksa konklusi (tujuan) terlebih dahulu sebagai praduga awal, kemudian memastikan gejala-gejala (kondisi) telah terpenuhi dan sistem mengeluarkan konklusi sebagai output. Apabila sistem menemukan ada bagian gejala-gejala (kondisi) yang tidak terpenuhi maka sistem akan memeriksa konklusi (tujuan) pada aturan selanjutnya (Hartati dan Iswanti, 46-47).

Untuk mendapatkan hasil yang akurat dalam pemecahan masalah pada suatu domain. Biasanya dibutuhkan aturan cukup banyak karena masing-masing aturan berisi detail pengetahuan. Jumlah aturan akan menggambarkan kompleksitas sistem pakar (Kusrini, 2008: 7-8).

Inferensi merupakan proses untuk menghasilkan informasi dari fakta yang diketahui atau diasumsikan. Inferensi adalah konklusi logis atau implikasi berdasarkan informasi yang tersedia (Kusrini, 2008: 8-12).

kaidah menyediakan cara formal untuk merepresentasikan yang dituliskan dalam bentuk jika-maka (*IF-THEN*). Kaidah *IF-THEN* menghubungkan antesenden (*antecedent*) dengan konsekuensi yang diakibatkannya. Berikut ini adalah contoh struktur kaidah *IF-THEN* yang menghubungkan obyek (Adedeji, 1992 dalam Hartati dan Iswanti, 2008: 25):

1. *IF* premis *THEN* konklusi
2. *IF* masukan *THEN* keluaran
3. *IF* kondisi *THEN* tindakan
4. *IF* antesenden *THEN* konsekuen
5. *IF* data *THEN* hasil
6. *IF* tindakan *THEN* tujuan

7. *IF* aksi *THEN* reaksi
8. *IF* gejala *THEN* diagnose

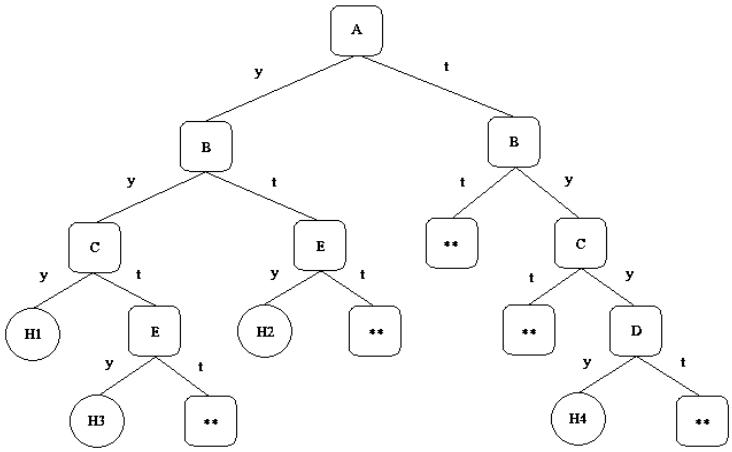
Premis mengacu pada fakta yang harus benar sebelum konklusi tertentu dapat diperoleh. Masukan mengacu pada data yang harus tersedia sebelum keluaran dapat diperoleh. Kondisi mengacu pada keadaan yang harus berlaku sebelum tindakan dapat diambil. Antesenden mengacu pada situasi yang terjadi sebelum konsekuensi dapat diamati. Data mengacu pada informasi yang harus tersedia sehingg suatu hasil akan dapat diperoleh. Tindakan mengacu pada kegiatan yang harus dilakukan sebelum hasil dapat diharapkan. Aksi mengacu pada kegiatan yang menimbulkan efek dari tindakan tersebut. Gejala mengacu pada keadaan yang menyebabkan adanya kerusakan atau keadaan tertentu yang mendorong adanya pemeriksaan (diagnosa) (Hartati dan Iswanti, 2008: 25-26).

Untuk mencapai kaidah produksi, terlebih dahulu melakukan pencarian pengetahuan yang kemuadian akan disajikan dalam bentuk pohon keputusan. Berikut ini merupakan contoh penyajian dalam bentuk tabel keputusan dan pohon keputusan (Hartati dan Iswanti, 2008: 26-39).

Tabel 2.1 Tabel Keputusan

Hipotesa / ***Evidence***	**Hipotesa 1**	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Evidence A	Ya	ya	Ya	tidak
Evidence B	Ya	tidak	Ya	ya
Evidence C	Ya	tidak	Tidak	ya
Evidence D	Tidak	tidak	Tidak	ya
Evidence E	Tidak	Ya	Ya	tidak

Sumber: Hartati dan Iswanti (2008: 32)



Keterangan:

A = *evidence* A,	H1 = hipotesa 1,	y	= ya
B = *evidence* B,	H2 = hipotesa 2,	t	= tidak
C = *evidence* C,	H3 = hipotesa 3,	**	= tidak menghasilkan hipotesa tertentu
D = *evidence* D,	H4 = hipotesa 4		

Gambar 2.2 Pohon Keputusan
(Sumber: Hartati dan Iswanti, 2008: 33)

Dari gambar 2.2 dapat diketahui bahwa hipotesa H1 terpenuhi jika memenuhi *evidence* A, B, dan C. Hipotesa H2 terpenuhi jika memiliki *evidence* A dan *evidence* E. Hipotesa H3 akan terpenuhi jika memiliki *evidence* A, B, dan E. Hipotesa H4 akan dihasilkan jika memenuhi *evidence* B, C, dan D. Notasi "y" mengandung arti memenuhi *node* (*evidence*) di atasnya, notasi "t" artinya tidak memenuhi.

Dalam sesi konsultasi pada sistem pakar, *node-node* yang mewakili *evidence* biasanya akan menjadi pertanyaan yang diajukan oleh sistem. Dengan melihat pohon keputusan pada gambar 2.2 permasalahan dapat saja terjadi pada awal sesi konsultasi yaitu pada saat sistem pakar menanyakan "apakah memiliki *evidence* A?". Permasalahannya merupakan apapun jawaban pengguna baik "ya"

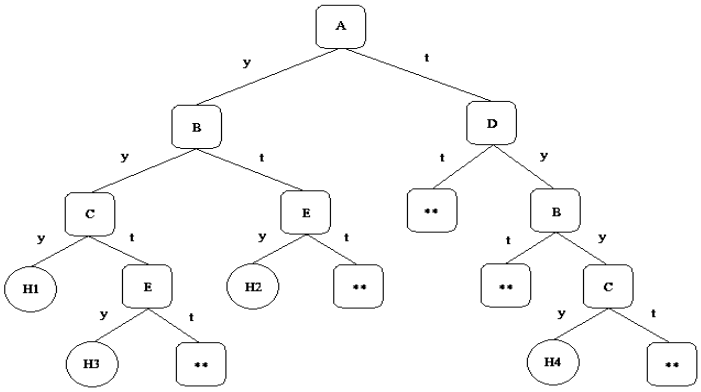
atau “tidak” maka sistem akan menanyakan *evidence* B. maka jawaban pengguna tidak akan mempengaruhi sistem. Salah satu cara alternative dalam mengatasi hal ini adalah dengan mengubah urutan pada tabel keputusan seperti terlihat pada tabel 2.2.

Tabel 2. 2 Alternatif Tabel Keputusan

Hipotesa / ***Evidence***	**Hipotesa 1**	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Evidence A	Ya	ya	Ya	Tidak
Evidence D	Tidak	tidak	Tidak	Ya
Evidence B	Ya	tidak	Ya	Ya
Evidence C	Ya	tidak	Tidak	Ya
Evidence E	Tidak	ya	Ya	Tidak

Sumber: Hartati dan Iswanti, (2008: 34)

Berdasarkan tabel 2.2 dapat dihasilkan pohon keputusan sebagai berikut:



Keterangan:

A = *evidence* A, H1 = hipotesa 1, y = ya

B = *evidence* B, H2 = hipotesa 2, t = tidak

C = *evidence* C, H3 = hipotesa 3, ** = tidak menghasilkan hipotesa tertentu

D = *evidence* D, H4 = hipotesa 4

Gambar 2.3 Alternatif Pohon Keputusan
(Sumber: Hartati dan Iswanti, 2008: 35)

Dilihat dari pohon keputusan gambar 2.3, masing-masing node yang mewakili *evidence* tertentu untuk kondisi "y" dan "t" sudah tidak mengarah pada *evidence* yang sama. Hal ini berarti jawaban pengguna yang berbeda akan mengarah pada pertanyaan yang berbeda pula.

Kaidah yang dapat dihasilkan berdasarkan pohon keputusan pada gambar 2.3 adalah sebagai berikut:

1. Kaidah 1: *IF* A *AND* B *AND* C *THEN* H1
2. Kaidah 2: *IF* A *AND* B *AND* E *THEN* H3
3. Kaidah 3: *IF* A *AND* E *THEN* H2
4. Kaidah 4: *IF* D *AND* B *AND* C *THEN* H4

Dalam implementsinya penggunaan representasi pengetahuan kaidah produksi banyak diterapkan pada sistem pakar karena representasi ini mudah untuk dipahami dan bersifat deklaratif sesuai dengan jalan pikiran manusia dalam menyelesaikan suatu masalah, dan mudah diinterpretasikan.

Menurut Sutujo, dkk (2011: 160-161) Sistem pakar memiliki kemampuan dan mafaat yang diberikannya, di antaranya (Sutojo, dkk., 2011: 160-161):

1. Meningkatkan produktifitas, karena sistem pakar dapat bekerja lebih cepat dari pada manusia.
2. Membuat seorang yang awam bekerja seperti layaknya seorang pakar.
3. Menigkatkan kualitas
4. Mampu mengkap pengetahuan dan kepakaran seseorang.
5. Dapat beroperasi di lingkungan yang berbahaya.
6. Memudahkan akses pengetahuan seorang pakar.

7. Handal. Sistem pakar tidak pernah menjadi bosan dan kelelahan atau sakit.
8. Meningkatkan kapabilitas sistem komputer.
9. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti.
10. Bisa digunakan sebagai media pelengkap dalam penelitian
11. Menigkatkan kemampuan untuk menyelesaikan masalah karena sistem pakar mengambil sumber pengetahhuan dari banyak pakar.

Selain manfaat yang diberikan , ada pun kekurangan yang dimiliki sistem pakar, di antaranya:

1. Biaya yang sangat mahal untuk membuat dan memeliharanya
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar
3. Sistem pakar tidak 100% benar

Sistem pakar adalah suatu program yang berbasiskan pengetahuan yang memiliki ciri-ciri sebagai berikut (Sutujo, dkk. 2011: 162):

1. Terbatas pada domain keahlian tertentu.
2. Dapat memberikan penalaran untuk data-data yang tidak lengkap atau tidak pasti.
3. Dapat menjelaskan alasan-alasan dengan cara yang dapat dipahami.
4. Bekerja berdasarkan kaidah/*rule* tertentu.
5. Mudah dimodifikasi.
6. Baisis pengetahuan dan mekanisme inferensi terpisah.
7. Keluarannya bersifat anjuran.
8. Sistem dapat mengaktifkan kaidah secara searah yang sesuai, dituntun dialog dengan pengguna.

2.1.2 Validasi Sistem

Validasi mengacu pada sekumpulan aktifitas yang berbeda yang menjamin bahwa sistem atau perangkat lunak yang dibangun telah sesuai dengan yang diharapkan. Beberapa pendekatan dalam melakukan pengujian untuk validasi sistem antara lain (A.S. dan Shalahuddin, 2013: 275-276):

1. *Black-Box Testing* (pengujian kotak hitam)

Pendekatan ini dilakukan dengan menguji sistem atau perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Tujuannya untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari sistem atau perangkat lunak sesuai dengan spesifikasi yang dibutuhkan.

Pengujian dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan menggunakan sistem atau perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan *black-box testing* harus dibuat dengan kasus benar dan kasus salah.

2. *White-Box Testing* (pengujian kotak putih)

Pendekatan ini dilakukan dengan menguji sistem atau perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi yang dibutuhkan. *White-box testing* dilakukan dengan memeriksa logika dari kode program. Pembuatan kasus uji dapat mengikuti standar pengujian dari standar pemrograman yang ada.

2.1.3 *Database* (basis data)

Menurut A.S. dan Shalahudin (2013: 43-44) sistem *database* adalah sistem terkomputerisasi yang tujuan utamanya adalan memelihara data atau informasi yang sudah diolah dan membuat informasi tersedia saat dibutuhkan. *Database* adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Kebutuhan basis data meliputi memasukkan, menyimpan, dan mengambil data serta membuat laporan berdasarkan data yang telah disimpan. Salah satu bentuk basis data yang dibutuhkan dalam sebuah sistem yaitu *Database Management System* (*DBMS*). *DBMS* adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data. Syarat minimal dari *DBMS* antara lain (A.S. dan Shalahuddin, 2013: 44-45):

1. Menyediakan fasilitas untuk mengelola akses data
2. Mampu menangani integritas data
3. Mampu menangani akses data yang dilakukan secara bersamaan
4. Mampu menangani *backup* data

DBMS memiliki beberaa versi diantaranya antara lain:

1. *DBMS* versi komersial, yaitu *Oracle, Microsoft SQL Server, IBM DB2*, dan *Microsoft Access*
2. *DBMS* versi *open source*, yaitu *MySQL, PostgreSQL, Firebird*, dan *SQLite*

2.1.4 Web

Menurut raharjo (2011: 2) *Web* adalah merupakan kependekan dari *WWW* (*World Wide Web*), yaitu merupakan layanan di dalam ajringan internet yang berupa ruang informasi. Dengan adanya *web, user* dapat memperoleh atau menemuukan informasi yang diinginkan dengan cara mengikuti *link* (*hyperlink*) yang disediakan dalam dokumen yang ditampilakn oleh *web browser*. Kini *web* telah menjadi antarmuka (*interface*) standar untuk internet seperti halnya aplikasi *email*.

2.2 Variabel Penelitian

Variabel penelitian adalah suatu atribut atau sifat atau nilai dari orang, obyek atau kegiatan yang mempunyai variasi tertentu yang ditetapkan oleh peneliti untuk dipelajari dan ditarik kesimpulannya. Variabel dalam penelitian ini adalah *Air Compressor* dan variabel penelitian yang ditetapkan yaitu kerusakan *Air Compressor* (Sugiyono, 2014: 38)

2.2.1 Kompresor Udara (*Air Compressor*)

Kompresor adalah alat pemampat atau pengkompresi udara dengan kata lain kompresor adalah penghasil udara mampat. Karena proses pemampatan, udara mempunyai tekanan yang lebih tinggi dibandingkan dengan tekanan udara lingkungan (1atm). Dalam keseharian, kita sering

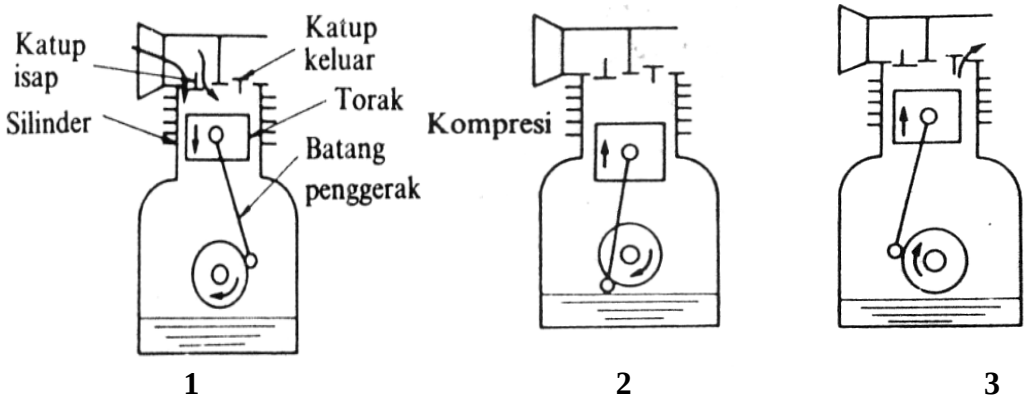
memanfaatkan udara mampat baik secara langsung atau tidak langsung. Sebagai contoh, udara manpat yang digunakan untuk mengisi ban mobil atau sepeda montor, udara mampat untuk membersihkan bagian-bagian mesin yang kotor di bengkel-bengkel dan manfaat lain yang sering dijumpai sehari-hari. Pada industri, penggunaan kompresor sangat penting, baik sebagai penghasil udara mampat atau sebagai satu kesatuan dari mesin-mesin. Kompresor banyak dipakai untuk mesin pneumatik, sedangkan yang menjadi satu dengan mesin yaitu turbin gas, mesin pendingin dan lainnya. (Sunyoto dkk, 2008: 180).

Gambar 2.4 Kompresor

(Sumber: http://www.sperre.com/compressors/classic/hl2-120#)

Proses kerja dari kompresor kerja tunggal dan ganda. Adapun urutan proses lengkapnya, tahap pertama adalah langkah hisap, torak bergerak ke bawah oleh tarikan engkol. Di dalam ruang silinder tekanan menjadi negatif di bawah 1 atm, katup hisap terbuka karena perbedaan tekanan dan udara terhisap.

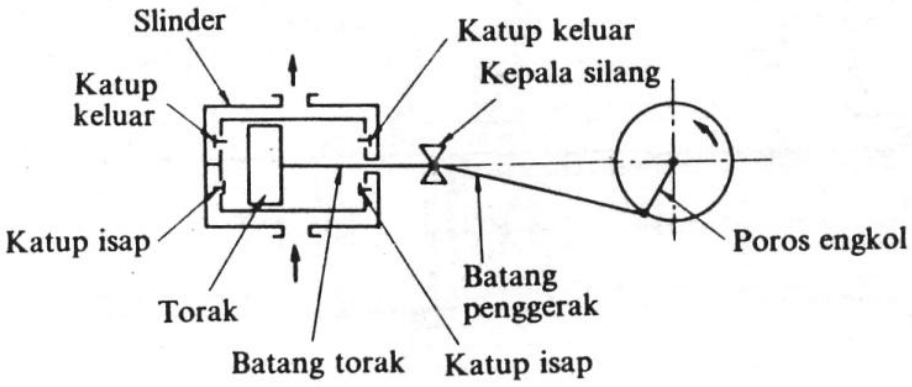
Kemudian torak bergerak keatas, katup hisap tertutup dan udara dimampatkan. Karena tekanan udara mampat, katup ke luar menjadi terbuka.



Gambar 2.5 Proses kerja dari kompresor torak tunggal
(Sumber: Sunyoto dkk, 2008: 182)

Keterangan:

1. Hisap: udara masuk kompresor karena tekanan di dalam silinder lebih rendah dari 1 atm.
2. Kompresi: udara di dalam kompresor dikompresi, tekanan dan temperatur udara naik.
3. Pengeluaran: karena tekanan udara mampat, katup ke luar terbuka dan udara mampat ke luar silinder.



Gambar 2.6 Proses kerja dari kompresor torak kerja ganda
(Sumber: Sunyoto, dkk, 2008: 183)

Gambar 2.3 di atas adalah kompresor torak kerja ganda. Proses kerjanya tidak berbeda dengan kerja tunggal. Pada kerja ganda, setiap gerakan terjadi sekaligus langkah penghisapan dan pengkompresian. Dengan kerja ganda, kerja kompresor menjadi lebih efisien.

Prinsip kerja kompresor dan pompa adalah sama, kedua mesin tersebut menggunakan energi luar kemudian diubah menjadi energi fluida. Pada pompa, di nosel ke luarnya energi kecepatan diubah menjadi energi tekanan, begitu juga kompresor pada katup ke luar udara mampat mempunyai energi tekanan yang besar. Hukum-hukum yang berlaku pada pompa dapat diaplikasikan pada kompresor.

Pada dasarnya kompresor klasifikasi kompresor berdasarkan tekanannya atau cara pemampatannya, kompresor berdasarkan pemanpatannya dibagi menajadi dua, diantaranya (Sunyoto, dkk, 2008: 133):

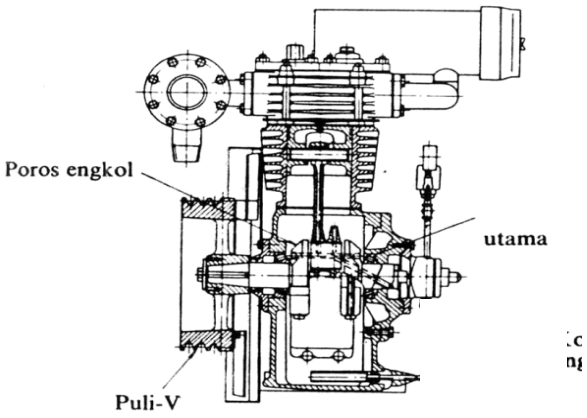
1. Jenis turbo adalah dengan menggunakan gaya sentrifugal yang diakibatkan oleh putaran impeler sehingga udara mengalami kenaikan energi yang akan diubah menjadi energi tekanan.

2. Jenis perpindahan adalah dengan memperkecil volume udara yang dihisap kedalam silinder atau stator dengan torak sudu.

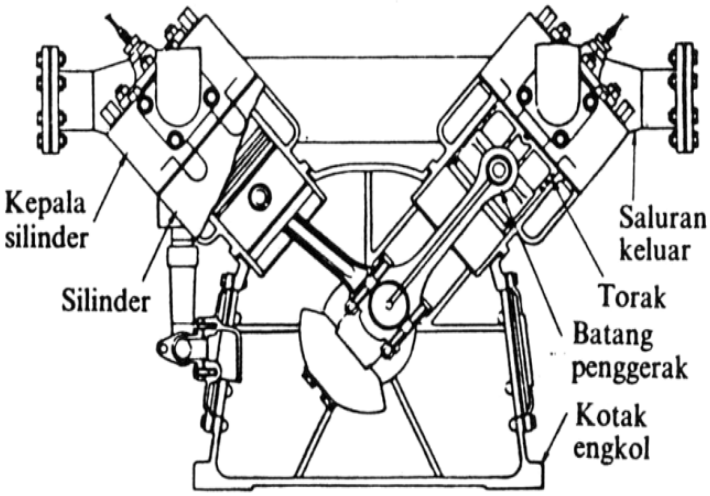
Kompresor merupakan mesin yang membutuhkan penggerak dari luar. Penggerak yang didgunakan adalah motor listrik atau motor bahan bakar. Penggunaan motor listrik mempunyai keunggulan yaitu tidak berbunyi berisik, tidak menimbulkan polusi, murah dan operasi pemeliharaannya mudah. Penggunaan motor listrik yang biasa digunakan jenis motor induksi dan sinkron (Sunyoto, dkk, 2008: 199).

Gambar 2.7 Motor sinkron
(Sumber: https://indonesian.alibaba.com/product-detail/y2-series-three-phase-mitsubishi-electric-induction-motor-1894855208.html)

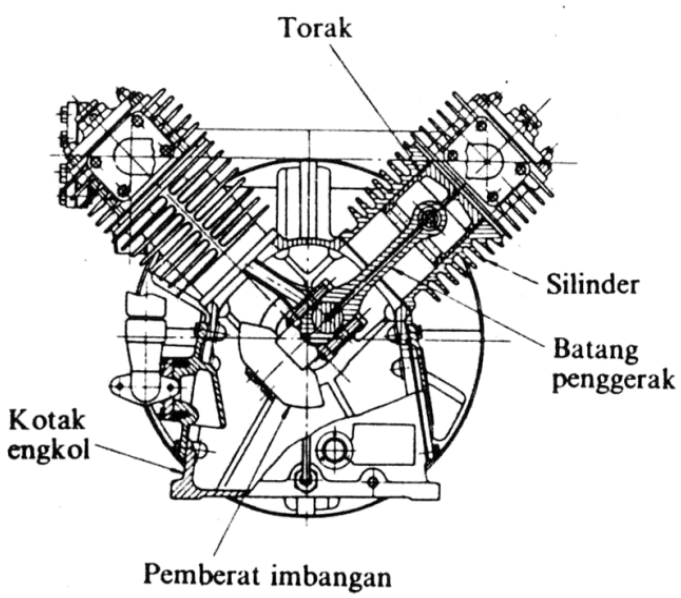
Terjadinya proses kompresi yang menaikan suhu udara mampat, maka pada siliinder torak dipasang sistem pendingin menggunakan sirip-sirip untuk pengdingin dengan udara.



Gambar 2.8 Kompresor torak dengan pendingin udara
(Sumber: Sunyoto, dkk, 2008: 203)



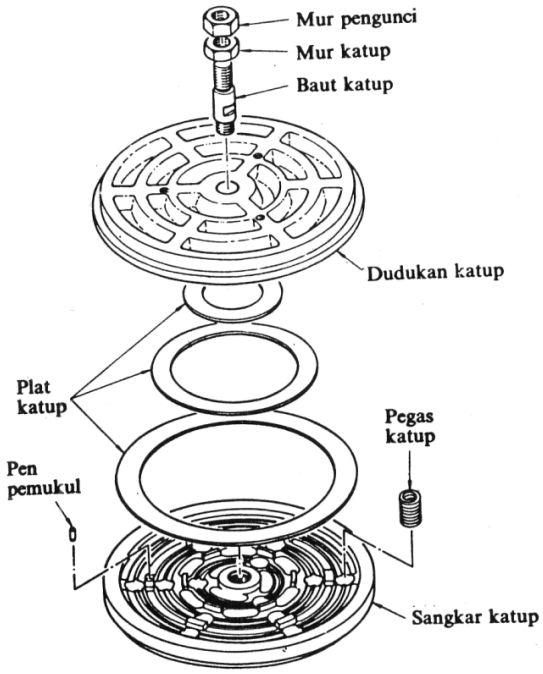
Gambar 2.9 Kontruksi kompresor torak silinder
(Sumber: Sunyoto, dkk, 2008: 203)



Gambar 2.10 Kontruksi kompresor torak silinder
(Sumber: Sunyoto, dkk, 2008: 203)

Fungsi torak sudah jelas adalah sebagai alat pemampat sehingga dengan pergerakan torak volume silinder dapat berubah-ubah. mengingat pentingnya fungsi tersebut, torak harus memiliki terbuat dari bahan yang kuat, tahan panas, dan ringan.

Pengaturan siklus udara dari dan ke dalam silinder diatur oleh mekanisme katup. Katup pada kompresor bekerja karena tekanan. Untuk katup hisap terbuka karena tekanan dalam silinder vakum sehingga dengan desakan tekanan luar katup terbuka. Sedangkan katup keluar terbuka karena tekanan silinder sudah cukup kuat untuk membuka katup ke luar. Pada saluran katup hisap dipasang penyaring udara, sehingga udara yang dihisap lebih bersih terbebas dari kotoran-kotoran yang dapat meyebabkan sumbatan pada katup atau saluran lainnya.



Gambar 2.11 Konstruksi katup kompresor jenis cincin
(Sumber: Sunyoto, dkk, 2008: 205)

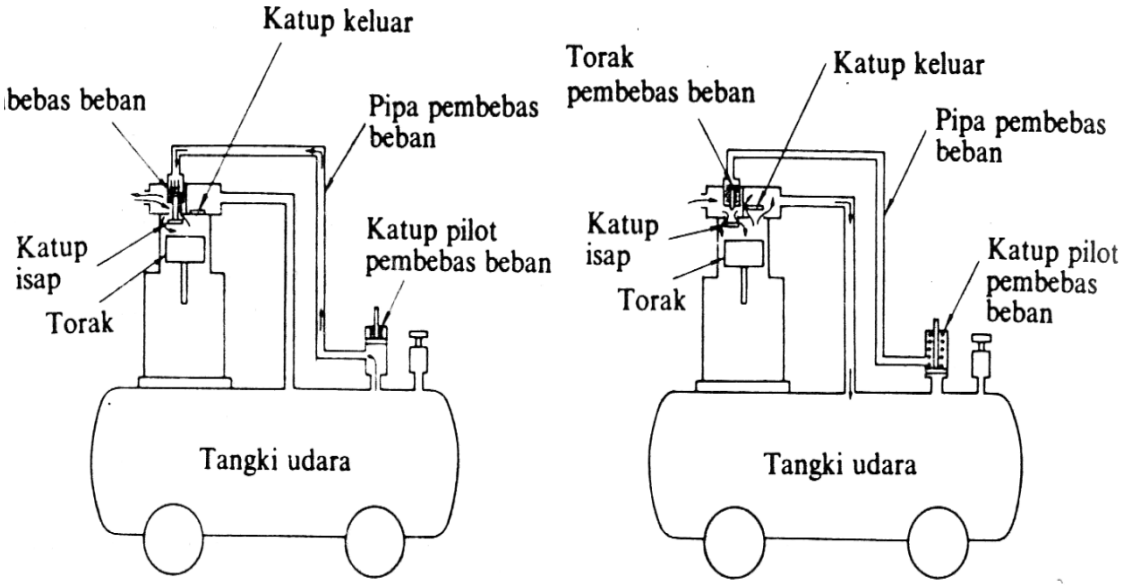
Komponen penting lainnya pada kompresor torak adalah poros engkol dan batang, Kedua komponen ini bertugas mengubah gerakan putar poros menjadi gerak bolak-balik torak. Gerakan putar diperoleh poros engkol dari motor penggerak yaitu motor bakar atau motor listrik. Poros motor penggerak dan poros engkol dapat dikopel langsung, atau dengan transmisi (roda gigi, sabuk, atau pul). Untuk memyeimbangkan gerakan dan juga memperhalus getaran pada poros engkol dipasang pemberat imbangan. Poros engkol dan peralatan tambahan lainnya ditopang dengan kotak engkol. Kotak engkol harus kuat dan mampu menahan getaran dari pergerakan torak pada silinder. Poros engkol ditopang dengan bantalan pada bak engkol. Pemilihan bantalan bergantung dari ukuran kompresornya. Bantalan luncur dengan terbelah dua atau empat banyak dipakai, untuk bantalan gelinding dipakai terutama yang berjenis bola.

Kompresor merupakan alat untuk melayani kebutuhan udara mampat dari mulai udara bertekanan rendah sampai udara bertekanan tinggi. Untuk peralatan pemampat udara dengan menggunakan tanki udara, apabila suplai udara bertekanan melebihi kapasitas yang dibutuhkan, maka tekanan naik tidak terkontrol pada tangki udara. Hal tersebut membahayakan yang berakibat tangki akan pecah. Untuk menghindari hal tersebut, diperlukan suatu katup pembebas beban (*unloader*). Dengan adanya alat ini maka laju udara akan teratur yang dihisap sesuai dengan laju alur aliran keluar sesuai yang dibutuhkan. Pembebasan beban ini digolongkan menurut azas kerjanya, yaitu:

1. Pembebas beban katup hisap
2. Pembebas celah katup
3. Pembebas beban trotel hisap.
4. Pembebas dengan pemutus otomatik

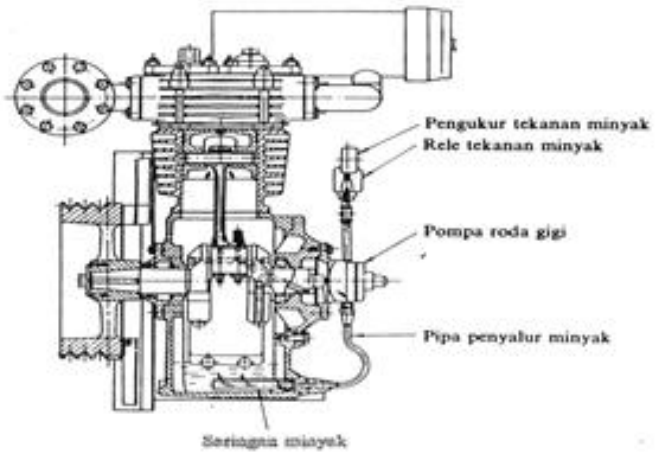
Untuk kompresor torak dengan tangki udara banyak menggunakan pembebas katup hisap dan pembebas dengan pemutus otomatik. Sedangkan untuk mengurangi beban pada waktu starter digunakan pembebas beban awal.

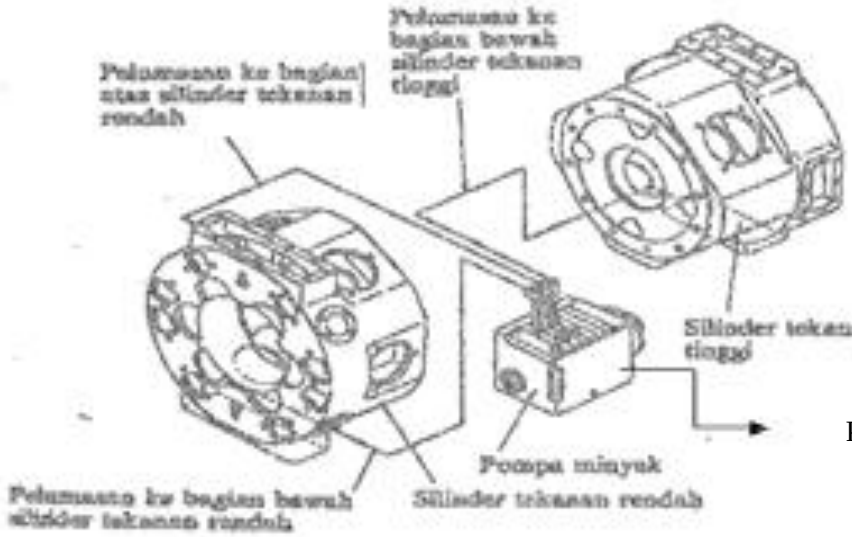
Metode pembebas katup hisap banyak dipakai pada kompresor kecil atau sedang. Proses menggunakan katup hisap, yang mana plat katupnya dapat dibuka terus pada langkah hisap atau kompresi sehingga udara dapat bergerak bebas ke luar masuk silinder tanpa terajadi kompresi.



Gambar 2.12 Pengaturan kapasitas kompresor
(Sumber: Sunyoto, dkk, 2008: 208)

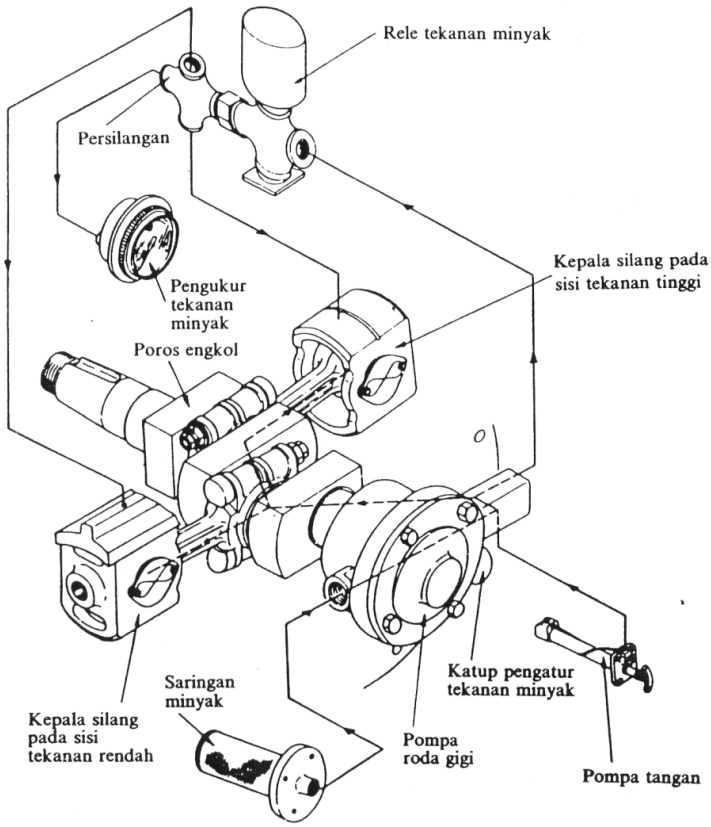
Pada saat proses tersebut katup hisap terbuka udara mampat dari tangki bebas ke luar, sehingga tekanan udara terus menurun pada tangki udara tidak dapat lagi menekan pegas pilot sehingga katup pilot pembebasan tekan tertutup.





Gambar 2.13 Pelumas paksa pada kompresor
(Sumber: Sunyoto, dkk, 2008: 209)

Komponen-komponen kompresor yang beroperasi pada beban yang tinggi menimbulkan proses pemenanasan yang cepat dikarenakan gesekan atau menerima panas dari pemampatan. Untuk mengurangi gesekan dan mendinginkan komponen-komponen seperti torak, dinding silinder, poros engkol, batang torak dan komponen terutama komponen yang bergerak, diperlukan pelumasan. Dengan pelumasan komponen-komponen akan beroperasi lebih halus, karena proses gesekan terlindungi oleh pelumas.



Gambar 2.14 Pelumasan luar kompresor torak
(Sumber: Sunyoto, dkk, 2008: 210)

Sistem pelumasan luar pada kompresor torak. Dengan metode ini minyak pelumas didistribusikan ke semua bagian komponen yang akan dilumasi dengan pompa minyak.tekanan pompa minyak diatur oleh sebuah alat pengatur tekanan. Sebelum didistribusikan minyak terlebih dahulu melalui penyaringan minyak pelumas. Jenis pompa minyak yang digunakan adalah pompa plunyer bertekanan tinggi. Untuk pelumasan luar digunakan pompa roda gigi (Sunyoto, dkk, 2008: 202-211).

Pada kompresor torak terdapat peralatan tambahan yang dipasang, diantaranya gigi (Sunyoto, dkk, 2008: 211):

1. Saringan udara, digunakan untuk menyaring udara yang dihisap kompresor sehingga udara lebih bersih dan bebas kandungan debu dan kotoran lainnya, terutama yang bersifat korosi.
2. Katup pengaman, katup ini harus ada pada instalasi kompresor.
3. Tangki udara, fungsi udara merupakan temapat menampung sekaligus pengatur kapasitas udara mampat.

2.2.2 Kerusakan Kompresor Udara

Kompresor udara dapat beroperasi secara optimal karena didukung oleh instrument pendukung yang memerlukan perawatan. Seringnya penggunaan dengan frekuensi secara berkesinambungan sehingga menyebabkan kurangnya perawatan dapat menimbulkan masalah yang dapat mempengaruhi kinerja kompresor. Berikut merupakan permasalah yang sering ditimbulkan antara lain (Sunyoto, dkk, 2008 : 222-224):

1. Pembebanan lebih dan pemanasan lebih pada motor pengerak.

Kompresor Udara merupakan suatu mesin yang bekerja dengan energi dari sumber lain. Seperti yang sudah diuraikan pada bab sebelumnya, sumber energi berupa motor penggerak yang umum digunakan secara luas adalah motor bakar dan motor listrik. Permasalahan akan muncul ada 2 kemungkin, yaitu apabila jumlah daya yang dibutuhkan kompresor adalah kurang dari harga normal operasi atau bearing motor mengalami keausan.

2. Pemanasan lebih pada udara hisap

Pada permasalahan ini, proses kompresi pada kompresor udara, semakin tinggi temperatur udara yang dihisap dengan rasio kompresi yang sama, akan menghasilkan udara mampat dengan temperatur yang lebih tinggi. Efek lainnya yang dapat terjadi adalah karena kerusakan pada katup-katup, udara tekan akan masuk silinder lagi dan dikompresikan lagi, kondisi ini menghasilkan udara tekan dengan temperatur sangat tinggi, proses kompresi bahkan berhenti sama sekali karena piston pada panas tinggi menjadi memuai dan kemudian macet.

3. Katup pengaman yang sering terbuka

Permasalahan terjadi apabila katup pembebas beban ada kerusakan karena tersumbat atau disetel pada kondisi nilai tekanan yang tinggi. Apabila hal tersebut terjadi pengontrolan tekanan menjadi kacau atau dengan kata lain tekanan berlebih di atas normal yang ke luar tidak terkontrol lagi, hal ini sangat membahayakan bagi operator kompresor.

4. Bunyi dan getaran

Kompresor bekerja untuk mengompresi udara dengan rasio tekanan tertentu. Semakin tinggi, semakin berat kerja kompresor, beban yang diterima komponen-komponen juga bertambah. Untuk kompresor Udara dengan waktu kerjanya lama, antar komponen biasanya terjadi kelonggaran (clearance) yang semakin bertambah. hal tersebut terjadi karena antar komponen saling bertumbukan, menggesek, lama kelamaan permukaan komponen tersebut mengalami abrasi dan menjadi aus. Jika proses abrasi berlangsung terus menerus

akan mengakibatkan komponen-komponen menjadi retak kemudian dapat pecah atau patah.

Pemasangan pondasi yang tidak baik juga dapat menimbulkan getaran yang merugikan. Pemasangan atara motor penggerak dengan kompresor yang tidak lurus akan menimbulkan banyak masalah terutama pada bantalan-bantalan akan terkena pembebanan yang tidak merata.

Aliran udara tekan yang melewati perpipaan juga dapat menimbulkan gangguan yaitu timbulnya resonansi di dalam pipa. Disamping itu, udara tekan yang melewati saluran yan berbelok akan menumbuk dan cenderung menimbulkan getaran apabila pondasi pipa tidak kuat.

Keausan komponen sebagian besar disebabkan oleh kurang adanya perhatian terhadap sistem pelumas dan kualitas dari pelumasnya. Pemakai pelumas yang tidak standar atau tidak tepat akan merugikan. Sebaiknya pemakaian pelumas sesuai dengan standar yang disarankan dari pabrik pembuat. Faktor penting yang perlu diperhatikan adalah penggantian minyak pelumas harus terjadwal dengan baik, sehingga kompresor beroperasi selalu dalam keadaan siap dan aman tanpa kemungkinan terjadi kerusakan.

5. Korosi

Fluida kerja dari kompresor adalah udara yang akan dimampatkan. Udara tersebut jika tercampur senyawa-senyawa asam atau basa akan sangat korosif. Apabila kompresor dalam keadaan mati, udara tekan akan mengalami pendinginan dan uap air dengan kandungan senyawa korosif yang akan mengembun dan dapat menempel pada komponen-komponen dan sebagian masuk

ke dalam minyak pelumas. Air dari pengembunan ini dapat menimbulkan korosi yaitu peristiwa bereaksinya bahan logam dengan zat korosif dan meghasilkan karat. Minyak pelumas juga berperan dalam proses korosi, hal ini terjadi jika minyak pelumas tidak terkontrol penggantiannya sehingga pelumas yang bersirkulasi banyak mengadung zat asam dan korosif terhadap logam.

6. Kapasitas kompresi rendah

Permasalahan ini terjadi pada kerusakan sistem kontrol otomatis dari kompresor sehingga menyebabkan kompresi pada kompresor rendah.

7. Katup pengaman tidak berfungsi

Kerusakan terjadi pada pengaturan sistem kontrol otomatis sehingga menyebabkan kompresor tidak bisa berhenti sehingga udara pada tabung kompresor berlebih.

8. Kompresor tidak bisa *start*

Hal ini terjadi karena beberapa sebab, diantaranya motor kelebihan beban, *fuse* putus, dan relay temperature rusak.

9. Keluaran udara dari kompresor tidak sesuai

Kerusakan ini diakibatkan oleh katup *intake* tidak sesuai beroperasi optimal sehingga pada saat katup tidak terbuka sepenuhnya, *switch* tekanan udara tidak bekerja.

2.3 *Software* Pendukung

Software pendukung merupakan beberapa perangkat lunak yang digunakan untuk mendukung pembuatan sistem pakar dalam penelitian ini. Perangkat lunak tersebut antara lain: *XAMPP, phpMyAdmin, PHP, Dreamweaver*, *MySQL,* dan *StarUML.*

2.3.1 *XAMPP* (*X Apache Mysql Php Perl*)

Gambar 2.15 Logo *XAMPP*

(Sumber:https://www.apachefriends.org/index.html)

Menurut Riyanto (2011: 1) *XAMPP* merupakan paket *PHP* dan *MySQL* berbasis *open source*, yang digunakan sebagai tool pembantu pengembangan aplikasi berbasis *PHP* dan *MySQL*. *XAMPP* mengkombinasikan beberapa paket perangkat lunak berbeda ke dalam satu paket.

2.3.2 *phpMyAdmin*

Gambar 2.16 Logo *phpMyAdmin*

(Sumber: https://www.phpmyadmin.net/static/images/logo-og.png)

phpMyAdmin adalah perangkat lunak yang ditulis dalam bahasa pemograman *PHP*, dimaksudkan untuk menangani administrasi *MySQL* melalui *Web*. *phpMyAdmin* mendukung berbagai operasi pada *MySQL* dan *MariaDB*. operasi (mengelola database, tabel, kolom, hubungan, indeks, *users*, *permissions*, dll) dapat dilakukan melalui antarmuka pengguna masih memiliki kemampuan untuk langsung mengeksekusi pernyataan *SQL* (www.phpmyadmin.net/).

2.3.4 *PHP: Hypertext Preprocessor* (*PHP*)

PHP (*Hypertext Preprocessor*) adalah bahsa script yang dapat ditanamkan atau disisipkan ke dalam *HTML*. *PHP* banyak dipakai untuk membuat program situs web dinamis. *PHP* sering juga gunakan untuk membangun sebuah *CMS*.

PHP adalah bahasa pemograman script *server-side* yang didesain untuk pengembangan *Web.* Disebut bahasa pemograman server side karrena PHP diproses pada komputer *server*. Hal ini berbeda dibandingkan dengan bahasa pemograman *client-side* seperti javascript yang diproses pada *Web Browser* (*client*). PHP dapat digunakan dengan gratis (*free*) dengan bersifat *Open Source* (Madcoms, 2016: 2).

2.3.5 *Dreamweaver CS5.5*

Dreamweaver merupakan software aplikasi yang digunakan sebagai editor *HTML* untuk medesain web secara visual. Aplikasi ini juga yang bisa dikenal dengan istilah *WYSIWYG* (*What You See Is What You Get*), yang intinya adalah bahwa pengguna tidak harus berurusan dengan tag-tag *HTML* yang cukup rumit untuk membuat sebuah halaman *web*. Selain itu Dreamweaver memberikan keleluasaan kepadapengguna untuk menggunakannya sebagai media penulisan bahasa pemograman *web*.

Dengan kemampuan fasilitas yang optimal dalam jendela desain membuat program ini memberikan kemudahan untuk mendesain *web* meskipun untuk para *web* desainer pemula sekalipun. Sedangkan kemampuan dreamweaver untuk berinteraksi dengan beberapa bahasa pemograman seperti *PHP*, *ASP*, *JavaScript*, dan yang lainnya.

Dreamweaver Creative Suite 5.5 atau disingkat biasa disingkat dengan *dreamweaver CS 5.5*. *Dreamwever CS 5.5* adalah *software* terkemuka untuk membangun dan mengedit *web* dengan menyediakan kemampuan visual dan

tingkat kode, yang dapat digunakan untuk membuat *website* berbasis standar dan desain untuk *Desktop*, *Mobile*, *Smartphone*, *Tablet*, dan perangkat lainnya (Madcoms, 2011: 2-3).

2.3.6 MySQL

Data base *MySQL* dapat dibuat menggunakan tampilan jendela *php MyAdmin* atau menggunakan sebuah script *PHP*. Untuk membuat dan menyiapkan database dengan menggunakan jendela *phpMyAdmin* karena akan lebih mudah langkah-langkah penggunaanya.

Penyimpanan data yang fleksibel dan cepat aksesnya sangat dibutuhkan dalam sebuah *website* yang interaktif dan dinamis. Database sendiri berfungsi sebagai penampung data yang anda inputkan melalui form *website*. Selain itu dapat juga dibalik dengan menampilkan data yang tersimpan dalam database ke dalam halaman *website*. Jenis database yang sangat populer digunakan pada banyak website di internet sebagai bank data adalah *MySQL*.selain itu *MySQL* dapat berjalan di berbagai platform, antara lain *Linux*, *Windows*, dan sebagainya (Madcoms, 2011: 260-261).

1. Memasukkan data (*insert*)

Bentuk *query* (perintah) untuk memasukkan data dapat dimodelkan sebagai berikut:

```
INSERT INTO nama_tabel(kolom1, kolom2, ...,kolomN)
VALUES (‘isikolom1’, ‘isikolom1’, …, ‘isikolomN’);
```

2. Mengubah data (*update*)

Bentuk *query* untuk mengubah data dapat dimodelkan sebagai berikut:

```
UPDATE nama_tabel
SET
        kolom1 = ‘isikolom1’
WHERE
        kolom2 = ‘isikolom2’;
```

3. Menghapus data (*delete*)

Bentuk *query* untuk menghapus data dapat dimodelkan sebagai berikut:

```
DELETE FROM nama_tabel
WHERE
        kolom1 = ‘isikolom1’;
```

4. Menampilkan data (*select*)

Bentuk *query* untuk menampilkan data dapat dimodelkan sebagai berikut:

```
SELECT kolom1, kolom2
FROM nama_tabel
WHERE
        kolom1 = ‘isikolom1’;
```

2.3.7 *StarUML*

UML (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek, (A.S. dan Shalahuddin, 2013: 133).

Munculnya *UML* dilandasi oleh kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak (A.S. dan Shalahuddin, 2013: 137-138).

Gambar 2.17 Logo *StarUML*

(Sumber: http://staruml.sourceforge.net/image/staruml-logo.jpg)

StarUML merupakan salah satu *CASE* (*Computer-Aided Software Engineering*) *tools* atau perangkat pembantu berbasis komputer untuk rekayasa perangkat lunak yang mendukung alur hidup perangkat lunak (*life cycle support*). *StarUML* termasuk ke dalam kelompok *upper CASE tools* yang mendukung perencanaan strategis dan pembangunan perangkat lunak (A.S. dan Shalahuddin, 2013: 122-123).

Terdapat 13 macam diagram dalam *UML 2.3* yang dibagi menjadi 3 kategori yaitu (A.S. dan Shalahuddin, 2013: 140-141):

1. *Structure diagrams*

Kategori ini terdiri dari kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Diagram UML yang termasuk dalam kategori ini antara lain *class diagram, object diagram, component diagram, composite structure diagram, package diagram,* dan *deployment diagram*.

2. *Behaviour diagrams*

Kategori ini terdiri dari kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada

sebuah sistem. Diagram UML yang termasuk dalam kategori ini antara lain *use case diagram, acitivity diagram,* dan *state machine diagram*.

3. *Interaction diagrams*

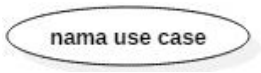
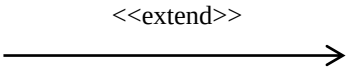
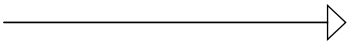
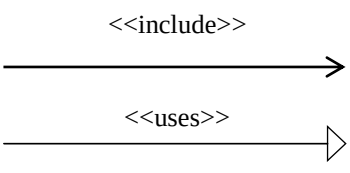
Kategori ini terdiri dari kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. Diagram UML yang termasuk dalam kategori ini antara lain *sequence diagram, communication diagram, timing diagram,* dan *interaction overview diagram*.

Menurut A.S. dan Shalahudin (2013: 18) *use case* dan *sequence diagram* merupakan bagian dari desain sistem. Dalam penelitian ini, diagram yang akan digunakan untuk desain sistem yaitu:

1. *Use case diagram*

Use case diagram merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Ada 2 hal utama yang terdapat pada *use case* yaitu aktor dan *use case*. Berikut ini adalah simbol-simbol yang digunakan dalam *use case diagram* (A.S. dan Shalahuddin, 2013: 155).

Tabel 2.3 Simbol Use Case Diagram

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor
asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor
Ekstensi/*extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu. Arah panah mengarah pada *use case* yang ditambahkan.
generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara 2 buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari fungsi lainnya. Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
Menggunakan/*include*/*uses* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* ini. Arah panah mengarah pada *use case* yang ditambahkan

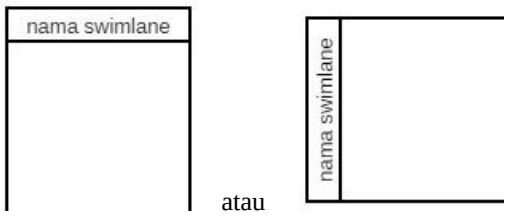
Sumber: A.S. dan Shalahuddin (2013: 162)

2. *Activity diagram*

Activity diagram merupakan diagram yang menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis. Jadi dapat

dikatakan bahwa *activity diagram* menggambarkan aktifitas sistem, akan tetapi bukan kegiatan yang dilakukan oleh aktor. Simbol-simbol yang digunakan dalam *activity diagram* ditampilkan dalam tabel berikut (A.S. dan Shalahuddin: 2013: 161-162).

Tabel 2.4 Simbol Activity Diagram

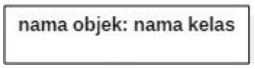
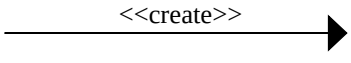
Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktifitas memiliki sebuah status awal
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu
Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktifitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktifitas memiliki sebuah status akhir
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi

Sumber: A.S. dan Shalahuddin (2013: 162-163)

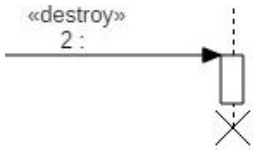
3. *Sequence diagram*

Sequence diagram menggambarkan kelakuan objek pada *use csase* dengan mendeskripsikan waktu hidup objek dan *message* (pesan) yang dikirimkan dan diterima antar objek. Banyaknya jumlah *sequence diagram* yang harus digambar minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri. Semakin banyak *use case* terdefinisi, semakin banyak pula *sequence diagram* yang harus dibuat. Simbol-simbol yang terdapat pada *sequence diagram* ditampilkan dalam tabel berikut (A.S. dan Shalahuddin, 2013: 165-166).

Tabel 2.5 Simbol Sequence Diagram

Simbol	Deskripsi
Aktor/*actor*  	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Garis hidup/*lifeline* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor
Objek  	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. Aktor tidak memiliki waktu aktif
Pesan tipe *create*  	Menyatakan suatu objek membuat objek yang lain. Arah panah mengarah pada objek yang dibuat

Tabel 2.6 Lanjutan

pesan tipe *call* 1 : nama_metode()	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi/metode.
Pesan tipe *send* 1 : masukan	Menyatakan bahwa suatu ojek mengirimkan data/masukan/informasi ke objek lainnya. Arah panah mengarah pada objek yang dituju
pesan tipe *return* 1 : keluaran	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu. Arah panah mengarah pada objek penerima
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek lain. Arah panah mengarah pada objek yang diakhiri

Sumber: A.S. dan Shalahuddin (2013: 166- 167)

2.4 Penellitian Terdahulu

Sebagai bahan pertimbangan dan teori pendukung dalam penelitian ini, maka penulis mencantumkan beberapa penelitian yang diambil dari beberapa jurnal ilmiah, yaitu:

Ahmad Hilmi , Dini Destiani (2015), Pengembangan Sistem Pakar Diagnosis Kerusakan Motor *Automatic* Non Injeksi berbasis Android, mobilitas hampir tidak mungkin dilakukan jika tidak menggunakan alat transportasi. Sebagian masyarakat telah menjadikan sepeda motor sebagai alat transportasi utama, sering terjadi kendala dari kendaraan sepeda motor itu yang dapat menyebabkan kerusakan sehingga dapat mengganggu aktifitas yang akan

dilakukan oleh pengguna. Masih banyak pengendara sepeda motor yang belum mengetahui berbagai kendala kerusakan yang dialami oelh kendaraan sepeda motor tersebut. Bagi pengendara yang tidak mengetahui jenis kerusakan itu akanlah sengat fatal apabila jenis kerusakan-kerusakan tersebut tidak ditangani. Metode penalaran menggunakan *forward chaining* berbasis android.

Guntur, Nita Merlina (2016), Sistem Pakar Diagnosa Kerusakan pada Mesin Pendigin Ruangan dengan Metode *Forward Chaining*. Pendingin ruangan AC (*Air Conditioner)* semakin dibutuhkan pada saat ini. Gangguan pada mesin pendingin ruangan akan mempengaruhi kenyamanan masyarakat, terlebih jika masyarakat tidak mengetahui bagaimana gejala awal kerusakan mesin pending ruangan. Seorang pakar dipanggil diaharapkan mampu cepat mendiagnosa kerusakan. Begitu juga dengan masyarakat umum dapat menangani masalah-masalah kecil yang terjadi. Metode yang digunakan adalah *Forward Chaining.*

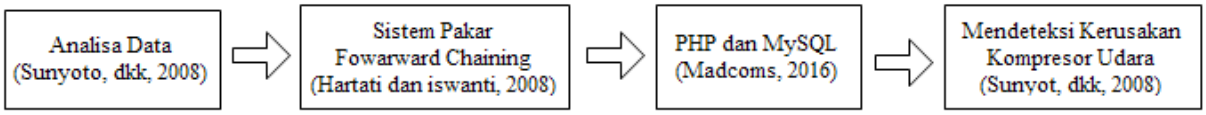
Ekka Pujo Ariesanto Akhmad, M. Taufik (2010), Pengembangan Sistem Pakar untuk Kerusakan Mesin Diesel. Kelangsungan dan keberhasilan operasi suatu kapal bergantung pada kemampuan individu yang dipekerjakan untuk menangani operasi kapal berasal dari pelatihan dan pembelajaran dari lembaga pendidikan, dan berdasarkan pengalaman. Untuk menangani kerusakan pada mesin diesel, maka perbaikan harus sesuai dengan diagnosis atau pelacakan kerusakan yang diberikan oleh pakar. Metode inferensi yang digunakan adalah *forward chaining* berbasis VB 6.0.

R.M. Nasrul Halim (2011), Sistem Pakar untuk Medeteksi Kerusakan Peralatan Elektronika dengan Bahasa Pemograman Visual Basic 6.0. Pengembangan sistem menjadi sesuatu yang sangat sulit untuk diimplementasikan, karena masih adanya keterbatasan sistem baik perangkat keras maupun perangkat lunak dalam proses pengolahan data. Dengan memanfaatkan sistem pakar diharapkan dapat membantu pengguna agar dapat bekerja dengan cepat serta diperoleh data sesuai dengan yang diinginkan.

Uky Yudatama (2008), Sistem pakar untuk Mediagnosis Kerusakan Mesin Mobil Panther Berbasis Mobile. Kerusakan pada mobil terjadi akibat kelalaian dalam melakukan perawatan. Pemilik mobil baru menyadari kerusakan mobil kemungkinan tidak dapat beroperasi sebagaimana mestinya. Oleh karena itu dalam pengunaan mobil kemungkinan besar membutuhkan perawatan berkala. Dengan cara mendeteksi kerusakan apa yang terjadi pada mesin mobil. Penyampaian informasi pun dilakukan mengunakan perangkat mobile dengan meminta request dari *user*. Metode yang digunakan *forward chaining* dan *backward chaining* menggunakan bahasa pemograman PHP berbasis Mobile.

2.5 Kerangka Pemikiran

Menurut Suriasumantri dalam (Sudaryono: 2015: 21) kerangka pemikiran adalah penjelasan yang bersifat sementara mengacu pada gejala-gejala yang menjadi objek permasalahan, secara teoritis menjelaskan tentang hubungan antar variabel yang dibangun dari beberapa teori yang telah dideskripsikan.



Gambar 2.18 Kerangka Pemikiran
Sumber: Data Penelitian (2016)