BAB II

TINJAUAN PUSTAKA

2.1 Sejarah Instansi

Sejarah instansi memaparkan tentang sejarah instansi, struktur organisaisi dan kegiatan bisnis yang dikerjakan oleh instansi.

2.1.1 Latar Belakang Yayasan Sukapura

Perlu disadari bahwa pendidikan adalah upaya untuk mempersiapkan peserta didik ke tarap dewasa menuju kemandirian. Dalam peklaksanaannya pendidikan merupakan tanggungjawab bersama antara pihak pemerintah, orang tua, siswa dan masyarakat/lembaga swadaya masyarakat. untuk itu semua pihak dituntut ikut berperan serta dalam pendidikan tersebut.

Sukapura adalah kepanjangan dari Yayasan Susila Karya Purnama Rasa, merupakan perubahan baru dari nama Yayasan Badan Peningkatan Kesejahteraan Guru Swasta yang disingkat YPKGBS yang bergerak dalam bidang ***penyelenggaraan pendidikan dan pengajaran jalur sekolah dan jalur luar sekolah, pelayanan kesehatan, berbagai bidang kesejahteraan sosial dan bidang kegiatan lain yang berhubungan langsung atau tidak langsung dengan tujuan Yayasan.***

Yayasan YPKGBS didirikan pada tanggal 31 Agustus 1989 dengan Akte Notaris Obing Chabir Adikusumah, SH. No. 025/31/VIII/1989, dan berubah

nama menjadi Yayasan Sukapura dengan nomor pendirian 11 November 1998 dengan Akte Notaris Gina R. Koswara, SH Nomor 8.

Dalam melaksanakan tugas dan kegiatan sehari-hari, Yayasan Sukapura beralamat di Jalan Sukapura No. 4 Kiaracondong Bandung 40285 Jawa Barat, dengan kepengurusan sebagai berikut :

Ketua : Drs. H. Djenal Kamaludin

Sekretaris : Suherman, S.Pd

Bendahara : Agah Prawira Atmaja

Anggota : - Wagiman

- Ny. Hj. Enung Khasanah

- Adang Sodikin, S.Pd

Salah satu bidang garapan dari Yayasan Sukapura adalah mendirikan Sekolah Luar Biasa (SLB) untuk para penyandang cacat mental (Tunagrahita) dengan nama

sekolah SLB B-C Sukapura yang beralamat di jalan Sukapura No. 4 Kelurahan Sukapura, Kecamatan Kiaracondong Kota Bandung dan berstatus Swasta.

Dalam kegiatan Proses Belajar-mengajar di SLB B-C Sukapura pada saat itu dipimpin oleh seorang kepala sekolah definitip sebagai bantuan dari pemerintah (Departemen Pendidikan Nasional) yaitu :

Nama : Drs. Yusup Supian

Tempat tanggal lahir : Tasikmalaya, 15 Agustus 1955

NIP : 130806388

Pangkat/Golongan : Pembina /IVa

Saat itu SLB B-C Sukapura mendidik 73 orang siswa yang terdiri dari : 18 orang anak Autis, dan 55 orang anak Tunagrahita.

Berdasarkan surat Ijin Operasional dari Dinas Pendidikan Provinsi Jawa Barat, maka pada bulan Oktober 2003 SLB B-C Sukapura berkembang dan terpisah menjadi dua bagian /lokal yang terdiri dari :

1. SLB-B Sukapura

Dengan Nomor Ijin Operasional : 421.9/3915-PLB tanggal 8 Oktober 2003 dengan kepala sekolah :

Nama : Miftah, S.Pd

Tempat tanggal lahir : Bandung, 04 Agustus 1953

NIP : 130540337

Pangkat/Golongan : Pembina /Iva

2. SLB-C Sukapura

Dengan Nomor Ijin Operasional : 421.9/3915-PLB tanggal 8 Oktober 2003 dengan kepala sekolah :

Nama : Drs. Yusup Supian

Tempat tanggal lahir : Tasikmalaya, 15 Agustus 1955

NIP : 130806388

Pangkat/Golongan : Pembina Tk.I/IVb

Pada tanggal 20 Oktober 2006 SLB B Sukapura terakreditasi dengan nilai B untuk jenjang pendidikan SDLB, dengan nomor sertifikat Akreditas : 0200/101/BAS/2006, sedangkan SLB C Sukapura terakreditasi pada tanggal 16 Februari 2005 dengan nilai B bernomor sertifikat 420/599-Dikmenti/2005-02/Dbd : 8.

A. Tujuan

Adapun tujuan pendirian Yayasan Sukapura sesuai dengan yang tercantum pada Akte Notaris (1998 : 3) sebagai berikut :

1. Membantu pemerintah menyelenggarakan kegiatan pendidikan, kebudayaan, social dan kesehatan untuk memperluas kesempatan belajar mengembangkan kebudayaan serta meningkatkan kesejahteraan bagi masyarakat.
2. Membangun manusia yang beriman dan taqwa, berbudi pekerti luhur, sehat jasmani dan rokhani, cerdas dan terampil sehingga dapat membangun diri dan bangsanya.
3. Ikut berupaya mewujudkan kesejahteraan rakyat yang mencapai keadilan sosial menuju masyarakat adil dan makmur.
4. Turut berpartisipasi dalam mewujudkan kesejahteraan rakyat untuk mencapai keadilan sosial menuju masyarakat adil dan makmur.

B. SLB-B Sukapura

SLB-B Sukapura merupakan lembaga pendidikan khusus bagi anak Tunagrahita sebagai pengembangan dari SLB B-C Sukapura yang didirikan pada bulan April tahun 1990 dan mendapat Izin Operasional dari Dinas Pendidikan Provinsi Jawa Barat Nomor : 421.9/39154 PLB Tanggal 3 Oktober 2003 dan telah diakreditasi pada tahun 2006, dengan akreditas nilai B untuk jenjang pendidikan SDLB-B. Jenjang pendidikan di SLB-B Sukapura mulai dari TKLB, SDLB, SMPLB dan SMALB dengan penyelenggaraan kegiatan belajar-mengajarnya dilaksanakan pada pagi hari.

Pada tahun pelajaran 2006/2007 SLB-B Sukapura mendidik 20 orang siswa yang tersebar diberbagai jenjang dengan jumlah tenaga pengajar sebanyak 6 (enam) orang terdiri dari :

4 (empat) orang guru PNS, dan 2 (dua) orang guru sukarelawan.

Dalam melaksanakan Kegiatan Belajar Mengajar (KBM) sehari-hari SLB-B Sukapura menempati 5 (lima) lokal yang terdiri dari : 2 (dua) lokal dipergunakan untuk ruangan belajar yang dibagi menjadi 4 (empat) ruang yang terpisah dengan menggunakan sekat sebagai batasnya, dengan masing-masing ukuran 3 X 5 M2, 1 (satu) lokal ruang Kepala Sekolah dan ruang tamu berukuran 5 X 5 M2, 2 (dua) lokal ruang dapur dan gudang berukuran 2 X 5 M2.

Mengingat SLB-B Sukapura baru mendapat Ijin Operasional dari Dinas Pendidikan Provinsi Jawa Barat pada tahun 2003, maka sarana dan prasarana yang diperlukan seperti Ruang BPBI, Ruang Latihan Bicara, mebeler, alat KBM, dan

buku-buku pelajaran belum lengkap sehingga membutuhkan fasilitas pendidikan yang layak untuk memenuhi syarat-syarat Standar Minimum Pendidikan Luar Biasa.

Saat ini SLB-B Sukapura mencoba mengembangkan Metode Percakapan yang harus ditunjang dengan adanya ruangan khusus, peralatan yang memadai dan buku-buku yang lengkap yang dapat diambil dari buku-buku pelajaran anak-anak normal untuk memacu dan meningkatkan kemampuan serta mereka dalam menghadapi Ujian Nasinal .

C. Visi dan Misi

1. Visi dan Misi Sekolah

"Terlayaninya Anak Berkebutuhan Khusus (ABK) di SLB-B Sukapura dalam bidang Pendidikan, keterampilan dan cakap dalam berkomunikasi sehingga menghasilkan lulusan yang beriman, bertaqwa dan berakhlak mulia serta terampilan dan bermutu pada tahun 2010".

2. Misi SLB-B Sukapura :

a. Menciptakan suasana yang kondusif, aktif, kreatif pada komponen-komponen sekolah
b. Melaksanakan pembelajaran dengan perencanaan yang terprogram disesuaikan dengan ke mampuan, potensi siswa melalui metode percakapan
c. Meningkatkan prestasi kerja guru secara optimal untuk melayani siswa

d. Meningkatkan kemampuan komunikasi anak sesuai dengan potensi yang dimiliki

e. Menanamkan rasa percaya diri dengan pembekalan keterampilan produktif dengan bakat dan minat siswa.

f. Bekerja sama dengan sekolah umum untuk mengembangkan siswa di sekolah inklusif

g. Bekerjasama dengan instansi, perusahaan guna menyalurkan bakat kemampuan dan ke terampilan Anak Berkebutuhan Khusus (ABK)

D. Data Guru

Table 2.1 Data guru SLB-B

NO	NAMA/NIP/NIGB	GOL/ RUANG	JABATAN GURU	JENIS GURU	BIDANG STUDY	SMT		JML JAM	KETERANGAN TUGAS KHUSUS
						I	II		
1	Miftah, S.Pd 130540337/ B.995409	VI/b	Guru Pembina Tk.I	Guru Mata Pelajaran	BPBI/ Latihan Bicara	12	12	24	Kepala Sekolah
2	Anna Budhiarti, S.Pd 131676384/ E.785770	VI/b	Guru Pembina Tk.I	Guru Kelas 3 SDLB	-	32	32	64	WKS Sie Kurikulum
3	Dikdik Dwiyana Koswara, S.Pd 132012123/ F.001591	IV/a	Guru Pembina	Guru Kelas 2 SMPLB	-	36	36	72	WKS Sie Kesiswaan
4	Yenni Surayani, S.Pd 132047178/ G.063145	III/d	Guru Dewasa Tk.I	Guru Kelas 6 SDLB	-	32	32	64	WKS Sie Sarana Prasarana
5	Siska Sugianti	-	-	Guru Kelas 1 SDLB	-	30	30	60	Kepramukaan

2.2 LANDASAN TEORI

Landasan teori menjelaskan beberapa teori yang berkaitan dengan permasalahan yang dibahas sebagai dasar pemahaman dalam sebuah sistem serta metode yang dipakai untuk kegiatan pengembangan terhadap sistem sendiri.

2.2.1 *Sistem pakar*

Sistem pakar atau *Expert System* biasa disebut juga dengan *knowledge-based system"* yaitu suatu aplikasi komputer yang ditujukan untuk membantu pengambilan keputusan atau pemecahan persoalan dalam bidang yang spesifik. Sistem ini bekerja dengan menggunakan pengetahuan *(knowledge)* dan metode analisis yang telah didefinisikan terlebih dahulu oleh pakar yang sesuai dengan bidang keahliannya. Sistem ini disebut sistem pakar karena fungsi dan perannya sama seperti seorang ahli yang harus memiliki pengetahuan, pengalaman dalam memecahkan suatu persoalan. Sistem biasanya berfungsi sebagai kunci penting yang akan membantu suatu sistem pendukung keputusan atau sistem pendukung eksekutif.

Sistem pakar terdiri dari dua komponen utama yaitu: basis pengetahuan *(knowledge base)* dan alat pengambilan kesimpulan (*inference engine).* Biasa pengetahuan didapat dari akumulasi pengetahuan pakar pada bidang tertentu.

Pengetahuan disini didefinisikan sebagai kumpulan data dan himpunan aturan untuk memanipulasi atau mengolah data untuk menjadi pengetahuan baru. Basis pengetahuan merupakan komponen penting dari suatu sistem pakar, besar

kecilnya kemampuan sistem pakar biasanya ditentukan oleh kapasitas dari basis pengetahuannya, sedangkan mesin pengambil keputusan adalah aplikasi yang membantu dan memandu pengguna sistem pakar dalam memanipulasi data dan memilih pengetahuan yang sesuai untuk mendapatkan kesimpulan.

2.2.2 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.

2.2.3 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.

2.2.4 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.

2.2.5 Keuntungan Pemakaian Sistem Pakar

1. Membuat seorang yang awam dapat bekerja seperti layaknya seorang pakar.

2. dapat bekerja dengan informasi yang tidak lengkap atau tidak pasti.
3. ES menyediakan nasihat yang konsisten dan dapat mengurangi tingkat kesalahan.

4. Membuat peralatan yang kompleks lebih mudah dioperasikan karena ES dapat melatih pekerja yang tidak berpengalaman.
5. ES tidak dapat lelah atau bosan, juga konsisten dalam memberi jawaban dan selalu memberikan perhatian penuh.
6. Memiliki kemampuan untuk memecahkan masalah yang kompleks.
7. Memungkinkan pemindahan pengetahuan ke lokasi yang jauh serta memperluas jangkauan seorang pakar, dapat diperoleh dan dipakai dimana saja.

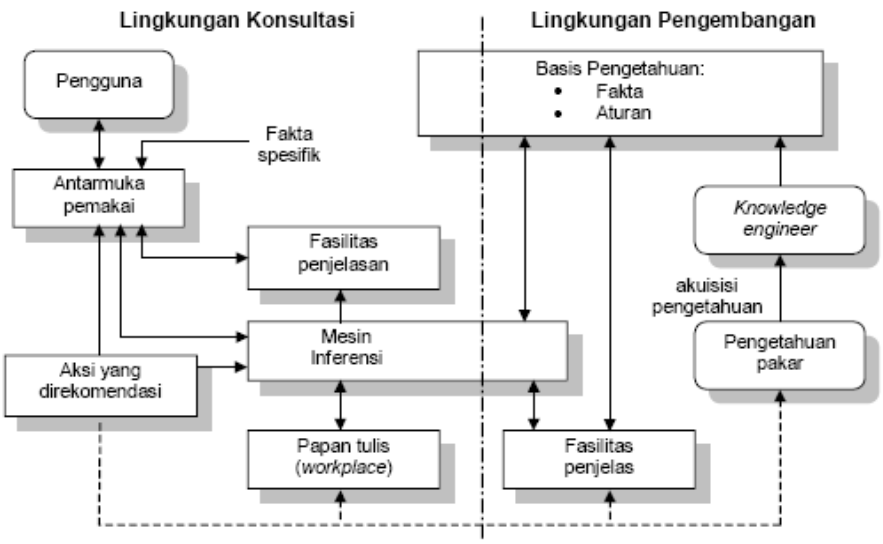
2.2.6 *Kelemahan Pemakaian Sistem Pakar*

1. Biaya yang diperlukan untuk membuat dan memeliharanya sangat mahal.
2. Sulit dikembangkan. Hal ini tentu saja erat kaitannya dengan ketersediaan pakar di bidangnya.
3. Sistem pakar tidak 100% bernilai benar.

2.2.7 Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian pokok, yaitu lingkungan pengembangan *(development environment)* dan lingkungan konsultasi *(consultation environment).*

Lingkungan pengembangan digunakan sebagai pembangun sistem pakar baik dari segi pembangun komponen maupun basis pengetahuan. Lingkungan konsultasi digunakan oleh seorang yang bukan ahli untuk berkonsultasi (Gambar 2.2 Struktur Sistem Pakar).



Gambar 2.1 Struktur Sistem Pakar

Papan Tulis *(Blackboard/Workplace),* adalah memori/lokasi untuk bekerja dan menyimpan hasil sementara. Biasanya berupa sebuah basis data. Antarmuka Pemakai *(User Interface).* Sistem Pakar mengatur komunikasi antara pengguna dan komputer. Komunikasi ini paling baik berupa bahasa alami, biasanya disajikan dalam bentuk tanya-jawab dan kadang ditampilkan dalam bentuk gambar atau grafik. Antarmuka yang lebih canggih dilengkapi dengan percakapan *(voice communication).*

Subsistem Penjelasan *(Explanation Facility).* Kemampuan untuk menjejak *(tracing)* bagaimana suatu kesimpulan dapat diambil merupakan hal yang sangat penting untuk transfer pengetahuan dan pemecahan masalah.

Komponen subsistem penjelasan harus dapat menyediakannya yang secara interaktif menjawab pertanyaan pengguna, Misalnya:

1. "Mengapa pertanyaan tersebut anda tanyakan?"
2. "Seberapa yakin kesimpulan tersebut diambil?"
3. "Mengapa alternatif tersebut ditolak?"
4. "Apa yang akan dilakukan untuk mengambil suatu kesimpulan?"
5. "Fakta apalagi yang diperlukan untuk mengambil kesimpulan akhir?"

Sistem Penghalusan Pengetahuan *(Knowledge Refining System).* Seorang pakar mempunyai sistem penghalusan pengetahuan artinya, mereka bisa menganalisa sendiri performa mereka, belajar dari pengalaman, serta meningkatkan pengetahuannya untuk konsultasi berikutnya. Pada Sistem Pakar, swa-evaluasi ini penting sehingga dapat menganalisa alasan keberhasilan atau kegagalan pengambilan kesimpulan., serta memperbaiki basis pengetahuannya.

2.2.8 Komponen-komponen Sistem Pakar

1. Basis Pengetahuan *(Knowledge Base)*

Pengetahuan merupakan kemampuan untuk membentuk model mental yang menggambarkan obyek dengan tepat dan mempresentasikannya dalam aksi yang dilakukan terhadap suatu obyek (Martin dan Oxman, 1988).

Pengetahuan dapat diklasifikasikan menjadi tiga, yaitu pengetahuan prosedural *(procedural knowledge),* pengetahuan deklaratif *(declaratif knowlwdge),* dan pengetahuan tacit *(tacit knowledge).* Pengetahuan prosedural lebih menekankan pada bagaimana melakukan sesuatu, pengetahuan deklaratif menjawab pertanyaan apakah sesuatu bernilai sal ah atau benar, sedangkan

pengetahuan tacit merupakan pengetahuan yang tidak dapat diungkapkan dengan bahasa. Basis Pengetahuan merupakan inti program Sistem Pakar dimana basis pengetahuan ini adalah representasi pengetahuan *(Knowledge Representation)* dari seorang pakar.

Pengetahuan dapat dipresentasikan dalam bentuk yang sederhana atau kompleks, tergantung dari masalahnya (Schnupp, 1989).

Ada beberapa model representasi yang penting yaitu : logika *(logic),* jaringan semantik *(semantic nets),* bingkai *(frame),* kaidah produksi *(production rule).*

a. Logika *(logic)*

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.*

b. Jaringan Sematik

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

c. 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.

d. Kaidah Produksi

Metode kaidah produksi biasanya dituliskan dalam bentuk jika maka *(if-theri)*. Kaidah ini dapat dikatakan sebagai hubungan implikasi dua bagian, yaitu: pertama jika (premise) dan yang kedua, yaitu muka (konkulasi). Apabila bagian jika dipenuhi maka bagian muka akan bernilai benar.

2. Basis Data *(Data Base)*

Basis Data adalah bagian yang mengandung semua fakta-fakta, baik fakta awal pada saat sistem mulai beroperasi maupun fakta-fakta yang didapatkan pada saat pengambilan kesimpulan yang sedang dilaksanakan. Dalam praktiknya, Basis data berada di dalam memori komputer. Kebanyakan Sistem Pakar mengandung Basis Data untuk menyimpan data hasil observasi dan data lainnya yang diburuhkan selama pengolahan.

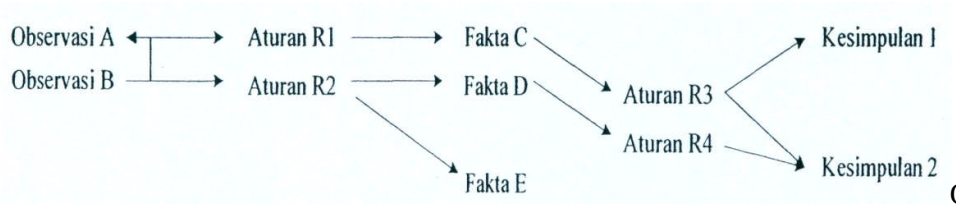
3. Mesin Inferensi (Inferensi Engine)

Mesin Inferensi adalah bagian yang mengandung mekanisme fungsi berpikir dan pola-pola penalaran sistem yang akan menganalisis suatu masalah tertentu dan selanjutnya akan mencari jawaban atau kesimpulan yang terbaik. Secara deduktif mesin inferensi memilih pengetahuan yang relevan dalam rangka mencapai kesimpulan. Dengan demikian sistem ini dapat menjawab pertanyaaan pemakai meskipun jawaban tersebut tidak tersimpan secara eksplisit di dalam basis pengetahuan. Mesin Inferensi memulai pelacakannya dengan mencocokan kaidah-kaidah dalam basis pengetahuan dengan fakta-fakta yang ada dalam basis data. Dibawah ini ada 2 macam metode inference, yaitu :

a. *Forward Chaining* (Pelacakan ke Depan)

Pendekatan yang dimotori oleh data (data driven). Dalam pendekatan ini pelacakan dimulai dari informasi masukan dan selanjutnya mencoba menggambarkan kesimpulan. Aruran dalam sistem merepresentasikan aksi-aksi yang harus diambil apabila terdapat suatu kondisi khusus pada item-item dalam memori kerja yang disebut himpunan aturan kondisi-aksi. Dalam metode ini, data digunakan untuk menentukan aturan mana yang akan dijalankan, kemudian aturan tersebut dijalankan. Mungkin proses menambahkan data ke memori kerja. Proses diulang sampai ditemukan suatu hasil. Aktivitas sistem dilakukan berdasarka siklus mengenal-beraksi *(recognize-act).*

Pelacakan ke depan mencari fakta yang sesuai dari bagian IF dari aturan IF-THEN (Gambar 2.3 menunjukkan proses forward chaining).



Ga

Gambar 2.2 *Forward Chaining*

b. *Backward Chaining* (Pelacakan Kebelakang)

Merupakan penalaran dari node tujuan dan bergerak ke belakang menuju keadaan awal, dalam penalaran ke belakang prosesnya disebut terarah, menggunakan pendekatan goal-driven, dimulai dari ekspektasi apa yang diinginkan terjadi *(hipotesis),* kemudian mengecek pada sebab-sebab yang mendukung (ataupun kontadiktif) dari ekspektasi tersebut.

Contoh :

Rl : IF suku bunga turun THEN harga obligasi naik

R2 : IF suku bunga naik THEN harga obligasi turun

R3 : IF suku bunga tidak berubah THEN harga obligasi tidak berubah

R4 : IF dolar naik THEN suku bunga turun

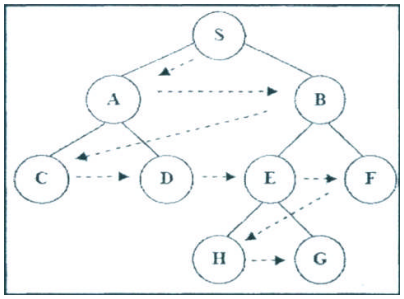
R5 : IF dolar turun THEN suku bunga naik

R6 : IF harga obligasi turun THEN beli obligasi

2.2.9 *Metode Pencarian*

2.2.9.1 Pencarian Melebar Pertama (*Breadth-First Search*)

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 2.4 Metode Breadth-First Search)



Gambar 2.3 Metode *Breath-first Search*

Keuntungan :

Tidak akan menemui jalan buntu

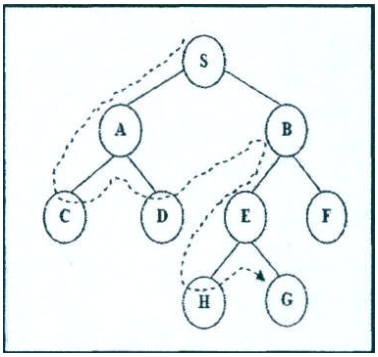
Jika ada 1 solusi, maka *breadth-first search* solusi akan menemukannya dan jika ada lebih dari satu solusi, maka solusi minimum akan ditemukan.

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).

2.2.9.2 Pencarian Mendalam Pertama *(Depth - First Search)*

Pada metode *Depth - First Search,* 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 2.10 Metode *Depth - First Search).*



Gambar 2.4 Metode *Depth-First Search*

Keuntungan :

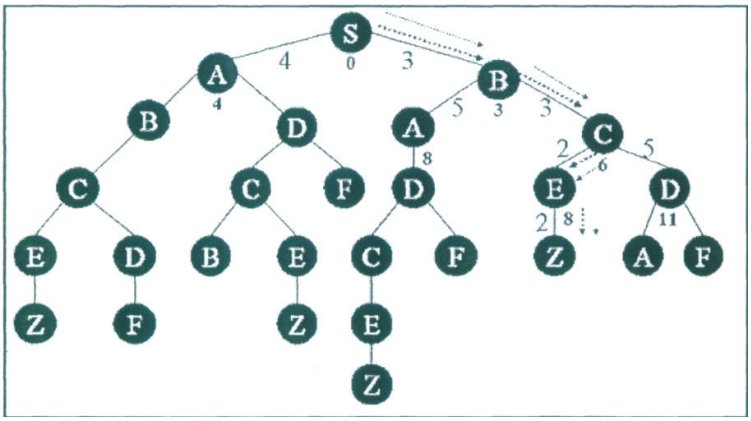
1. Membutuhkan memori relatif kecil, karena hanya node-node pada lintasan yang aktif saja yang disimpan.
2. Secara kebetulan, metode *Depth — First Search* akan menemukan solusi tanpa harus menguji lebih banyak lagi dalam ruang keadaan.

Kelemahan :

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

2.2.10.3 Pencarian Terbaik Pertama *(Best First Search)*

Metode Pencarian Terbaik Pertama *(Best First Search)* merupakan kombinasi dari metode *depth first search* dan metode *breadth first search* 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 search,* pencarian diperbolehkan mengunjungi node di lebih rendah dan jika ternyata node dilevel lebih tinggi memiliki nilai heuristik lebih buruk.



Gambar 2.5 Metode Best First Search;

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 optima.

2.2.10 *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.

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

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.

2.2.11 *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. 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.

2.2.12 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. 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.

2.2.12.2 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).*

2.2.12.3 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.

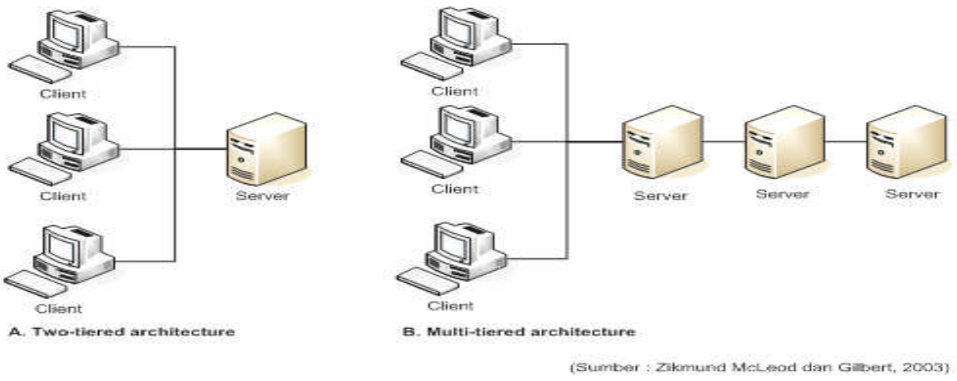
2.2.12.4 Sistem *Client-Server*

Arsitektur *client server* dapat dikatakan arsitektur yang paling baik dari paling banyak dipakai di berbagai tempat. Pada arsitektur ini ada dua komponen utama yaitu *client* merupakan terminal yang menjalankan *(request)* basis data dan *server* merupakan terminal memberikan pelayanan *(response)* terhadap permintaan *(request)* yang dilakukan oleh terminal *client* atas proses tertentu. Dimana terletak *client* berisi aplikasi basis data dan *server* berisi DBMS dan basis data.

Setiap aktivitas yang dikehendaki oleh pemakai akan lebih dulu ditangani oleh *client*. *Client* selanjutnya mengupayakan agar semua proses sebisa mungkin

ditangani sendiri. Jika ada proses yang harus melibatkan data yang tersimpan pada basis data yang tersimpan di *server* seperti mengakses *stored procedure* atau lainya, barulah *client* melakukan kontak dengan s*erver*. Contohnya ada kebutuhan untuk mendapatkan informasi yang dikirim yang diwakili oleh baris-baris *(row)* dengan kriteria-kriteria tertentu *(*proses *query, stored procedure)* dari tabel yang diinginkan.

Untuk lebih jelasnya, arsitektur dari sistem *client-server*, seperti pada (Gambar 2.7.)

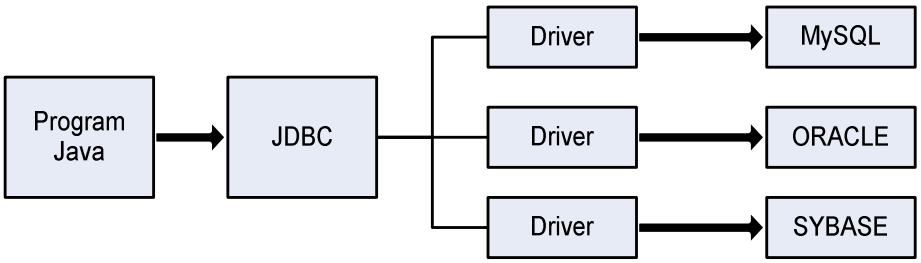


Gambar 2.6 Arsitektur *Client-Server*

2.2.13 *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 II.13 di bawah ini.



Gambar 2.7 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 (Rikyanto, 2002).

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.

2.2.14 MySQL

MySQL merupakan susunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL *(Structure Query Language).* Kendala dari suatu sistem *database* (DBMS) dapat diketahui dari cara kerja *optimezernya* dalam melakukan perintah-perintah SQL yang dibuat user ataupun program-program aplikasinya" KAD[8]. Sebagai *database server,* MySQL dapat dikatakan lebih unggul dibanding *database server* lainnya dalam query data. MySQL adalah salah satu dari sekian banyak sistem *database* yang merupakan terobosan solusi yang tepat dalam aplikasi *database* ,

MySQL adalah multi *user database* yang menggunakan bahasa *Strucktured Query Language* (SQL). *MySQL* mampu menangani data yang cukup besar. Perusahaan yang mengembangkan *MySQL* yaitu TcX, mengaku menyimpan data lebih dari 40 *database,* 10.000 tabel dan sekitar 7 juta baris, totalnya kurang lebih 100 *Gigabyte* data. SQL adalah bahasa standar yang digunakan untuk mengakses *database server.* Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi 25 dan digunakan sebagai standar industri.

Dengan menggunakan SQL, proses akses *database* menjadi lebih *user - friendly* dibandingkan dengan menggunakan *dBASE* atau *Clipper* yang masih menggunakan perintah - perintah pemrograman.A/y-S'jgL merupakan *software database* yang paling populer di lingkungan Linux, kepopuleran ini karena ditunjang performa *query* dari databasenya yang saat ini bisa dikatakan paling

cepat dan jarang bermasalah. *MySQL* ini juga sudah dapat berjalan pada lingkungan *Windows.*

Perintah untuk mengelola *database* dibagi menjadi 3 (tiga) kelompok, diantaranya :

1. Perintah untuk mendefinisikan data/DDL *(Data Definition Language).*
2. Perintah untuk memanipulasi data/DML *(Data Manipulation Language).*
3. Perintah untuk mengendalikan data/DCL *(Data Control Language).*

2.2.15 *Eclipse*

Eclipse adalah IDE (*integrated development environment*) *tools Java editor* profesional, untuk mendesain, melakukan *coding*, dan pengembangan dari *JSE, J2EE*, dan *web application. Tools* ini memiliki fitur *visual editing* dengan menambahkan *plugin WindowBuilder* yang memungkinkan *user* membuat aplikasi tanpa harus menulis barisan kode *Java. Eclipse* membantu membangun aplikasi desktop dan *web applications* dengan menggunakan bahasa pemrograman *Java.*

2.2.16 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.* 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.* [6].

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*

Terdapat empat macam hubungan didalam penggunaan UML, yaitu; *dependency, association, generalization,* dan *realization.*

Pertama, sebuah *dependency* adalah hubungan semantik antara dua benda/things yang mana sebuah benda berubah mengakibatkan benda satunya akan berubah pula. Umumnya sebuah *dependency* digambarkan sebuah panah dengan garis terputus-putus.

Kedua, sebuah *association* adalah hubungan antar benda struktural yang terhubung diantara obyek. Kesatuan obyek yang terhubung merupakan hubungan khusus, yang menggambarkan sebuah hubungan struktural diantara seluruh atau sebagian. Umumnya *assosiation* digambarkan dengan sebuah garis yang dilengkapi dengan sebuah label, nama, dan status hubungannya.

Ketiga, sebuah *generalization* adalah menggambarkan hubungan khusus dalam obyek anak/*child* yang menggantikan obyek *parent* / induk . Dalam hal ini, obyek anak memberikan pengaruhnya dalam hal struktur dan tingkah lakunya kepada obyek induk.

Keempat, sebuah *realization* merupakan hubungan semantik antara pengelompokkan yang menjamin adanya ikatan diantaranya. Hubungan ini dapat diwujudkan diantara *interface* dan kelas atau *elements,* serta antara *use cases* dan *collaborations.*

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. [6].

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. [6].

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*. [6].

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*.