BAB II

LANDASAN TEORI

1.1 Profil Tempat Penelitian

Pada tinjauan tempat penelitian ini akan dibahas mengenai sejarah berdirinya, struktur organisasi serta uraian tugas dari masing-masing bagian yang terlibat.

1.1.1 Sejarah Singkat Klinik dr.Suzanna T.Ngadiman M.Kes

Klinik dr.Suzanna T.Ngadiman M.Kes sebagai salah satu instansi di bidang kesehatan didirikan pada bulan April 2001. Klinik ini didirikan karena melihat belum adanya klinik terdekat, sehingga masyarakat sekitar sedikit kesulitan dalam hal pengobatan. Mengenai sejarah berdirinya klinik, tidak banyak informasi yang dapat dikumpulkan, karena tidak adanya bukti tertulis (arsip).

Klinik dr.Suzanna T.Ngadiman M.Kes mempunyai misi menyelenggarakan pelayanan prima, terjangkau dan memuaskan, sehingga diharapkan masyarakat sekitar dapat terbantu dengan adanya klinik tersebut.

Seperti umumnya suatu instansi, Klinik dr.Suzanna T.Ngadiman M.Kes juga memiliki struktur organisasi. Organisasi adalah sistem saling pengaruh mempengaruhi antara orang dalam kelompok kerjasama untuk mencapai suatu tujuan tertentu yang sama. Struktur organisasi menunjukan bagaimana departemen-departemen / bagian-bagian dalam organisasi dikoordinasikan bersama-sama melalui suatu jalur wewenang dan tanggung jawab.

1.1.2 Struktur Organisasi Klinik dr.Suzanna T.Ngadiman M.Kes

Secara garis besar, komponen yang terlibat beserta tugas dan wewenangnya dapat dijelaskan sebagai berikut :

1. Penanggung Jawab

Penanggung jawab memiliki tugas dan wewenang:

a. Menetapkan kebijakan dan menentukan rencana jangka panjang bagi klinik
b. Memberikan tugas atau wewenang kepada bagian-bagian serta mengawasi tugas yang diberikan dalam pelaksanaanya yang dikerjakan oleh masing-masing bagian
c. Mensahkan laporan yang telah dibuat oleh setiap bagian
d. Bertanggung jawab terhadap seluruh kegiatan klinik

2. Dokter

Dokter mempunyai tugas dan wewenang:

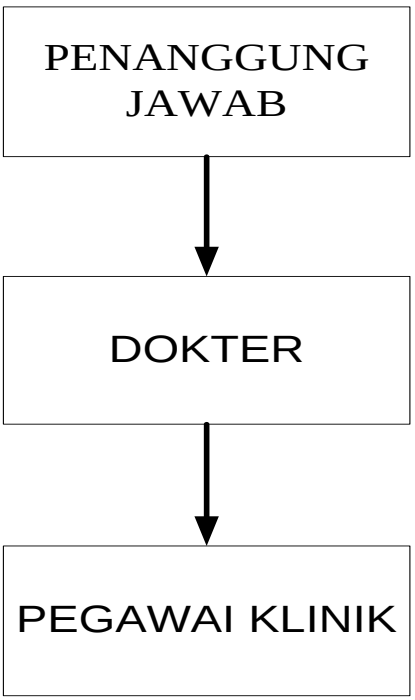
a. Menangani pasien setiap hari.
b. Bertanggung jawab terhadap riwayat kesehatan pasien (rekam medis pasien)

3. Pegawai Klinik

Petugas Administrasi mempunyai tugas dan wewenang:

a. Mengatur segala kegiatan administrasi klinik serta kearsipan klinik
b. Melakukan pengecekan terhadap stok obat, data pasien, rekam medis pasien.
c. Membuat laporan-laporan

Struktur Organisasi Klinik dr.Suzanna T.Ngadiman M.Kes dapat digambarkan seperti gambar dibawah 2.1



Gambar 2. 1 Struktur Organisasi Klinik

1.1.3 Definisi Penyakit dalam Ilmu Kedokteran

Penyakit adalah suatu keadaan atau kondisi tidak wajar (abnormal) dari tubuh manusia yang menyebabkan rasa tidak nyaman, rasa kesakitan, disfungsi organ-organ tubuh manusia. [4]

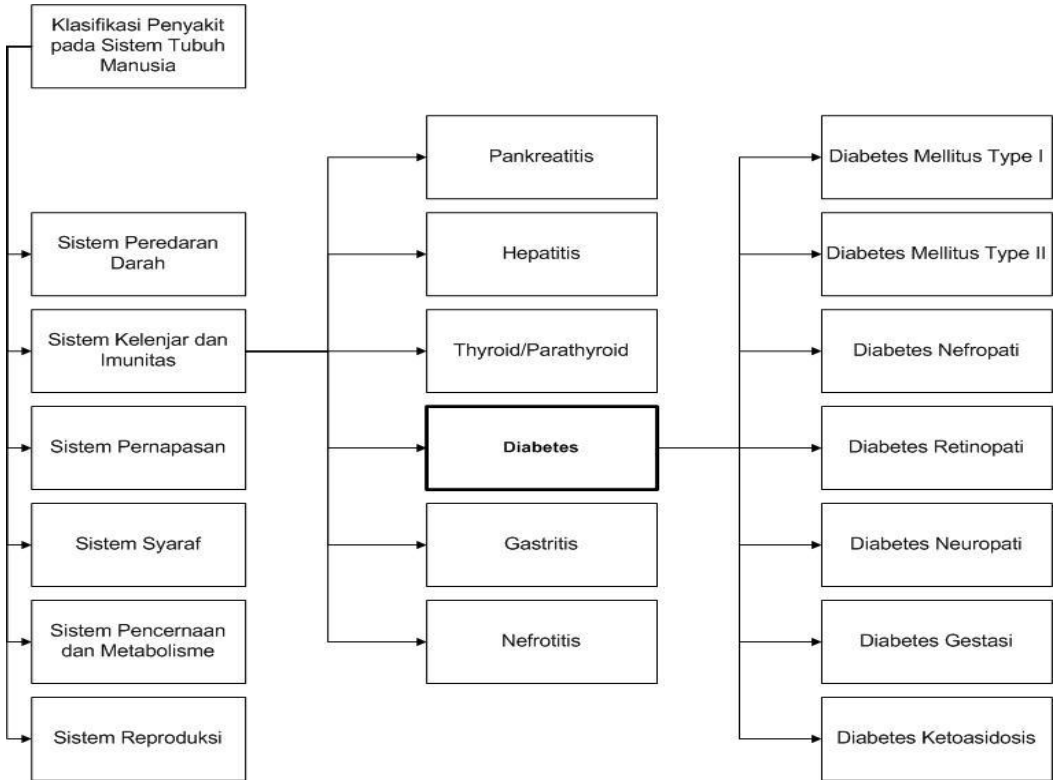
Jenis penyakit dalam tubuh manusia sangat bermacam-macam. Hingga saat ini ada lebih 10.000 jenis penyakit yang sudah diketahui penyebabnya dan cara penanganan terhadap penyakit tersebut. Tiap-tiap penyakit memiliki gejala yang sangat bervariasi mulai dari yang *simple* sampai pada kombinasi beberapa gejala penyakit yang ada.

Mengetahui gejala penyakit bisa berdasarkan pada faktor psikis atau pun berdasarkan pada faktor klinis. Faktor psikis adalah gejala abnormal dari tubuh

(fisik) yang terlihat oleh mata, seperti demam, batuk, pembengkakan organ-organ tubuh, dan lain-lain. Sedangkan faktor klinis merupakan gejala abnormal yang hanya bisa diketahui melalui test-test yang dilakukan seperti test darah, test urine, test kotoran, dan lain-lain.

1.1.3.1 Klasifikasi Penyakit Berdasarkan Sistem Tubuh Manusia

Adapun klasifikasi penyakit berdasarkan sistem tubuh manusia dapat dilihat dalam gambar berikut :



Gambar 2. 2 Gambar Klasifikasi Penyakit

Pada Gambar 2.2, terlihat penyakit diabetes yang merupakan domain sistem pakar yang akan dibangun dalam tugas akhir ini, termasuk dalam penyakit pada sistem kelenjar dan imunitas.

1.1.3.2 Diabetes

Diabetes atau sering disebut sebagai penyakit kencing manis atau penyakit gula darah, merupakan golongan penyakit kronis yang ditandai dengan peningkatan kadar gula dalam darah sebagai akibat adanya gangguan system metabolisme dalam tubuh. Tahun 400 SM, seorang penulis India menamainya "penyakit kencing madu". Nama diabetes mellitus sendiri (diabetes=mengalir terus,mellitus= manis) akhirnya diberikan oleh Areteus sekitar tahun 200 SM [10]. Secara garis besar diabetes dapat dibagi menjadi dua jenis, yaitu :

1. Diabetes Mellitus tipe 1.
2. Diabetes Mellitus tipe 2.

Namun pada kenyataanya, sering terjadi komplikasi penyakit lainnya yang dirasakan oleh pasien penderita diabetes. Hal ini berkaitan dengan kadar gula darah yang tinggi terus menerus, sehingga berakibat rusaknya pembuluh darah, saraf dan struktur internal lainnya. Penderita diabetes bisa mengalami berbagai komplikasi jangka panjang jika diabetesnya tidak dikelola dengan baik [2].

Komplikasi yang dapat timbul antara lain :

1. Komplikasi pada wanita hamil (Gestasi Diabetes).
2. Komplikasi pada jaringan saraf (Neuropati Diabetes).
3. Komplikasi pada mata (Retinopati Diabetes).
4. Komplikasi pada ginjal (Nefropati Diabetes).
5. Komplikasi pada metabolisme keton (Ketoasidosis Diabetes).

Sebenarnya masih banyak lagi komplikasi yang terjadi, hanya saja dipilih beberapa komplikasi yang dinilai paling sering terjadi pada penderita diabetes.

Umumnya pengobatan yang dilakukan pada penderita diabetes adalah menggunakan obat hipoglikemik oral atau sering disingkat dengan OHO. Namun untuk beberapa kondisi, pengobatan divariasikan juga dengan terapi insulin [10]. Saat ini dikenal obat OHO yaitu :

1. Golongan *Sulphoniluria* (generasi 1,2,3) misalnya *Daonil, DiamicronAmaryl.*
2. Golongan *Biguanid*, misalnya *Glucophage.*
3. Golongan *Alphaglukosidase Inhibitor*, misalnya *Glucobay.*
4. Golongan *Thiazolidiones, Pioglitazone (Actos), Rosiglitazone (Avandia)*
5. Golongan *Glinid Repaglinid*, misalnya *Novonorm*
6. Golongan *Incretin* / penghambat enzim DPP-4, *sitagliptin* (*Januvia*), dan *Vidagliptin (Galvus).*

1.2 Kecerdasan Buatan

Pada topik ini akan membahas tentang definisi kecerdasan buatan, konsep kecerdasan buatan dan lingkup kecerdasan buatan.

1.2.1 Definisi Kecerdasan Buatan

Kecerdasan Buatan atau Artificial Intelligence didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Kecerdasan buatan ini merupakan cabang dari ilmu komputer yang concerned dengan pengotomatisasi tingkah laku cerdas. Karena itu kecerdasan buatan harus didasarkan pada prinsip-prinsip teoretikal dan terapan yang menyangkut struktur data yang digunakan dalam

representasi pengetahuan (knowledge representation), algoritma yang diperlukan dalam penerapan pengetahuan itu, serta bahasa dan teknik pemrograman yang dipakai dalam implementasinya [3]. Pada awal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan jaman, maka peran komputer semakin mendominasi kehidupan umat manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, lebih dari itu, komputer diharapkan untuk dapat diberdayakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia .

Lebih detailnya, pengertian kecerdasan buatan dapat dipandang dari berbagai sudut pandang, antara lain:

1. Sudut pandang kecerdasan

   Kecerdasan buatan akan membuat mesin menjadi ‘cerdas’ (mampu berbuat seperti apa yang dilakukan oleh manusia).

2. Sudut pandang penelitian

   Kecerdasan buatan adalah suatu studi bagaimana membuat agar komputer dapat melakukan sesuatu sebaik yang dikerjakan oleh manusia.

3. Sudut pandang bisnis

   Kecerdasan buatan adalah kumpulan peralatan yang sangat powerful dan metodologis dalam menyelesaikan masalah-masalah bisnis.

4. Sudut pandang pemrograman

   Kecerdasan buatan meliputi studi tentang pemrograman simbolik, penyelesaian masalah (problem solving) dan pencarian (searching).

Untuk melakukan aplikasi kecerdasan buatan ada dua bagian utama yang sangat dibutuhkan yaitu:

a. Basis pengetahuan (knowledge base), berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan lainnya.
b. Motor inferensi (inference engine), yaitu kemampuan menarik kesimpulan berdasarkan pengalaman.

1.2.2 Konsep Kecerdasan Buatan

Ada beberapa konsep yang harus dipahami dalam kecerdasan buatan, diantaranya:

1. Turing Test – Metode Pengujian Kecerdasan.

   Turing Test merupakan sebuah metode pengujian kecerdasan yang dibuat oleh Alan Turing. Dalam konsep ini, penanya (manusia) akan diminta untuk membedakan yang mana merupakan jawaban manusia dan mana yang merupakan jawaban komputer. Apabila tidak dapat membedakan, maka Turing berpendapat bahwa mesin tersebut dapat di asumsikan CERDAS.

2. Pemprosesan Simbolik

   Komputer semula didesain untuk pemprosesan numerik, sedangkan manusia dalam berfikir dan menyelesaikan masalah lebih bersifat simbolik. Sifat penting dari AI adalah bagian dari ilmu komputer yang melakukan proses secara simbolik dan non algoritmik dalam menyelesaikan masalah.

3. Heuristic

   Heuristic merupakan sustu strategi untuk melakukan proses pencarian (search) ruang problem secara selektif, yang memandu proses pencarian yang kita lakukan sepanjang jalur yang memiliki kemungkinan sukses paling besar.

4. Penarikan Kesimpulan (Inferencing)

   AI mencoba membuat mesin memiliki kemampuan berfikir atau mempertimbangkan (reasoning). Kemampuan berfikir (reasoning) termasuk didalamnya proses penarikan kesimpulan (inferencing) berdasarkan fakta-fakta dan aturan dengan menggunakan metode hueristik atau pencarian lainnya.

5. Pencocokan Pola (Pattern Matching)

   AI bekerja dengan metode pencocokan pola (Pattern Matching) yang berusaha untuk menjelaskan objek, kejadian (event) atau proses, dalam hubungan logik atau komputasional.

1.2.3 Lingkup Kecerdasan Buatan pada Aplikasi Komersial

Makin pesatnya perkembangan teknologi menyebabkan adanya perkembangan dan perluasan lingkup yang membutuhkan kehadiran kecerdasan buatan. Karakteristik ‘cerdas’ sudah mulai dibutuhkan di berbagai disiplin ilmu dan teknologi. Kecerdasan buatan tidak hanya dominan di bidang ilmu komputer (informatika), namun juga sudah merambah di berbagai disiplin ilmu yang lain. Irisan antara psikologi dan kecerdasan buatan melahirkan sebuah area yang dikenal dengan nama Cognition &

Psycolinguistics. Irisan antara teknik elektro dengan kecerdasan buatan melahirkan berbagai ilmu seperti: pengolahan citra, teori kendali, pengenalan pola dan robotika.

Adanya irisan penggunaan kecerdasan buatan di berbagai disiplin ilmu tersebut menyebabkan cukup rumitya untuk mengklasifikasikan kecerdasan buatan menurut disiplin ilmu yang menggunakannya. Untuk memudahkan hal tersebut, maka pengklasifikasian lingkup kecerdasan buatan didasarkan pada output yang diberikan yaitu aplikasi komersial (meskipun sebenarnya kecerdasan buatan itu sendiri bukan merupakan medan komersial). Lingkup utama dalam kecerdasan buatan adalah:

1. Sistem Pakar (Expert System). Disini komputer digunakan sebagai sarana untuk menyimpan pengetahuan para pakar. Dengan demikian komputer akan memiliki keahlian untuk menyelesaikan permasalahan dengan meniru keahlian yang dimiliki oleh pakar.
2. Pengolahan Bahasa Alami (Natural Language Processing). Dengan pengolahan bahasa alami ini diharapkan user dapat berkomunikasi dengan komputer dengan menggunakan bahasa sehari-hari.
3. Pengenalan Ucapan (Speech Recognition). Melalui pengenalan ucapan diharapkan manusia dapat berkomunikasi dengan komputer dengan menggunakan suara.
4. Robotika & Sistem Sensor (Robotics & Sensory Systems).
5. Computer Vision, mencoba untuk dapat menginterpretasikan Gambar atau objek-objek tampak melalui komputer.

6. Intelligent Computer-Aided Instruction. Komputer dapat digunakan sebagai tutor yang dapat melatih dan mengajar.
7. Game Playing.

1.3 Sistem Pakar

Pada subbab ini akan diuraikan teori yang berkaitan dengan sistem pakar, mulai dari pengertian itu sendiri hingga sejarahnya

1.3.1 Definisi Sistem Pakar

Sistem Pakar adalah sebuah perangkat lunak komputer yang memiliki basis pengetahuan untuk domain tertentu dan menggunakan penalaran inferensi menyerupai seorang pakar dalam memecahkan masalah. Seorang pakar adalah seseorang yang memiliki pengetahuan dan pengalaman spesifik dalam suatu bidang, misalnya pakar kesehatan, pakar komputer, dan lain-lain. Ketika hendak membuat suatu keputusan yang kompleks atau memecahkan masalah seringkali kita meminta nasehat atau berkonsultasi dengan seorang ahli atau pakar. Keahlian/kepakaran (*expertise*) diperoleh melalui pengalamannya selama bertahun-tahun pada suatu bidang tertentu melalui serangkaian pelatihan (*training*), pembelajaran (*learning*), dan lain-lain. Jadi, Sistem Pakar dibuat untuk mentransfer keahlian / kepakaran dari seorang pakar ke komputer, kemudian ke orang lain yang bukan pakar [1].

Walaupun sistem pakar menyajikan saran / nasehat kepada pengguna, tetapi memiliki perbedaan dengan sistem pendukung keputusan (*Decision Support System*). Berikut perbedaan sistem pakar dengan sistem pendukung keputusan [12]:

Tabel 2. 1 Perbedaan Decision Support System (DSS) dengan Sistem Pakar

Karakteristik	***Decision Support System* (DSS)**	**Sistem Pakar (*Expert System*)**
Fungsi	Mencerminkan gaya dan kemampuan manajer untuk memecahkan masalah	Membuat keputusan melebihi kemampuan manajer
Alur Penalaran	Tidak terperinci	Terperinci dengan jelas
Kemampuan member alasan	Tidak ada	Ada, terbatas
Kemampuan menjelaskan	Terbatas	Ada
Pembuat Keputusan	Manusia dan/atau sistem	Sistem
Metode manipulasi data	Numerik	Simbolik

Seiring perkembangan zaman, sistem pakar memiliki sub sistem masing-masing. Secara umum, kategori sub sistem tersebut terbagi menjadi :

1. Interpretasi
2. Proyeksi
3. Diagnosis
4. Perencanaan
5. Monitoring
6. Desain
7. Instruksi
8. Pengendalian/control
9. Simulasi
10. Seleksi

1.3.2 Konsep dasar Sistem Pakar

Konsep dasar dari sistem pakar yaitu meliputi keahlian *(expertise),* ahli *(experts),* pemindahan keahlian *(transfering expertise),* inferensi *(inferencing),* aturan *(rules)* dan kemampuan memberikan penjelasan *(explanation capability).*

Keahlian *(expertise)* adalah pengetahuan yang mendalam tentang suatu masalah tertentu, dimana keahlian bisa diperoleh dari pelatihan/ pendidikan, membaca dan pengalaman dunia nyata. Ada dua macam pengetahuan yaitu pengetahuan dari sumber yang ahli dan pengetahuan dari sumber yang tidak ahli. Pengetahuan dari sumber yang ahli dapat digunakan untuk mengambil keputusan dengan cepat dan tepat.

Ahli *(experts)* adalah seorang yang memiliki keahlian tentang suatu hal dalam tingkatan tertentu, ahli dapat menggunakan suatu permasalahan yang ditetapkan dengan beberapa cara yang berubah-ubah dan merubahnya kedalam bentuk yang dapat dipergunakan oleh dirinya sendiri dengan cepat dan cara pemecahan yang mengesankan.

Ahli seharusnya dapat untuk menjelaskan hasil yang diperoleh, mempelajari sesuatu yang baru tentang domain masalah, merestrukturisasi pengetahuan kapan saja yang diperlukan dan menentukan apakah keahlian mereka relevan atau saling berhubungan.

1.3.3 Tujuan Sistem Pakar

Tujuan dari sistem pakar adalah untuk memindahkan kemampuan *(transferring expertise)* dari seorang ahli atau sumber keahlian yang lain ke dalam

komputer dan kemudian memindahkannya dari komputer kepada pemakai yang tidak ahli (bukan pakar). Proses ini meliputi empat aktivitas yaitu:

1. Akuisi pengetahuan *(knowledge acquisition)* yaitu kegiatan mencari dan mengumpulkan pengetahuan dari para ahli atau sumber keahlian yang lain.
2. Representasi pengetahuan *(knowledge representation)* adalah kegiatan menyimpan dan mengatur penyimpanan pengetahuan yang diperoleh dalam komputer. Pengetahuan berupa fakta dan aturan disimpan dalam komputer sebagai sebuah komponen yang disebut basis pengetahuan.
3. Inferensi pengetahuan *(knowledge inferencing)* adalah kegiatan melakukan inferensi berdasarkan pengetahuan yang telah disimpan didalam komputer.
4. Pemindahan pengetahuan *(knowledge transfer)* adalah kegiatan pemindahan pengetahuan dari komputer ke pemakai yang tidak ahli.

1.3.4 Bentuk Sistem Pakar

1. Mandiri merupakan sistem pakar yang murni berdiri sendiri, tidak digabung dengan perangkat lunak lain, bisa dijalankan pada komputer pribadi dan mainframe.
2. Terkait atau tergabung merupakan sistem pakar hanya bagian dari program yang lebih besar. Program tersebut biasanya menggunakan teknik algoritma konvensional tapi bisa mengakses sistem pakar yang ditempatkan sebagai subrutin, yang bisa dimanfaatkan setiap kali dibutuhkan.
3. Terhubung adalah sistem pakar yang berhubungan dengan software lain. Misalnya spreadsheet, DBMS, program grafik. Pada saat proses inferensi,

sistem pakar bisa mengakses data dalam spreadsheet atau DBMS atau program grafik bisa dipanggil untuk menayangkan output visual.

4. Sistem Mengabdi Merupakan bagian dari komputer khusus yang diabdikan kepada fungsi runggal. Sistem tersebut bisa membantu analisa data radar dalam pesawat tempur atau membuat keputusan intelejen tentang bagaimana memodifikasi pernbangunan kimiawi.

1.3.5 Ciri-ciri Sistem Pakar

Adapun ciri-ciri dari Sistem Pakar adalah sebagai berikut [3]:

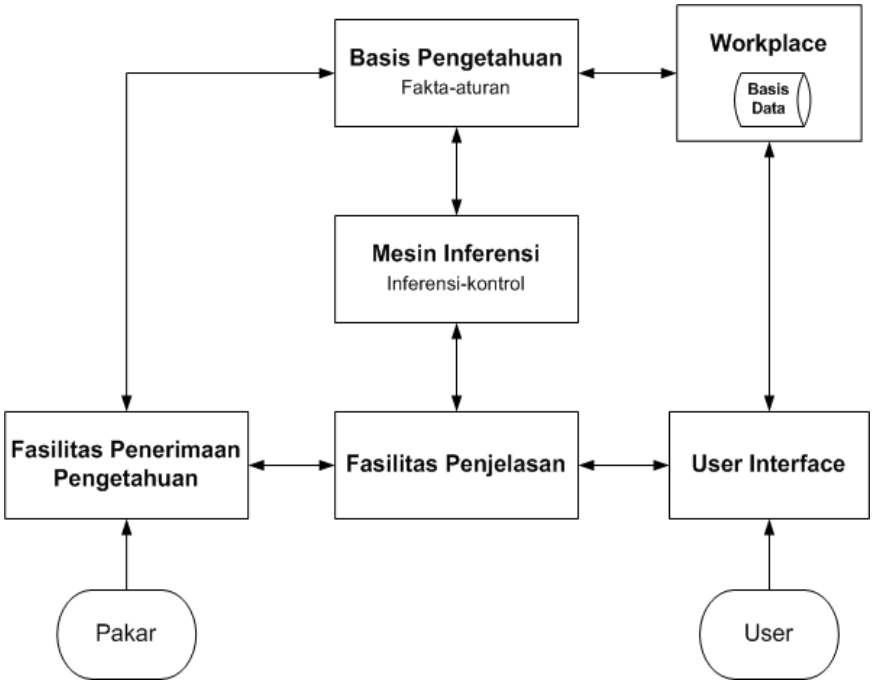
1. Spesifik pada domain keahlian tertentu.
2. Dapat memberikan penalaran-penalaran yang tidak pasti.
3. Dapat mengemukakan rangkaian alasan-alasan yang diberikannya dengan cara yang dapat dipahami.
4. Mempunyai kaidah/rule tertentu.
5. Keluarannya bersifat saran, nasehat, atau solusi permasalahan.
6. Dirancang untuk dapat dikembangkan secara bertahap dan mudah dimodifikasi.
7. Memiliki kemampuan untuk beradaptasi.

1.3.6 Arsitektur Sistem Pakar

Ada empat komponen utama dalam system pakar, yaitu [1] :

1. Antarmuka pengguna (*User Interface*).
2. Basis pengetahuan (*Knowledge Base*).
3. *Workplace*.
4. Mesin Inferensi (*Inference Engine*).

Arsitektur sistem pakar dapat digambarkan sebagai berikut :



Gambar 2. 3 Arsitektur Sistem Pakar

1.3.6.1 Antarmuka Pengguna (User Interface)

User interface merupakan penghubung antara program system pakar dengan pengguna (user). Pada bagian ini akan terjadi dialog antara program dengan pengguna. Program akan menanyakan pertanyaan yang nantinya harus dijawab oleh pengguna kemudian menyajikan informasi yang dapat dimengerti oleh pengguna. Suatu *interface* yang baik haruslah di desain bersifat *user friendly*, artinya mudah digunakan oleh pengguna.

1.3.6.2 Basis Pengetahuan (Knowledge Base)

Basis pengetahuan merupakan inti program dari system pakar ini dimana basis pengetahuan merupakan representasi pengetahuan (*Knowledge Representation*) dari seorang pakar. Basis pengetahuan ini tersusun atas fakta dan aturan (*rule*). Fakta adalah informasi tentang objek dalam suatu permasalahan,

sedangkan aturan (*rule*) adalah cara bagaimana memperoleh fakta baru dari fakta yang sudah diketahui [12]. Basis pengetahuan mendapatkan fakta tersebut melalui seorang pakar yang mentransferkan pengetahuannya melalui tahapan akuisisi pengetahuan (*knowledge acquistion*). Akuisisi pengetahuan (*knowledge acquisition*) adalah proses transfer, akumulasi dan transformasi keahlian dari seorang pakar ke dalam program komputer. Akusisi pengetahuan dilakukan selama proses pengembangan system. Ada beberapa metode utama dalam proses akusisi pengetahuan, yaitu:

1. Wawancara.
2. Observasi.
3. Analisis.
4. Induksi dari contoh.

1.3.7 Workplace

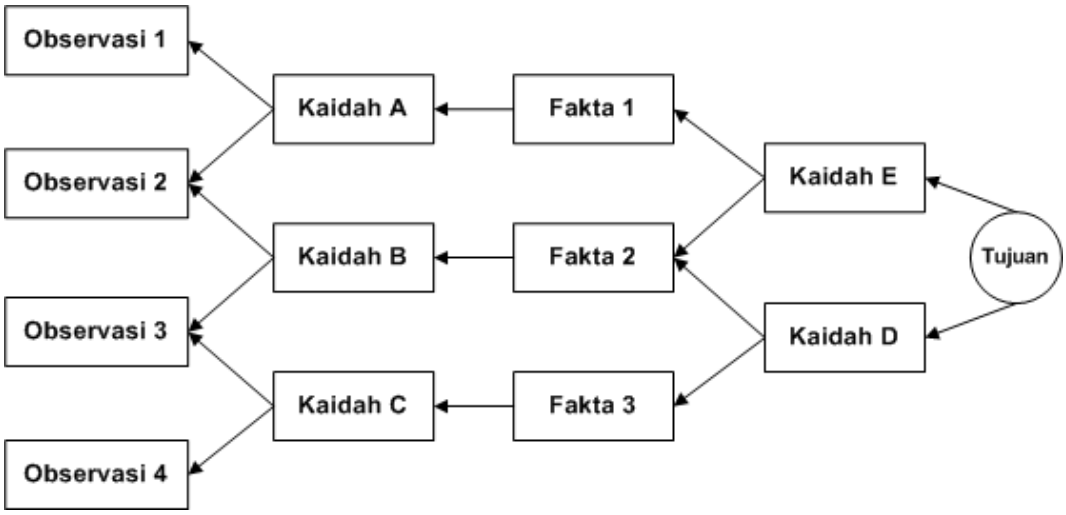
Workplace merupakan area dari sekumpulan memori kerja (*working memory*) yang digunakan untuk merekam hasil-hasil observasi dan kesimpulan yang dicapai, kemudian menyimpannya ke dalam basis data.

1.3.7.1 Mesin Inferensi (Inference Engine)

Mesin inferensi adalah bagian yang mengandung mekanisme fungsi berpikir dan pola-pola penalaran system yang digunakan oleh seorang pakar. Mekanisme akan menganalisa suatu masalah tertentu dan selanjutnya akan mencari jawaban dan kesimpulan yang terbaik. Mesin inferensi memulai pelacakannya dengan mencocokan aturan-aturan dalam basis pengetahuan dengan fakta-fakta yang ada. Ada dua metode inferensi, yaitu :

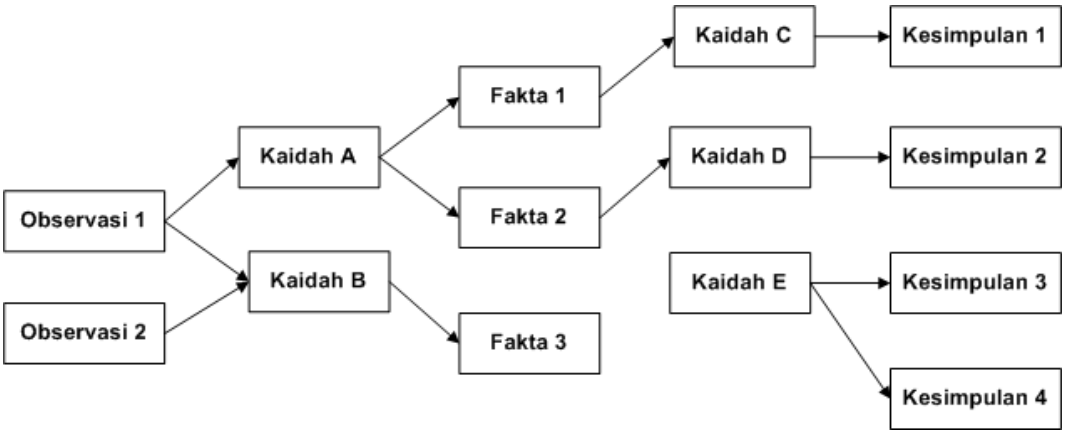
1. Pelacakan ke belakang (*backward chaining*)

   Penalaran dimulai dari tujuan (kesimpulan) menuju aturan-atuarn yang mendukung tujuan tersebut.



Gambar 2. 4 Backward Chaining

2. Pelacakan ke depan (*forward chaining*)



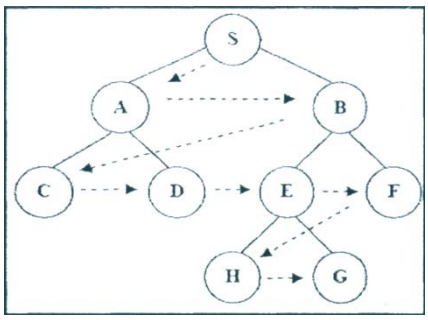
Gambar 2. 5 Forward Chaining

Penalaran dimulai dari sekumpulan data kemudian ditarik kesimpulannya.

Kedua metode inferensi tersebut dipengaruhi oleh tiga macam penulusuran,yaitu:

1. Pencarian Melebar Pertama (*Breadth-First Transversal*)

Pada metode Breadth-First Search ini adalah semua node pada level n akan dikunjungi terlebih dahulu sebelum mengunjungi node-node pada level n+1. pencarian dimulai dari node akar terus ke level 1 dari kiri ke kanan, kemudian berpindah ke level berikutnya dari kiri ke kanan hingga solusi ditemukan (Gambar 2.6 Metode Breadth-First Search)



Gambar 2. 6 Metode Breath-first Transversal

Keuntungan :

Tidak akan menemui jalan buntu

Jika ada 1 solusi, maka *breadth -first search* solusi akan menemukannya dan jika ada lebih dari satu solusi, maka solusi minimum akan ditemukan.

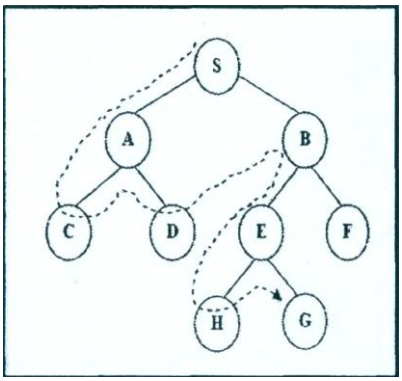
Kelemahan :

1. Membutuhkan memori yang cukup banyak
2. Membutuhkan waktu yang cukup lama, karena akan menguji n level untuk mendapatkan solusi pada level yang ke-(n+l).

2. Pencarian Mendalam Pertama *(Depth - First Transversal)*

Pada metode *Depth - First Transversal*, Proses pencarian dilakukan pada semua anaknya sebelum dilakukan pencarian ke node-node yang selevel.

Pencarian dimulai dari node akar ke level yang lebih tinggi. Proses diulangi terus hingga ditemukan solusi (Gambar 2.7 Metode *Depth - First Transversal).*



Gambar 2. 7 Metode Depth-First Transversal

Keuntungan :

1. Membutuhkan memori relatif kecil, karena hanya node-node pada lintasan yang aktif saja yang disimpan.
2. Secara kebetulan, metode *Depth — First Transversal* akan menemukan solusi tanpa harus menguji lebih banyak lagi dalam ruang keadaan.

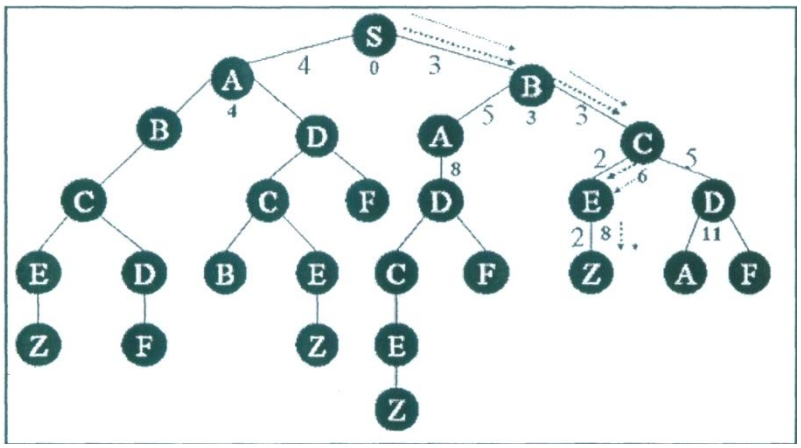
Kelemahan :

1. Memungkinkan tidak ditemukannya tujuan yang diharapkan
2. Hanya mendapat satu solusi pada setiap pencarian

3. Pencarian Terbaik Pertama *(Best First Transversal)*

Metode Pencarian Terbaik Pertama *(Best First Transversal)* merupakan kombinasi dari metode *depth first transversal* dan metode *breadth first transversal* dengan mengambil kelebihan dari kedua metode tersebut. Pada setiap langkah proses pencarian terbaik pertama, kita memilih node-node dengan menerapkan fungsi heuristik yang memadai pada setiap node atau simpul yang kita pilih dengan menggunakan aturan-aturan tertentu untuk menghasilkan

penggantinya. Pada *Best First Transversal*, pencarian diperbolehkan mengunjungi node di lebih rendah dan jika ternyata node dilevel lebih tinggi memiliki nilai heuristik lebih buruk.



Gambar 2. 8 Metode Best First Transversal

Keuntungan :

1. Membutuhkan memori yang relatif kecil, karena hanya node-node pada lintasan yang aktif saja yang disimpan.
2. Menemukan solusi tanpa hams menguji lebih banyak node.

Kelemahan :

1. Memungkinkan terjebak pada nilai optimal.

1.3.8 Representasi Pengetahuan

Pengetahuan dapat dipresentasikan dalam bentuk yang sederhana atau kompleks, tergantung dari masalahnya [8]. Ada beberapa model representasi yang penting, yaitu : Logika (logic), Pohon, Jaringan Semantik, Bingkai (frame), naskah (script), dan sistem produksi.

1.3.8.1 Logika

Logika merupakan suatu pengkajian ilmiah tentang serangkaian penalaran, sistem kaidah, dan prosedur yang membantu proses penalaran. Logika merupakan bentuk representasi pengetahuan yang paling tua, yang menjadi dasar dari teknik representasi *high level*.

1.3.8.2 Pohon

Pohon merupakan struktur penggambaran pohon secara hirarkis. Struktur pohon terdiri dari node-node yang menunjukan objek, dan *arc* (busur) yang nenunjukan hubungan antar objek.

1.3.8.3 Jaringan Semantik

Jaringan semantik merupakan suatu gambaran dari pengetahuan yang memperlihatkan hubungan hierarki dari objek-objek. Objek dipresentasikan dalam bentuk node dan hubungan antara objek dinyatakan oleh garis penghubung beratribut.

1.3.8.4 Bingkai (*Frame*)

Bingkai (*frame*) yaitu blok – blok berisi pengetahuan mengenai objek tertentu, kejadian, lokasi, situasi dari elemen-elemen lain yang menggambarkan objek tersebut secara rinci, dimana rincian objek tersebut disimpan ke dalam sebuah slot yang menggambarkan berbagai atribut dan karakteristik dari objek.

1.3.8.5 Naskah (*Script*)

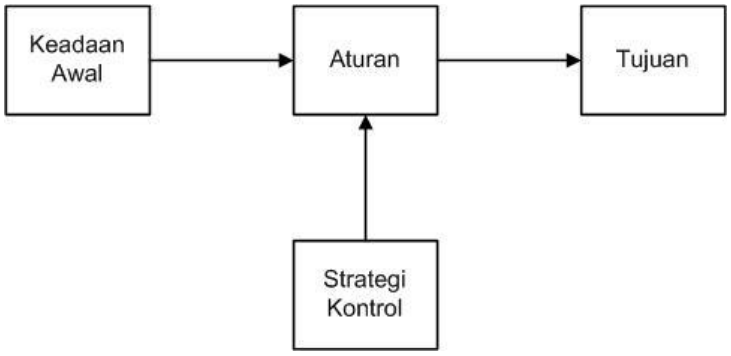
Script adalah skema representasi pengetahuan yang sama dengan frame, yaitu merepresentasikan pengetahuan berdasarkan karakteristik yang sudah

dikenal sebagai pengalaman-pengalaman. Perbedaannya, frame menggambarkan objek sedangkan *script* menggambarkan urutan peristiwa. Dalam menggambarkan urutan peristiwa, s*cript* menggunakan slot yang berisi informasi tentang orang, objek, dan tindakan-tindakan yang terjadi dalam suatu peristiwa.

1.3.8.6 Sistem Produksi

Sistem produksi secara umum terdiri dari komponen-komponen sebagai berikut(Gambar 2.9) :

1. Ruang keadaan, yang berisi keadaan awal, tujuan ,dan kumpulan aturan yang digunakan untuk mencapai tujuan.
2. Strategi kontrol, yang berguna untuk mengarahkan bagaimana proses pencarian akan berlangsung dan mengendalikan arah eksplorasi



Gambar 2. 9 Sistem Produksi

Sistem produksi ini merupakansalah satu bentuk representasi pengetahuan yang sangat populer dan banyak digunakan adalah sistem produksi. Representasi pengetahuan dengan sistem produksi, pada dasarnya berupa aplikasi aturan (*rule*) yang berupa :

1. *Antecedent*, yaitu bagian mengekspresikan situasi atau premis (pernyataan nerawalan IF).

2. Konsekuaen, yaitu bagian yang menyatakan suatu tindakan tertentu atau konklusi yang diterapkan jika suatu situasi atau premis bernilai benar (pernyataan berawalan THEN).

Konsekuansi atau konklusi yang dinyatakan pada bagian THEN baru dinyatakan benar, jika bagian IF pada sistem tersebut juga benar atau sesuai dengan aturan tertentu.

1.3.9 Basis Data

Basis data adalah kumpulan file- file yang mempunyai kaitan antara satu file dengan file lain sehingga membentuk satu bangunan data untuk menginformasikan satu perusahaan, instansi dalam batasan tertentu [5].

Basis data relasional adalah basis data yang mengorganisasikan data dalam bentuk tabel. Tabel dibentuk dengan mengelompokan data yang mempunyai subjek yang sama [5].

Basis data pada prinsipnya ditujukan untuk pengaturan data agar terdapat kemudahan dalam pengambilan kembali data tersebut. Berikut ini terdapat beberapa tujuan dari basis data diantaranya yaitu:

1. kecepatan dan kemudahan *(speed).*
2. efesiensi ruang penyimpanan *(space).*
3. keakuratan *(accuracy).*
4. ketersediaan *(avaibility).*
5. kelengkapan *(completeness).*
6. keamanan *(security).*
7. kebersamaan *(sharability).*

Kumpulan file yang saling berkaitan bersama dengan program untuk pengelolaannya disebut sebagai DBMS *(Database Management System). Database* adalah kumpulan datanya, sedangkan program pengelolaannya berdiri sendiri dalam satu paket program yang komersial untuk membaca data, mengisi data, menghapus data, melaporkan data dalam *database*.

Beberapa definisi dalam perencanaan *database* yang perlu untuk diketahui adalah sebagai berikut:

a. *Entity* : Orang, tempat kejadian, kosep yang informasinya direkam.

b. *Attribute* : Setiap *entity* mempunyai atribut untuk mewakili satu *entity*/ atribut disebut juga sebagai data elemen, data field atau data item.

c. *Data Value* : Data aktual atau informasi yang disimpan pada tiap data elemen.

d. *Record/tufle*: Kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang *entity* secara lengkap.

e. *File* : Kumpulan *record-record* sejenis yang mempunyai panjang elemen yang sama, atribut yang sama namun berbeda *data value*nya.

1.3.10 SQL (Structured Query Language)

SQL (Structured Query Language) merupakan standar bahasa yang digunakan untuk memanipulasi/*Data Manipulation Language (DML)* dan mendefinisikan/*Data Definition Language (DDL)* objek-objek basis data sesuai dengan permintaan *(request)* dari pengguna sistem [5]. Dimana *DML (Data Manipulatoin Language)* adalah sekumpulan perintah-perintah *query* yang digunakan untuk melakukan manipulasi data seperti penyimpanan data ke suatu

tabel, lalu kemudian mengubahnya atau menghapusnya atau hanya sekedar menampilkan saja, sedang *DDL (Data Definition Language)* merujuk pada kumpulan perintah *query* yang digunakan untuk mendefinisikan objek-objek basis data seperti pembuatan tabel basis data atau indeks primer/sekunder dan lain-lain.

1.3.11 Arsitektur Sistem Basis Data

Arsitektur sistem merujuk pada konfigurasi sistem secara keseluruhan yang akan menjadi tempat hidup dari DBMS basis data dan aplikasi yang memanfaatkannya [5]. Bagaimana tempat wujud dari tempat hidup tersebut yang juga akan menentukan bagaimana para pemakai sistem berinteraksi dengannya, sudah seharusnya ditetapkan sejak memulai perancangan basis data atau paling tidak sebelum melaksanakan tahap implementasi basis data.

Dari waktu ke waktu, sejalan dengan kemajuan teknologi, arsitektur yang dapat kita pilih semakin beragam atau semakin banyak variannya, yang berubah juga keunggulannya pada tiap jenis arsitektur tersebut. Arsitektur saat ini yang dikatakan paling baik dan banyak ditempatkan di berbagai tempat dan di masa yang akan datang dapat beralih ke arsitektur lain yang memang sama sekali baru atau merupakan pengembangan dari arsitektur yang lama, yang menjadi utama adalah menentukan arsitektur yang cocok yang tentu saja bukan hanya keunggulan teknologinya tetapi disamping itu biaya yang sesuai dengan anggaran organisasi yang akan mengaplikasikannya.

1.3.11.1 Sistem Tunggal

Pada arsitektur ini *Database Management System (DBMS),* basis data dan aplikasi basis data ditempatkan pada mesin (komputer) yang sama. Dengan

demikian, pemakai yang dapat menggunakannya setiap saat juga hanya satu orang *(single user).*

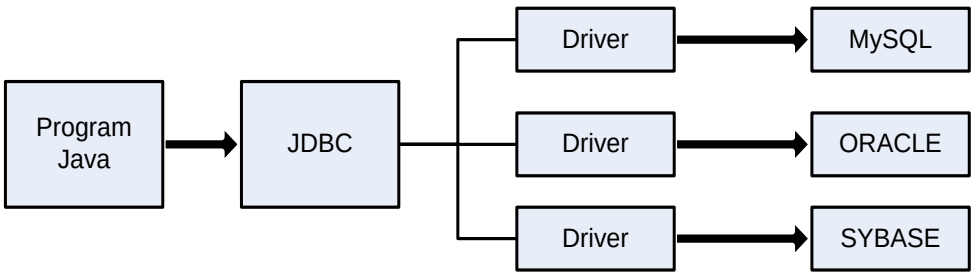
1.3.11.2 Sistem Tersentralisasi

Arsitektur ini terdiri dari sebuah mesin *server* dan sejumlah terminal (yang menjadi tempat *user* berinteraksi dengan sistem), yang tersentralisasi dalam arsitektur ini dapat mencakup basis data, DBMS, dan aplikasi basis data atau basis data saja.

1.3.12 Java Database Connectivity (JDBC)

Java Database Connectivity (JDBC) adalah *Application Programming Interface (API)* yang dikembangkan oleh *Sun Microsystem* untuk menyediakan akses data universal dalam bahasa pemrograman JAVA. JDBC merupakan bagian dari Java 2 (dua) SDK *Standard Edition*. Inti JDBC adalah *package.sql* dan *javax.sql* (bagian dari J2EE). JDBC menyediakan mekanisme untuk koneksi *database*, sintaks untuk mengirimkan *query* dan melakukan transaksi serta struktur data yang mewakili hasil *query*.

Diagram JDBC seperti terlihat pada Gambar 2.10 di bawah ini.



Gambar 2. 10 Diagram JDBC

JDBC merupakan koleksi dari API yang terdiri dari sekumpulan *class* dan *interface* yang ditulis dalam pemrograman Java sebagai *standart API* untuk

developer database maupun *software database* sehingga memungkinkan pembuatan aplikasi *database* yang *portable* dengan bahasa Java [7].

Dengan JDBC kita dapat membuat program aplikasi *database* yang dapat berganti *server* tanpa mengubah banyak kode dari program yang sudah ada sehingga memberikan portabilitas tinggi bagi aplikasi yang telah dibuat.

Dalam JDBC terdapat tujuh langkah standar untuk menangani *database*:

1) Memanggil/*load driver JDBC*.
2) Mendefinisikan koneksi *database*.
3) Melakukan koneksi *database*.
4) Membuat objek *statement*.
5) Melakukan *query dan update*.
6) Memproses hasil.
7) Menutup koneksi.

1.3.13 Sistem Database Oracle

Sistem Database Oracle adalah basis data relasional yang terdiri dari kumpulan data dalam suatu sistem manajemen basis data RDBMS. Perusahaan perangkat lunak Oracle memasarkan jenis basis data ini untuk bermacam-macam aplikasi yang bisa berjalan pada banyak jenis dan merk perangkat keras komputer (platform).

Basis data Oracle ini pertama kali dikembangkan oleh Larry Ollison, Bob Miner dan Ed Oates lewat perusahaan konsultasinya bernama Software Development Laboratories (SDL) pada tahun 1977. Pada tahun 1983, perusahaan ini berubah nama menjadi Oracle Corporation sampai sekarang.

Oracle merupakan DBMS yang paling rumit dan paling mahal di dunia, namun banyak orang memiliki kesan yang negatif terhadap Oracle. Keluhan-keluhan yang mereka lontarkan mengenai Oracle antara lain adalah terlalu sulit untuk digunakan, terlalu lambat, terlalu mahal, dan bahkan Oracle dijuluki dengan istilah "ora kelar-kelar" yang berarti "tidak selesai-selesai" dalam bahasa Jawa. Jika dibandingkan dengan MySQL yang bersifat gratis, maka Oracle lebih terlihat tidak kompetitif karena berjalan lebih lambat daripada MySQL meskipun harganya sangat mahal.

Namun yang mereka tidak perhitungkan adalah bahwa Oracle merupakan DBMS yang dirancang khusus untuk perusahaan yang berukuran besar, bukan untuk ukuran kecil dan menengah. Kebutuhan perusahaan berukuran besar tidaklah sama dengan perusahaan yang kecil atau menengah yang tidak akan berkembang menjadi besar. Perusahaan seperti perhotelan yang berukuran besar membutuhkan fleksibilitas dan skalabilitas agar dapat memenuhi tuntutan akan data dan informasi yang bervolume besar dan terus menerus bertambah besar. Kapasitas penyimpanan data dalam Oracle mencapai TeraByte.

Kelebihan lain dari Oracle yaitu hanya akan memanfaatkan resource yang default atau diberikan saja. Sehingga ketika sudah running di server, kadang perlu ditunning agar kecepatan dan kerjanya bisa maksimal. Tunning dapat dilakukan kapan saja dan berkali-kali. Ini sangat menguntungkan karena pemakaian resource dapat dikontrol dengan mudah. Tetapi efek buruknya adalah jika salah dalam melakukan tunning atau tidak ditunning sama sekali, oracle akan bekerja sangat lambat.

1.3.14 UML

UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan *software* berbasis *object-oriented* [9]. UML sendiri juga memberikan standar penulisan sebuah sistem *blue print*, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem *software*.

Untuk dapat memahami UML membutuhkan bentuk konsep dari sebuah bahasa model, dan mempelajari 3 (tiga) elemen utama dari UML seperti *building block*, aturan-aturan yang menyatakan bagaimana *building block* diletakkan secara bersamaan, dan beberapa mekanisme umum (*common*). Tiga macam yang terdapat dalam *building block* adalah katagori benda/*things*, hubungan, dan diagram. Benda/*things* adalah abstraksi yang pertama dalam sebuah model, hubungan sebagai alat komunikasi dari benda-benda, dan diagram sebagai kumpulan / group dari benda-benda/*things.*

1) Benda/*Things*

   Benda adalah hal yang sangat mendasar dalam model UML, juga merupakan bagian paling statik dari sebuah model, serta menjelaskan elemen-elemen lainnya dari sebuah konsep dan atau fisik. Bentuk dari beberapa benda/*thing*s adalah sebagai berikut:

   Pertama, adalah sebuah kelas yang diuraikan sebagai sekelompok dari *object* yang mempunyai atribut, operasi, hubungan yang semantik. Sebuah

kelas mengimplementasikan satu atau lebih *interfaces*. Sebuah kelas dapat digambarkan sebagai sebuah persegi panjang, yang mempunyai sebuah nama, atribut, dan metoda pengoperasiannya.

Kedua, yang menggambarkan *'interface'* merupakan sebuah antar-muka yang menghubungkan dan melayani antar kelas dan atau elemen. *'Interface'*/antar-muka mendefinisikan sebuah set/kelompok dari spesifikasi pengoperasian, umumnya digambarkan dengan sebuah lingkaran yang disertai dengan namanya. Sebuah antar-muka berdiri sendiri dan umumnya merupakan pelengkap dari kelas atau komponen.

Ketiga, adalah *collaboration* yang didefinisikan dengan interaksi dan sebuah kumpulan/kelompok dari kelas-kelas/elemen-elemen yang bekerja secara bersama-sama. C*ollaborations* mempunyai struktur dan dimensi. Pemberian sebuah kelas memungkinkan berpartisipasi didalam beberapa *collaborations* dan digambarkan dengan sebuah '*elips*' dengan garis terpotong-potong.

Keempat, sebuah '*use case*' adalah rangkaian/uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. '*use case*' digunakan untuk membentuk tingkah-laku benda/*things* dalam sebuah model serta direalisasikan oleh sebuah collaboration. Umumnya *'use case'* digambarkan dengan sebuah *'elips'* dengan garis yang solid, biasanya mengandung nama.

Kelima, sebuah node merupakan fisik dari elemen-elemen yang ada pada saat dijalankannya sebuah sistem, contohnya adalah sebuah komputer,

umumnya mempunyai sedikitnya *memory* dan *processor*. Sekelompok komponen mungkin terletak pada sebuah *node* dan juga mungkin akan berpindah dari node satu ke node lainnya. Umumnya node ini digambarkan seperti kubus serta hanya mengandung namanya.

2) Hubungan / *Relationship*

Relationship adalah koneksi antar model elemen. Interaksi antara actor dan use-case dalam use-case model biasanya digunakan association relationship yaitu :

a. Uses

Hubungan uses menunjukkan bahwa prosedur dari use-case merupakan bagian dari prosedur yang menggunakan use-case. Tanda panah menunjukkan keadaan tidak mengakibatkan pemanggilan prosedur dalam menggunakan use-case. Relasi uses antara use-case ditunjukkan dengan panah generalisasi dari use-case. Use-case yang dilakukan secara berulang, digunakan untuk meminimalkan pekerjaan.

b. Include

Hubungan include menggambarkan suatu use-case seluruhnya meliputi kegunaan dari use-case lainnya. Sebuah use-case dapat meng-include fungsionalitas use-case lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa use-case yang di-include dieksekusi secara normal. Sebuah use-case dapat di-include oleh lebih dari use-case lain, sehingga duplikasi fungsional dapat dihindari.

c. Extend

Hubungan extend antar use-case berarti bahwa suatu use-case merupakan tambahan kegunaan dari use-case yang lain jika kondisi atau syarat tertentu dipenuhi. Jika prosedur dari use-case merupakan alternatif untuk menjelaskan use-case lain. Use-case akan dikerjakan apabila salah satu syarat terpenuhi.Hubungan generalisasi antar use-case menunjukkan bahwa use-case yang satu merupakan spesialisai dari yang lain.

3) Diagram

UML terdiri atas pengelompokkan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model.

Diagram-diagram dari UML adalah *use-case, class, object, state, sequence, collaboration, activity, component,* dan *deployment diagram.*

a. Use Case Diagram

Use case adalah suatu diagram yang membantu pengembangan sistem bekerja dengan user untuk menentukan kegunaan sistem. Koleksi dari *use case* melukiskan apa yang diinginkan *user* terhadap sebuah sistem. *Use case* bertujuan untuk menentukan bagaimana *actors* berinteraksi dengan sebuah sistem.

b. Activity Diagram

Activity diagram adalah bagian dari UML yang digunakan untuk menggambarkan tahapan dari setiap proses bisnis yang ada agar lebih mudah memahami proses bisinis yang terjadi.

Dalam *activity diagram* tiap aktivitas direpresentasikan dengan *rounded rectangle* yang dihubungkan dengan anak panah untuk menggambarkan transisi dari satu aktivitas ke aktivitas lain. *Activity diagram* dimulai dari *initial state* dan diakhiri dengan *final state*.

c. Class Diagram

Class diagram adalah bagian dari UML yang menggambarkan sebuah kumpulan dari kelas-kelas yang ada dan hubungan diantara kelas tersebut dimana setiap kelas mempunyai *attributes* dan *operations*.

Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). Selain itu, c*lass diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

d. Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang

men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*.

1.4 Pemrograman Berorientasi Objek

Pada subbab ini akan diuraikan teori yang berkaitan dengan Pemrograman Berorientasi Objek (*Object Oriented Programing)*, mulai dari pengertian itu sendiri hingga sejarahnya

1.4.1 Definisi Pemrograman Berorientasi Objek

OOP/PBO merupakan paradigma pemrograman yang popular saat ini yang telah menggantikan teknik pemrograman berbasis prosedur. Object Oriented Programing yang berarti pula Pemrograman Beorientasi Objek sudah ditemukan sekitar tahun 1960 dan dikembangkan pada permulaan tahun 1970.

Pemrograman Berorientasi Objek (Object Oriented Programming/OOP) merupakan pemrograman yang berorientasikan kepada objek, dimana semua data dan fungsi dibungkus dalam class-class atau object-object.Setiap object dapat menerima pesan, memproses data, mengirim, menyimpan dan memanipulasi data[7]. Beberapa object berinteraksi dengan saling memberikan informasi satu terhadap yang lainnya.

Masing-masing object harus berisikan informasi mengenai dirinya sendiri dan dapat dihubungkan dengan Object yang lain. Pemrograman berorientasi objek berbeda dengan pemrograman prosedural yang hanya menggunakan satu halaman kebawah untuk mengerjakan banyak perintah atau statement. Penggunaan pemrograman berorientasi objek sangat benyak sekali, contoh : java, php, perl, c#, cobol, dan lainnya.

Dalam konsep Pemrograman Berorientasi Objek dikenal beberapa istilah umum, yaitu: atrribute, method, class, object,

1.4.2 Attribute

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas, Atribut dapat memiliki hak akses private, public maupun protected [7].

Sebuah atribut yang dinyatakan sebagai private hanya dapat diakses secara langsung oleh kelas yang membungkusnya, sedangkan kelas lainnya tidak dapat mengakses atribut ini secara langsung. Sebuah atribut yang dinyatakan sebagai public dapat diakses secara langsung oleh kelas lain di luar kelas yang membungkusnya.

Sebuah atribut yang dinyatakan sebagai protected tidak dapat diakses secara langsung oleh kelas lain di luar kelas yang membungkusnya, kecuali kelas yang mengaksesnya adalah kelas turunan dari kelas yang membungkusnya.

Atribut juga biasa disebut state/ properties/ field merupakan bagian yang dimiliki oleh sebuah class yang tidak melakukan operasi, tetapi kepadanya dilakukan operasi sehingga dapat merubah nilai dari attribute tersebut

1.4.2.1 Karakteristik Atribut

Dalam class, atribut disebut sebagai variabel. Atribut dapat membedakan antara satu object dengan object yang lain. Sebagai contoh, pada class : mahasiswa, terdapat object mahasiswa si A, dan object mahasiswa si B.

Yang membedakan antara object si A dan si B adalah NPM-nya (Nomor Pokok Mahasiswa) yang merupakan atribut dari object tersebut. Pada atribut, terdapat pula dua istilah variabel, yaitu Instance Variabel dan Class Variable.

Dimana pada tiap variabel memiliki karakteristik tersendiri, diantaranya : Pada Instance Variabel à tiap object memiliki instance variabel dan menyimpan nilainya tersendiri. Sedangkan pada Class Variabel à atribut yang dimiliki oleh semua object yang berasal dari class yang sama, serta semua object memiliki nilai class variabel yang sama. atribut yang dimiliki object dari class yang sama.

1.4.3 Method

Method adalah fungsi atau prosedur yang dibuat oleh seorang programmer didalam suatu Class. Dengan kata lain, method pada sebuah kelas hampir sama dengan fungsi atau prosedur pada pemrograman prosedural [7].

Pada sebuah method di dalam sebuah kelas juga memiliki izin akses seperti halnya atribut pada kelas, izin akses itu antara lain private, public dan protected yang memiliki arti sama pada izin akses atribut yang telah dibahas sebelumnya. Sebuah kelas boleh memiliki lebih dari satu method dengan nama yang sama asalkan memiliki parameter masukan yang berbeda sehingga kompiler atau interpreter dapat mengenali method mana yang dipanggil.

Hal ini dinamakan overloading. Di dalam sebuah kelas, terdapat juga yang disebut sebagai method atau atribut statis yang memiliki kata kunci static. Maksud dari statis di sini adalah method yang dapat diakses secara berbagi dengan semua objek lain tanpa harus membuat objek yang memiliki method statis tadi (tanpa proses new), tapi sebuah method statis mempunyai keterbatasan yaitu hanya dapat mengakses atribut atau method lain di dalam kelas yang membungkusnya yang juga bersifat statis. Method statis biasanya diimplementasikan untuk method main.

1.4.3.1 Karakteristik Method

Didalam suatu class, behavior disimpan dalam bentuk Method. Method merupakan serangkaian statemen / perintah (perintah = baris program) dalam suatu class yang menghandle task tertentu. Method merupakan hal-hal yang bisa dilakukan oleh object dari suatu class.

Method didefinisikan pada class akan tetapi dipanggil melalui object. Method memiliki peranan diantaranya :

- merubah nilai atribut dari suatu object,
- menerima informasi dari object lain,
- mengirim informasi ke obyek lain untuk melakukan suatu task (cara obyek berkomunikasi dengan obyek lain adalah dengan menggunakan method).

1.4.4 Class

Merupakan model yang berisi kumpulan attribute dan method dalam suatu unit untuk suatu tujuan tertentu [7]. Sebagai contoh class manusia memiliki attribute berat, tinggi, usia kemudian memiliki method makan, minum, tidur.

Method dalam sebuah class dapat merubah attribute yang dimiliki oleh class tersebut. Sebuah class merupakan dasar dari modularitas dan struktur dalam pemrograman berorientasi object.

Class didefinisikan sebagai sebuah blueprint(denah), atau prototipe, yang mendefiniskan variabel-variabel dan metode-metode yang umum untuk semua objek dari n jenis tertentu (n maksudnya jumlah tertentu). Sebuah kelas menyerupai sebuah struktur yang merupakan tipe data sendiri, misalkan tipe data titik yang terdiri dari koordinat x dan y. Bahasa Java telah menggunakan sebuah kelas untuk menyatakan tipe data titik karena bahasa pemrograman Java merupakan bahasa pemrograman berorientasi objek murni sehingga tidak mengenal struktur, melainkan mengenal apa yang disebut dengan kelas.

Perbedaan sebuah kelas dengan sebuah struktur adalah sebuah kelas dapat berdiri sendiri dan dapat digunakan untuk berbagai keperluan kelas-kelas yang lain, sedangkan sebuah struktur tidak dapat berdiri sendiri. Sebuah kelas lebih fleksibel untuk digunakan oleh kelas lain tanpa harus membongkar kode program utama, sedangkan jika digunakan struktur maka kode program harus dibongkar unuk disalin bagian strukturnya ke kode program utama yang lain.

Sebuah file dapat terdiri dari berbagai kelas, namun biasanya pada bahasa pemrograman Java sebuah file hanya terdiri dari satu kelas yang disimpan dengan nama kelas, misal file List.java berisi kelas List. Namun jika kelas yang dibuat misalnya public class nama_kelas, maka kelas itu harus disimpan dalam satu file hanya untuk satu kelas. Setelah dilakukan kompilasi maka pada Java akan ada sebuah file .class yang berisi bytecode dari setiap kelas.

Jika sebuah file terdiri dari dua kelas, maka setelah dilakukan kompilasi akan dihasilkan dua buah file .class yang nantinya akan dibaca oleh interpreter Java saat program dieksekusi. Sebuah kelas saat program dieksekusi dan perintah new dijalankan, maka akan dibuat sebuah objek. Class adalah template untuk pembuatan objek. Class juga memiliki anggota, diantaranya : atribut dan method.

Pemahaman singkatnya :

- Class merupakan pola / template yang menggambarkan kumpulan object yang mempunyai sifat dan perilaku yang sama.
- Class merupakan struktur data dari object (blueprint dari sebuah object), dan proses pembuatan object dari sebuah class disebut juga dengan instantiation.
- Class merupakan definisi data dan fungsi dari suatu hal
- Di dalam class dideklarasikan variabel dan method yang dimiliki oleh obyek

1.4.4.1 Karakteristik Class

Class memiliki beberapa karakteristik, diantaranya : anggota class terdiri dari atribut dan method. Tiap-tiap anggota class memiliki control pengaksesan tersendiri, maksudnya adalah apakah anggota class tersebut dapat diakses dengan bebas (dengan tipe public) atau hanya dapat diakses melalui sebuah interface.

Dalam hal ini, interface adalah device yang digunakan untuk komunikasi antar object berbeda yang tidak memiliki hubungan apapun. Interface bisa dikatakan sebagai protokol komunikasi antar object tersebut.

1.4.5 Object

Merupakan perwujudan dari class, setiap object akan mempunyai attribute dan method yang dimiliki oleh class-nya, contohnya: amir, ahmad, yani merupakan object dari class manusia. Setiap object dapat berinteraksi dengan object lainnya meskipun berasal dari class yang berbeda**.**

Object merupakan sesuatu yang memiliki identitas (nama), pada umumnya juga memiliki data tentang dirinya maupun object lain dan mempunyai kemampuan untuk melakukan sesuatu dan bisa bekerja sama dengan objek lainnya [7]. Pada dasarnya semua benda yang ada di dunia dapat dianggap sebagai sebuah object. Rumah, mobil, sepeda motor, meja, dan komputer merupakan contoh-contoh object yang ada di dunia nyata. Object adalah implementasi dari class. Secara sederhananya, dapat dikatakan terdiri dari properti (atribut) dan method.

Pemahaman singkatnya :

• Objek merupakan sebuah objek khusus yang digunakan untuk mengakses anggota objek (atribut atau method) dari dalam objek itu sendiri.

• Apabila terdapat variabel lain yang memiliki nama yang sama dengan nama atribut di dalam objek, “this” dapat digunakan untuk merujuk ke atribut objek.

1.4.5.1 Karakteristik Object

Setiap object memiliki dua karakteristik yang utama, yaitu atribut dan behavior. Atribut merupakan status object dan behavior merupakan tingkah laku dari object tersebut. Contoh sederhana adalah sepeda. Object sepeda memiliki atribut : pedal, roda, dan gigi, kecepatan serta memiliki tingkah laku seperti pacu

dan rem. Sama seperti penggambaran di dunia nyata, pengembangan program dengan konsep OOP juga menggunakan model object yang akan menyimpan atributnya dalam variabel dan tingkah lakunya dalam method atau fungsi-fungsi/prosedur. Deskripsinya :

Contoh objek : Sepeda

- Sepeda memiliki atribut (state) : pedal, roda, jeruji, dan warna.
- Sepeda memiliki tingkah laku (behaviour) : kecepatannya menaik, kecepatannya menurun, dan perpindahan gigi sepeda.

Dalam pengembangan perangkat lunak berorientasi objek, objek dalam perangkat lunak akan menyimpan state-nya dalam variabel dan menyimpan informasi tingkah laku (behaviour) dalam method-method atau fungsi-fungsi/prosedur.

1.4.6 Konsep-konsep dalam Pemrograman Berorientasi Objek

1. **Abstraction**

Adalah suatu cara untuk melihat suatu object dalam bentuk yang lebih sederhana. Sebagai contoh kita tidak melihat sebuah mobil sebagai ribuan komponen elektronik, sistem mekanik dengan empat buah ban, jok, stir kemudi dan sebagainya. Dengan Abstraction, suatu sistem yang kompleks dapat dipandang sebagai kumpulan subsistem-subsistem yang lebih sederhana, seperti halnya mobil merupakan suatu sistem yang terdiri atas berbagai subsistem, seperti subsistem kemudi, subsistem pengereman dan sebagainya.

Abstraksi merupakan kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti.

Setiap objek dalam sistem melayani sebagai model dari “pelaku” abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

2. Encapsulation

Merupakan suatu mekanisme untuk menyembunyikan atau memproteksi suatu proses dari kemungkinan interferensi atau penyalahgunaan dari luar sistem dan sekaligus menyederhanakan penggunaan sistem tersebut.

Memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak. Hanya metode dalam objek tersebut yang diberi ijin untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

3. Inheritance

Merupakan konsep mewariskan attribute dan method yang dimiliki oleh sebuah class kepada class turunannya. Dengan konsep ini class yang dibuat cukup mendefinisikan attribute dan method yang spesifik didalamnya, sedangkan attribute dan method yang lebih umum akan didapatkan dari class yang menjadi induknya.

Mengatur polimorfisme dan enkapsulasi dengan mengijinkan objek didefinisikan dan diciptakan dengan jenis khusus dari objek yang sudah

ada.Objek-objek ini dapat membagi (dan memperluas) perilaku mereka tanpa haru mengimplementasi ulang perilaku tersebut (bahasa berbasis-objek tidak selalu memiliki inheritas.)

4. Polymorphism

Merupakan konsep yang memungkinkan digunakannya suatu interface yang sama untuk memerintah suatu object agar melakukan suatu tindakan yang mungkin secara prinsip sama tetapi secara proses berbeda.

Model data berorientasi objek disebut dapat memberi fleksibilitas yang lebih, kemudahan mengubah program dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya selain itu pendekatan OOP lebih mudah dikembangkan dan dirawat.

Polimorfisme melalui pengiriman pesan. Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan, dengan metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesan tersebut dikirim.

Contohnya, bila sebuah burung menerima pesan “gerak cepat”, dia akan menggerakan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tungal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang berbeda di saat yang

berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan bahasa fungsional yang mencapai polimorfisme melalui penggunaan fungsi kelas-pertama.

1.5 Jaringan Komputer

Pada subbab ini akan diuraikan teori yang berkaitan dengan jaringan komputer

1.5.1 Definisi Jaringan Komputer

Jaringan komputer adalah sebuah sistem yang terdiri atas komputer, software dan perangkat jaringan lainnya yang bekerja bersama-sama untuk mencapai suatu tujuan yang sama [11]. Tujuan dari jaringan komputer adalah:

- Membagi sumber daya: contohnya berbagi pemakaian printer, CPU,
- Komunikasi: contohnya surat elektronik, instant messaging, chatting
- Akses informasi: contohnya web browsing

Agar dapat mencapai tujuan yang sama, setiap bagian dari jaringan komputer meminta dan memberikan layanan (service). Pihak yang meminta/menerima layanan disebut klien (client) dan yang memberikan/mengirim layanan disebut pelayan (server). Arsitektur ini disebut dengan sistem client-server, dan digunakan pada hampir seluruh aplikasi jaringan komputer.

Berdasarkan skala :

1. Local Area Network (LAN): suatu jaringan komputer yang menghubungkan suatu komputer dengan komputer lain dengan jarak yang terbatas.

2. Metropolitant Area Network (MAN): prinsip sama dengan LAN, hanya saja jaraknya lebih luas, yaitu 10-50 km.
3. Wide Area Network (WAN): jaraknya antar kota, negara, dan benua. ini sama dengan internet.

Berdasarkan fungsi : Pada dasarnya setiap jaringan komputer ada yang berfungsi sebagai client dan juga server. Tetapi ada jaringan yang memiliki komputer yang khusus didedikasikan sebagai server sedangkan yang lain sebagai client. Ada juga yang tidak memiliki komputer yang khusus berfungsi sebagai server saja. Karena itu berdasarkan fungsinya maka ada dua jenis jaringan komputer:

a. Client-server

   Yaitu jaringan komputer dengan komputer yang didedikasikan khusus sebagai server. Sebuah service/layanan bisa diberikan oleh sebuah komputer atau lebih. Contohnya adalah sebuah domain seperti www.detik.com yang dilayani oleh banyak komputer web server. Atau bisa juga banyak service/layanan yang diberikan oleh satu komputer.

b. Peer-to-peer

   Yaitu jaringan komputer dimana setiap host dapat menjadi server dan juga menjadi client secara bersamaan. Contohnya dalam file sharing antar komputer di Jaringan Windows Network Neighbourhood ada 5 komputer (kita beri nama A,B,C,D dan E) yang memberi hak akses terhadap file yang dimilikinya. Pada satu saat A mengakses file share dari B bernama data_nilai.xls dan juga memberi akses file soal_uas.doc kepada C. Saat A

mengakses file dari B maka A berfungsi sebagai client dan saat A memberi akses file kepada C maka A berfungsi sebagai server. Kedua fungsi itu dilakukan oleh A secara bersamaan maka jaringan seperti ini dinamakan peer to peer

1.6 Topologi Jaringan

Pada subbab ini akan diuraikan teori yang berkaitan dengan topologi jaringan.

1.6.1 Definisi Topologi Jaringan

Topologi pada dasarnya adalah peta dari sebuah jaringan. Topologi jaringan terbagi lagi menjadi dua, yaitu topologi secara fisik (physical topology) dan topologi secara logika (logical topology). Topologi secara fisik menjelaskan bagaimana susunan dari kabel dan komputer dan lokasi dari semua komponen jaringan. Sedangkan topologi secara logika menetapkan bagaimana informasi atau aliran data dalam jaringan.

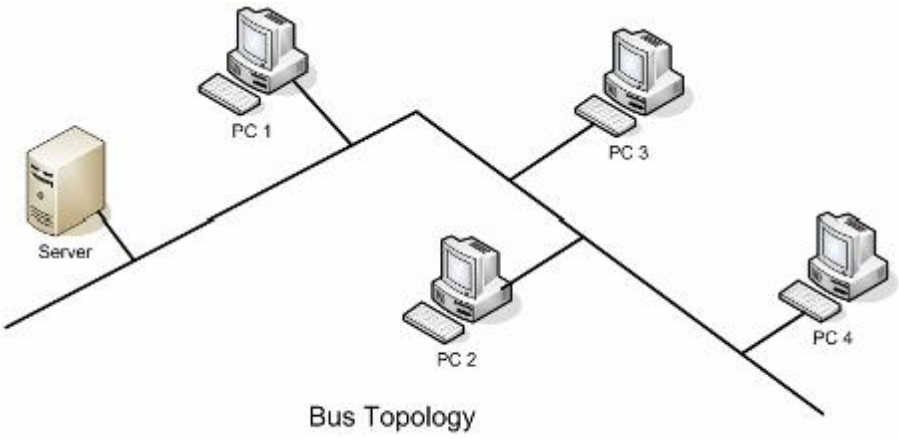
Kabel atau koneksi dalam physical topology seringkali mengenai media jaringan (atau media fisik). Memilih bagaimana komputer-komputer akan dihubungkan dalam suatu jaringan sangat penting (terlebih lagi dalam jaringan perusahaan). Pemilihan topologi yang salah akan membuat sangat sulit untuk membenarkannya, karena hal tersebut tentu saja merugikan. Sangat penting untuk memilih bagaimana topologi yang tepat untuk jaringan yang akan digunakan. Biasanya suatu organisasi atau perusahaan merubah susunan fisik dan media fisik jaringannya sekali dalam sepuluh tahun. Jadi sangat penting untuk memilih konfigurasi yang tepat [11].

Topologi yang paling umum digunakan:

- Bus
- Star (Bintang)
- Ring
- Mesh

1.6.2 Topologi Bus

Topologi bus seringkali digunakan ketika jaringannya berukuran kecil, simpel, atau bersifat sementara. Sangat sederhana dalam instalasi, dan ekonomis dalam hal biaya.



Gambar 2. 11 Topologi Bus

1.6.2.1 Bagaimana Jaringan Bus Bekerja

Tipikal dari jaringan bus, kabel hanya satu atau lebih wires, tanpa adanya alat tambahan yang menguatkan sinyal atau melewatkannya terus dari komputer ke komputer. Topologi bus merupakan topologi yang pasif. Ketika satu komputer mengirim sinyal up (dan down), semua komputer dalam jaringan menerima informasi, tetapi hanya satu komputer yang menyetujui informasi

tersebut (komputer yang memiliki alamat yang sama dengan alamat yang menjadi tujuan dalam pesan). Sedangkan komputer yang lainnya akan menghiraukan pesan tersebut.

Topologi dari jaringan bus menggunakan broadcast channel yang berarti setiap komputer atau peralatan yang terhubung dapat mendengar setiap pengiriman dan semuanya memiliki prioritas yang sama dalam menggunakan jaringan untuk mengirimkan data.

Keuntungan dari penggunaan Topologi Bus

Terdapat beberapa keuntungan dari penggunaan topologi bus:

- Bus adalah topologi yang sederhana, dapat diandalkan untuk penggunaan pada jaringan yang kecil, mudah untuk digunakan, dan mudah untuk dimengerti.
- Bus hanya membutuhkan kabel dalam jumlah yang sedikit untuk menghubungkan komputer-komputer atau peralatan-peralatan yang lain dan oleh karena itu biayanya lebih murah dibandingkan dengan susunan pengkabelan yang lain.
- Cukup mudah apabila kita ingin memperluas jaringan pada topologi bus. Dua kabel dapat digabungkan pada kabel yang lebih panjang dengan menggunakan BNC barrel connector, membuat kabel menjadi lebih panjang dan membolehkan komputer-komputer lain untuk untuk dihubungkan ke dalam jaringan.
- Sebuah repeater dapat digunakan untuk memperluas jaringan, repeater digunakan untuk menguatkan sinyal sehingga dapat menempuh jarak yang

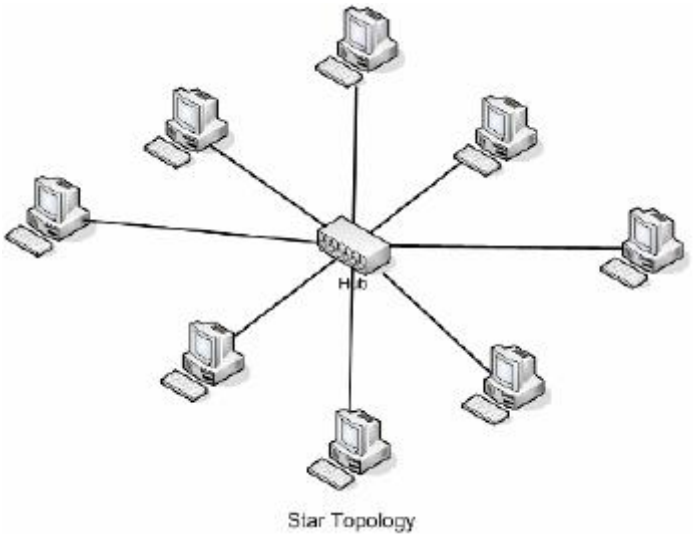
lebih jauh.

1.6.2.2 Kerugian dari penggunaan Topologi Bus

- Traffic (lalu lintas) yang padat akan sangat memperlambat bus. Karena setiap komputer dapat mengirim setiap waktu dan komputer-komputer yang ada pada jaringan bus tidak saling berkoordinasi satu sama lain dalam menyediakan waktu untuk mengirim.
- Dalam jaringan bus sejumlah komputer akan menghabiskan sejumlah bandwidth (kapasitas untuk mengirimkan informasi) dengan komputer-komputer yang saling mengganggu satu sama lain daripada berkomunikasi.
- Setiap barrel connector yang digunakan sebagai penghubung memperlemah sinyal elektrik yang dikirimkan, dan kebanyakan akan menghalangi sinyal untuk dapat diterima dengan benar.
- Sangat sulit untuk melakukan troubleshoot pada bus. Apabila ada kabel yang putus atau komputer yang tidak berfungsi dimanapun antara dua komputer akan menyebabkan komputer-komputer tersebut tidak dapat berkomunikasi satu sama lain.
- Lebih lambat dibandingkan dengan topologi yang lain.

1.6.3 Topologi Bintang (Star)

Dalam topologi star, semua kabel dihubungkan dari komputer-komputer ke lokasi pusat (central location), dimana semuanya terhubung ke suatu alat yang dinamakan hub.



Gambar 2. 12 Topologi Star

Topologi star digunakan dalam jaringan yang padat, ketika endpoint dapat dicapai langsung dari lokasi pusat, kebutuhan untuk perluasan jaringan, dan membutuhkan kehandalan yang tinggi. Topologi ini merupakan susunan yang menggunakan lebih banyak kabel daripada bus dan karena semua komputer dan perangkat terhubung ke central point. Jadi bila ada salah satu komputer atau perangkat yang mengalami kerusakan maka tidak akan mempengaruhi yang lainnya (jaringan).

1.6.3.1 Bagaimana Jaringan Star Bekerja

Setiap komputer dalam jaringan bintang berkomunikasi dengan central hub yang mengirimkan kembali pesan ke semua komputer (dalam broadcast star network) atau hanya ke komputer yang dituju (dalam switched star network). Hub dalam broadcast star network dapat menjadi aktif ataupun pasif. Active hub

memperbaharui sinyal elektrik yang diterima dan mengirimkannya ke semua komputer yang terhubung ke hub. Hub tipe tersebut sering disebut juga dengan multiport repeater. Jika kita menggunakan hub memiliki 32 port, dengan seluruh port terisi, maka collision akan sering terjadi yang akan mengakibatkan kinerja jaringan menurun. Untuk menghindari hal tersebut kita bisa menggunakan switch yang memiliki kemampuan untuk menentukan jalur tujuan data. Active hub dan switch membutuhkan tenaga listrik untuk menjalankannya. Pasisive hub, seperti wiring panel atau blok punch-down, hanya berfungsi sebagai titik koneksi (connection point) dan tidak melakukan penguatan sinyal atau memperbaharui sinyal. Passive hub tidak membutuhkan tenaga listrik untuk menjalankannya.

1.6.3.2 Keuntungan dari penggunaan topologi star:

- Cukup mudah untuk mengubah dan menambah komputer ke dalam jaringan yang menggunakan topologi star tanpa mengganggu aktvitas jaringan yang sedang berlangsung.
- Pusat dari jaringan star merupakan tempat yang baik untuk menentukan diagnosa kesalahan yang terjadi dalam jaringan. Intelligent hub merupakan hub yang dilengkapi dengan microprocessors yang selain memiliki fitur sebagai tambahan untuk mengulang sinyal jaringan juga melakukan monitor yang terpusat dan manajemen terhadap jaringan.
- Apabila satu komputer yang mengalami kerusakan dalam jaringan maka komputer tersebut tidak akan membuat mati seluruh jaringan star. Hub dapat mendeteksi kesalahan dalam jaringan dan memisahkan komputer

yang rusak tersebut dari jaringan dan memperkenankan jaringan untuk beroperasi kembali.

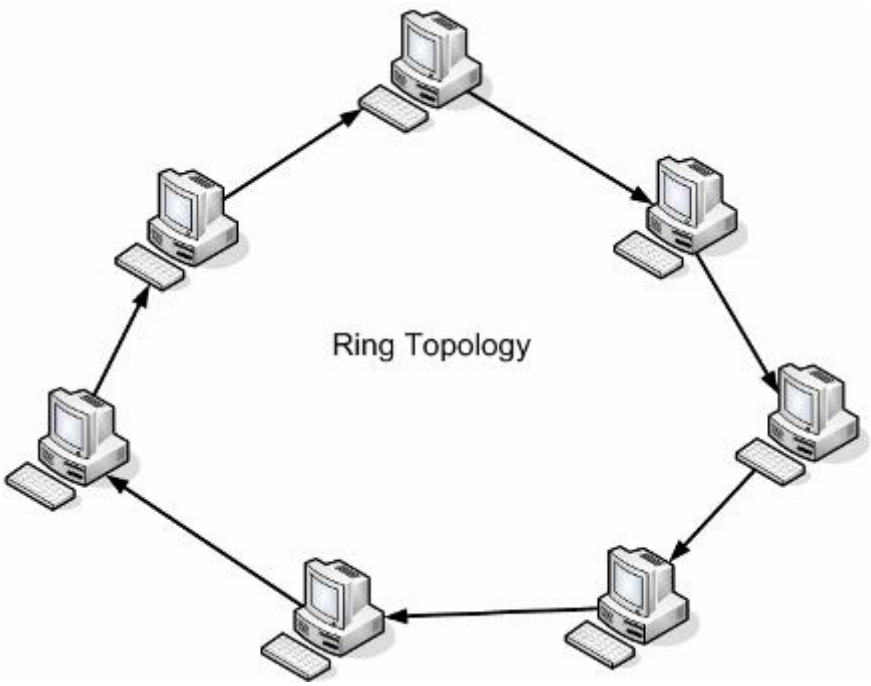
- Kita dapat menggunakan beberapa tipe kabel di dalam jaringan yang sama dengan hub yang dapat mengakomodasi tipe kabel yang berbeda.

1.6.3.3 Kekurangan dari penggunaan Topologi Star

- Topologi star mempunyai kekurangan sebagai berikut:
- Memiliki satu titik kesalahan, terletak pada hub. Jika hub pusat mengalami kegagalan,maka seluruh jaringan akan gagal untuk beroperasi.
- Memerlukan alat pada central point untuk mem-broadcast ulang atau pergantian traffic jaringan (switch network traffic).
- Membutuhkan lebih banyak kabel karena semua kabel jaringan harus ditarik ke satu central point, jadi lebih banyak membutuhkan lebih banyak kabel daripada topologi jaringan yang lain.

1.6.4 Topologi Cincin (Ring)

Penempatan kabel yang sederhana. Pada topologi ring, setiap komputer terhubung ke komputer selanjutnya, dengan komputer terakhir terhubung ke komputer yang pertama. Tetapi sayangnya, jika akan dilakukan penambahan atau pengurangan komputer dalam jaringan tentu saja akan mengganggu keseluruhan jaringan.



Gambar 2. 13 Topologi Ring

Topologi ring digunakan dalam jaringan yang memiliki performance tinggi, jaringan yang membutuhkan bandwidth untuk fitur yang time-sensitive seperti video dan audio, atau ketika performance dibutuhkan saat komputer yang terhubung ke jaringan dalam jumlah yang banyak.

1.6.4.1 Bagaimana Jaringan Ring Bekerja

Setiap komputer terhubung ke komputer selanjutnya dalam ring, dan setiap komputer mengirim apa yang diterima dari komputer sebelumnya. Pesan-pesan mengalir melalui ring dalam satu arah. Setiap komputer yang mengirimkan apa yang diterimanya, ring adalah jaringan yang aktif. Tidak ada akhir pada ring.

Beberapa jaringan ring melakukan token passing. Pesan singkat yang disebut dengan token dijalankan melalui ring sampai sebuah komputer menginginkan untuk mengirim informasi ke komputer yang lain. Komputer tersebut lalu mengubah token tersebut, dengan menambahkan alamatnya dan menambah data, dan mengirimnya melalui ring. Lalu setiap komputer secara berurutan akan menerima token tersebut dan mengirimkan informasi ke komputer selanjutnya sampai komputer dengan alamat yang dituju dicapai atau token kembali ke komputer pengirim (asal pengirim pesan). Komputer penerima akan membalas pesan ke asal pengirim pesan tadi mengindikasikan bahwa pesan sudah diterima. Lalu asal pengirim pesan akan membuat token yang lain dan menaruhnya di dalam jaringan, dan token tersebut akan terus berputar sampai ada komputer lain yang menangkap token tersebut dan siap untuk memulai pengiriman.

1.6.4.2 Keuntungan dari penggunaan topologi ring:

- Tidak ada komputer yang memonopoli jaringan, karena setiap komputer mempunyai hak akses yang sama terhadap token.
- Data mengalir dalam satu arah sehingga terjadinya collision dapat dihindarkan.

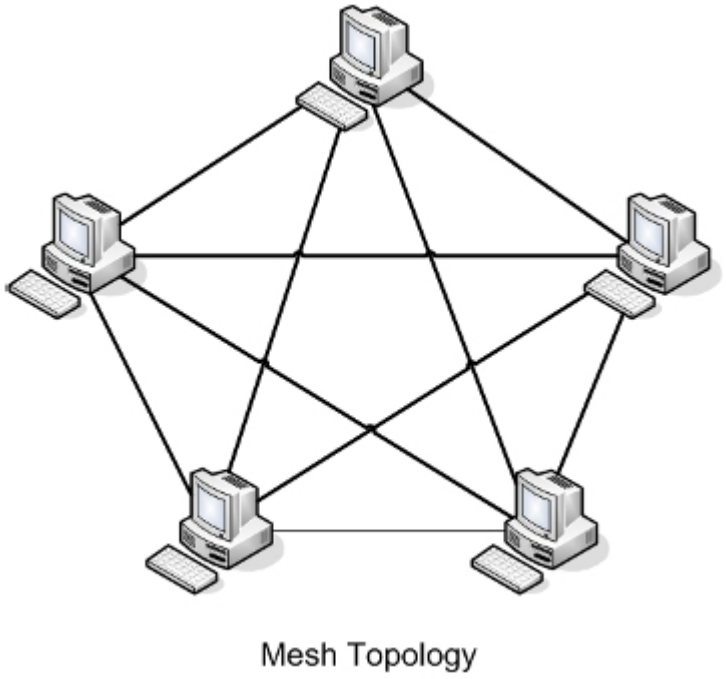
1.6.4.3 Kekurangan dari penggunaan Topologi Ring

- Apabila ada satu komputer dalam ring yang gagal berfungsi, maka akan mempengaruhi keseluruhan jaringan.
- Sulit untuk mengatasi kerusakan di jaringan yang menggunakan topologi ring.

- Menambah atau mengurangi komputer akan mengacaukan jaringan.
- Sulit untuk melakukan konfigurasi ulang.

1.6.5 Topologi Mesh

Di antara topologi yang lain topologi mesh memiliki hubungan yang berlebihan antara peralatan-peralatan yang ada. Jadi susunannya, setiap peralatan yang ada didalam jaringan saling terhubung satu sama lain. Dapat dibayangkan jika jumlah peralatan yang terhubung sangat banyak, tentunya ini akan sangat sulit sekali untuk dikendalikan dibandingkan hanya sedikit peralatan saja yang terhubung.



Gambar 2. 14 Topologi Mesh

1.6.5.1 Instalasi Mesh

Kebanyakan jaringan yang menggunakan topologi mesh akan mengalami kesulitan dalam instalasi jika peralatan yang terhubung jumlahnya

bertambah banyak, karena jumlah hubungan yang disambungkan semakin banyak jumlahnya. Jadi jika ada n peralatan (komputer) yang akan kita sambungkan, maka perhitungannya adalah n(n-1)/2. Jadi jika komputer yang terhubung semakin banyak maka semakin banyak pula hubungan yang akan diatur. Topologi ini cocok untuk digunakan pada sistem yang kecil.

1.6.5.2 Keuntungan dari penggunaan topologi mesh:

- Terjaminnya kapasitas channel komunikasi, karena memiliki hubungan yang berlebih.
- Relatif lebih mudah untuk dilakukan troubleshoot.

1.6.5.3 Kerugian dari Penggunaan Topologi Mesh

- Topologi mesh mempunyai kekurangan sebagai berikut:
- Sulitnya pada saat melakukan instalasi dan melakukan konfigurasi ulang saat jumlah komputer dan peralatan-peralatan yang terhubung semakin meningkat jumlahnya.
- Biaya yang besar untuk memelihara hubungan yang berlebih.