# BAB II

# TINJAUAN PUSTAKA

## II.1 Landasan Teori

### II.1.1 Kecerdasan Buatan ( *Artificial Intelligence )*

Artificial Intelligence / AI adalah aktifitas mengikut sertakan mesin seperti computer yang mempunyai kemampuan untuk menampilkan tingkah laku yang dianggap intelligent jika diban dingkan dengan manusia. (Untung Arisandi, 2006 : 6)

Pengertian kecerdasan buatan dapat di pandang dari berbagai sudut pandang, antara lain :

1. Sudut pandang kecerdasan. Kecerdasan buatan akan membuat mesin menjadi cerdas dalam arti mampu berbuat seperti apa yang dilakukan manusia.
2. Sudut pandang penelitian. Kecerdasan buatan adalah suatu studi bagaimana membuat agar komputer dapat melakukan sesuatu sebaik yang dikerjakan manusia.
3. Sudut pandang bisnis. Kecerdasan buatan adalah kumpulan peralatan yang sangat powerful dan metodologis dalam mnyelesaikan masalah-masalah bisnis.
4. Sudut pandang pemrograman. Kecerdasan meliputi studi tentang pemrograman simbolik, penyelesaian masalah dan pencarian.

   tau proses, dalam hubungan logik atau komputasional.

Jika dibandingkan dengan kecerdasan alami (kecerdasan yang dimiliki manusia), kecerdasan buatan , khusunya dalam bidang sistem pakar memiliki beberapa keuntungan secara komersial, antara lain (Rika Rosnelly, 2012)

1. Meningkatkan kecerdasan
2. Mengurangi biaya ( *Reduced coast* )
3. Mengurangi bahaya (*Reduced danger* )
4. Permanan ( *permanence* )
5. Keahlian multiple
6. Meningkatkan kehandalan
7. Penjelasan
8. Respon yang cepat
9. Stabil
10. Pembimbing pintar
11. Basis data cerdas

### II.1.2 Pengertian sistem

Sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu. Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel-variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Sistem bisa berupa abstraksi atau fisis (Gordon B. Davis, 2007).

### II.1.3 Pengertian informasi

Informasi adalah hasil pengolahan data atau fakta yang di kumpulkan dengan cara tertentu , informasi disajikan dalam bentuk yang mudah di pahami dan merupakan pengetahuan yang relevan yang di butuhkan dalan menambah wawasan bagi pemakainya guna mencapai suatu tujuan ( Budi Sutedjo Dharma Utomo , 2006 : 12 )

### II.1.4 Pengertian sistem informai

Sistem informasi adalah kumpulan elemen yang saling berhubungan antara satu dengan yang lain yang membentuk suatu kesatuan untuk mengintegrasi data, memproses dan menyimpan serta mendistribusikan informasi. Dengan kata lain , sistem informasi merupakan elemen – elemen yang saling berinteraksi secara sistematis dan teratur untuk menciptakan dan membentuk aliran informasi yang akan mendukung pembuatan keputusan dan melakukan kontrol terhadap jalanya perusahaan. ( Budi Sutedjo Dharma Oetomo, 2006 : 10 )

### II.1.5 Pengertian Sistem Pakar

Sistem Pakar ( *Expert System* ) adalah sistem komputer yang ditujukan untuk meniru semua aspek ( *Emulates* ) kemampuan mengambil keputusan ( *Decision making* ) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah. ( Rika Rosnelly , 2012 : 2 )

Sistem pakar (*expert system*) mulai dikembangkan pada pertengahan tahun 1960 -an oleh *Artificial Intelligence Corporation*. Sistem pakar yang muncul pertama kali adalah *General-purpose Problem Solver* (GPS) yang merupakan sebuah *predecessor* untuk menyusun langkah-langkah yang dibutuhkan untuk mengubah situasi awal menjadi state tujuan yang telah ditentukan sebelumnya dengan menggunakan domain masalah yang kompleks. Sistem pakar dapat diterapkan untuk persoalan di bidang industri, pertanian, bisni, kedokteran, militer, komunikasi dan transportasi, pariwisata, pendidikan, dan lain sebagainya. Permasalahan tersebut bersifat cukup kompleks dan terkadang tidak memiliki algoritma yang jelas di dalam pemecahannya, sehingga dibutuhkan kemampuan seorang atau beberapa ahli untuk mencari sistematika penyelesaiannya secara evolutif.

Sistem pakar merupakan program yang dapat menggantikan keberadaan seorang pakar. Alasan mendasar mengapa sistem pakar dikembangkan menggantikan seorang pakar adalah sebagai berikut :

1. Dapat menyediakan kepakaran setiap waktu dan di berbagai lokasi.
2. Secara otomatis mengerjakan tugas-tugas rutin yang membutuhkan seorang pakar.
3. Seorang pakar akan pensiun atau pergi.
4. Menghadirkan atau menggunkan jasa seorang pakar memerlukan biaya yang mahal.
5. Kepakaran dibutuhkan juga pada lingkungan yang tidak bersahabat (*hostile environment*).

## II.2 Teori dan Metode penulisan skripsi

### II.2.1 Konsep Dasar Sistem Pakar

Pengetahuan dari suatu sistem pakar mungkin dapat direpresentasikan dalam sejumlah cara. Salah satu metode yang paling umum untuk merepresentasikan pengetahuan adalah dalam bentuk tipe aturan (*rule*) *IF..Then* (Jika..maka). Walaupun cara diatas sangat sederhana, namun banyak hal yang berarti dalam membangun sistem pakar dengan mengekspresikan pengetahuan pakar dalam bentuk aturan diatas. Konsep dasar dari suatu sistem pakar mengandung beberapa unsur/elemen, yaitu: (Rika Rosnelly, 2012 : 6 )

1. Keahlian

   Keahlian merupakan suatu penguasaan pengetahuan dibidang tertentu yang didapatkan dari pelatihan, membaca atau pengalaman.

2. Ahli

   Seorang ahli adalah seorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan (domain), menyusun kembali pengetahuan, memecah aturan-aturan jika diperlukan dan menentukan relevan tidaknya keahlian mereka.

3. Pengalihan keahlian

   Pengahlian keahlian dari para ahli ke komputer untuk kemudian dialihkan lagi keorang lain yang bukan ahli (tujuan utama sistem pakar). Proses ini membutuhkan aktivitas, yaitu: tambahan pengetahuan (dari para ahli atau sumber-sumber lainnya), representasi pengetahuan yang berupa fakta dan

prosedur (ke komputer), inferensi pengetahuan dan pengalihan pengetahuan ke pengguna.

4. Inferensi

   Mekanisme inferensi merupakan perangkat lunak yang melakukan penalaran dengan menggunakan pengetahuan yang ada untuk menghasilkan suatu kesimpulan atau hasil akhir.

5. Aturan

   Aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

6. Kemampuan menjelaskan.

   Kemampuan komputer untuk memberikan penjelasan kepada pengguna tentang sesuatu informasi tertentu dari pengguna dan dasar yang dapat digunakan oleh komputer untuk dapat menyimpulkan suatu kondisi.

### II.2.2 Bidang-Bidang Pengembangan Sistem Pakar

Ada beberapa kategori pengembangan sistem pakar, antara lain:

1. **Identifikasi masalah dan kebutuhan.** Mengkaji dengan pasti tentang masalah yang akan dikomputerisasi dan apakah dengan sistem pakar bisa lebih membantu atau tidak ?
2. **Menentukan masalah yang cocok.** Dalam pembahasan ini ada beberapa syarat yang harus di penuhi agar sistem pakar dapat bekerja dengan baik , salah satunya adalah domain masalah tidak terlalu luas

3. **Mempertimbangkan alternatif.** Dalam hal ini ada 2 alternatif yaitu menggunakan sistem pakar atau komputer tradisional
4. **Menghitung pengembalian investasi.** Termasuk diantaranya biaya pembuatan sistem pakar, biaya pemeliharaan, dan biaya training
5. **Memilih alat pengembangan.** Bisa digunakan software pembuat sistem pakar atau dirancang dengan menggunakan bahasa pemrograman
6. **Rekayasa pengetahuan.** Perlu dilakukan penyempurnaan terhadap aturan – aturan yang sesuai
7. **Perencanaan sistem.** Perencanaan banyak digunakan dalam bidang bisnis dan keuangan suatu proyek, di mana sistem pakar dalam membuat perencanaan suatu pekerjaan berdasarkan jumlah tenaga kerja, biaya dan waktu sehingga pekerjaan menjadi lebih efisien.
8. **Melengkapi pengembangan.** Termasuk perkembangan *prototype* apabila sistem yang telah ada sudah sesuai dengan keinginan
9. **Menguji dan mencari kesalahan sistem.** Sistem pakar dengan seleksi mengidentifikasikan pilihan terbaik dari beberapa daftar pilihan kemungkinan solusi.
10. **Memelihara sistem.** Dalam hal ini harus di lakukan adalah mempebaharui pengetahuan, mengganti pengetahuan yang sudah ketinggalan, dan meluweskan sistem agar bisa lebih baik lagi dalam menyelesaikan masalah.

### II.2.3 Unified Modeling Language ( UML )

*Unified Modelling Language* (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta di itulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak.Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT( Object Modeling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering).

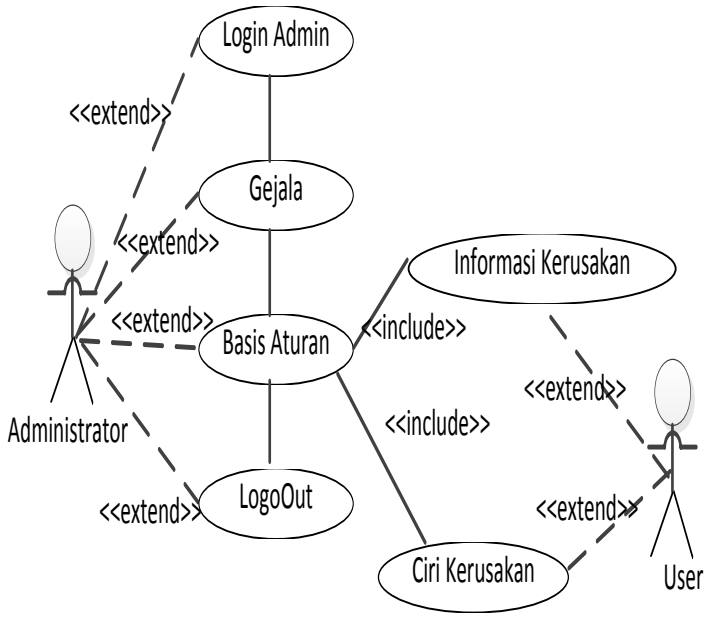
### II.2.4 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan

“bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu. Use case diagram dapat sangat membantu bila kita sedang menyusun requirement sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang test case untuk semua feature yang ada pada sistem. 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 akan dipanggil setiap kali use case yang meng-include dieksekusi secara normal. Sebuah use case dapat di-include oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang common.

( Suhariman Dirgantoro , 2010 : 12 )

Contoh dari use case diagrama dapat dilihar pada gambar dibawah ini :

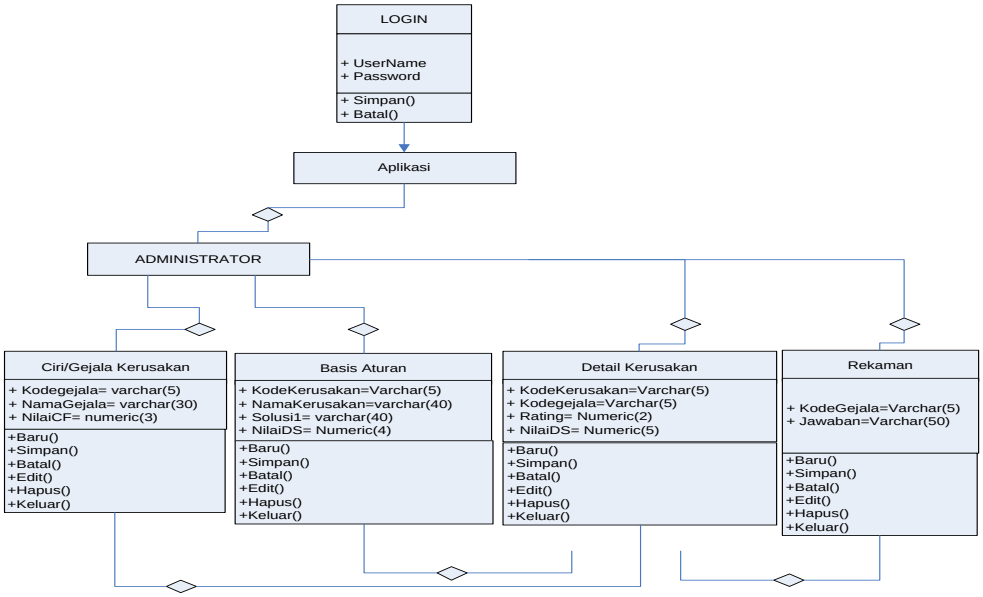



**Gambar. II.1 Use case diagram**

**(Sumber : Belajar UML , 2010 )**

### II.2.5 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaantersebut (metoda/fungsi). ( Suhariman Dirgantoro , 2010 : 20 ) Contoh class diagram dapat dilihat pada gambar dibawah ini .



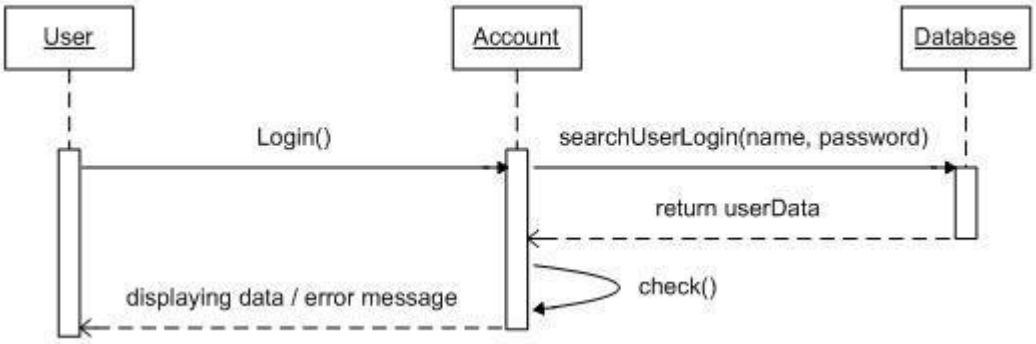

**Gambar. II.2 Class Diagram**

**(Sumber : Belajar UML , 2010 )**

### II.2.6 Squence Diagram

Diagram sekuen menggambarkan hubungan antara objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Diagram sekuen dapat berupa sebuah diagram sekuen yg bersifat umum yang menunjukkan semua skenario yang mungkin untuk semua use case, atau terpisah pisah untuk tiap skenario tungal di dlm use case . ( Suhariman Dirgantoro , 2010 : 28 )

Contoh squence diagram dapat di lihat pada gambar di bawah ini.



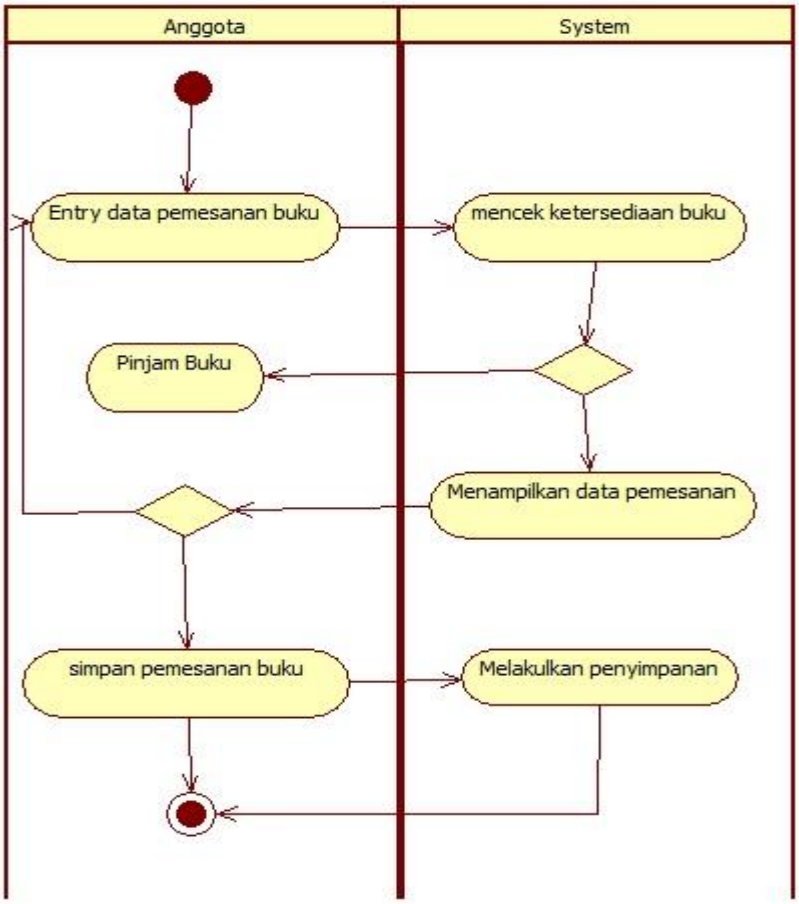

**Gambar. II.3 Squence Diagram**

**(Sumber : Belajar UML , 2010 )**

### II.2.6 Activity Diagram

Activity diagram memiliki pengertian yaitu lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. Memiliki struktur diagram yang mirip flowchart atau data flow diagram pada perancangan terstruktur. Memiliki pula manfaat yaitu apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Dan activity dibuat berdasarkan sebuah atau beberapa use case pada use case diagram. (Suhariman Dirgantoro ,2010 : 33)

Berikut adalah contoh dari activity diagram




**Gambar. II.4 Activity Diagram**

**(Sumber : Belajar UML , 2010 )**

### II.2.7 Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu: lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan sistem pakar digunakan

untuk memasukkan pengetahuan pakar kedalam lingkungan sistem pakar, sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar. Komponen-komponen yang terdapat dalam sistem pakar antara lain adalah segabai berikut (Kusrini, 2008):

1. Antarmuka pengguna (*user interface*)

   *User interface* merupakan mekanisme yang digunakan oleh pengguna dan sistem pakr untuk berkomunikasi. Antarmuka menerima informasi dari pemakai dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem. Pada bagian ini terjadi dialog antara program dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan informasi (*input*) dari pemakai, juga memberikan informasi (*output*) kepada pemakai.

2. Basis Pengetahuan

   Basis pengetahuan berisi pengetahuan-pengetahuan dalam penyelesaian masalah dalam domain tertentu.Ada dua bentuk pendekatan basis pengetahuan yang sangat umum digunakan, yaitu :

   a. Penalaran berbasis aturan (*Rule-Based Reasoning*)

      Pengetahuan direpresentasikan dengan menggunakan aturan berbentuk : IF-THEN. Bentuk ini digunakan apabila memiliki sejumlah pengetahuan pakar pada suatu permasalahan tertentu, dan pakar dapat menyelesaikan masalah tersebut secara berurutan.

   b. Penalaran berbasis kasus (*Case-Based Reasoning*)

Basis pengetahuan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu solusi untuk keadaan yang terjadi sekarang.

3. Akuisisi Pengetahuan (*knowledge acquisition*)

   Akuisisi pengetahuan adalah akumulasi, transfer, dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan kedalam program komputer. Dalam tahap ini *knowledge engineer* berusaha menyerap pengetahuan untuk selanjutnya di transfer ke dalam basis pengetahuan.Terdapat empat metode utama dalam akuisisi pengetahuan, yaitu: wawancara, analisis protocol, observasi pada pekerjaan pakar dan induksi aturan dari contoh.

4. Mesin inferensi

   Mesin inferensi merupakan perangkat lunak yang melakukan penalaran dengan menggunakan pengetahuan yang ada untuk menghasilkan suatu kesimpulan atau hasil akhir. Dalam komponen ini dilakukan permodelan proses berfikir manusia.

5. *Workplace*

   *Workplace* merupakan area dari sekumpulan memori kerja yang digunakan untuk merekam hasil-hasil dan kesimpulan yang dicapai. Ada tiga tipe keputusan yang direkam, yaitu :

   a. Rencana : Bagaimana menghadapi masalah.
   b. Agenda : Aksi-aksi yang potensial yang sedang menunggu untuk eksekusi.

c. Solusi : calon aksi yang akan dibangkitkan.

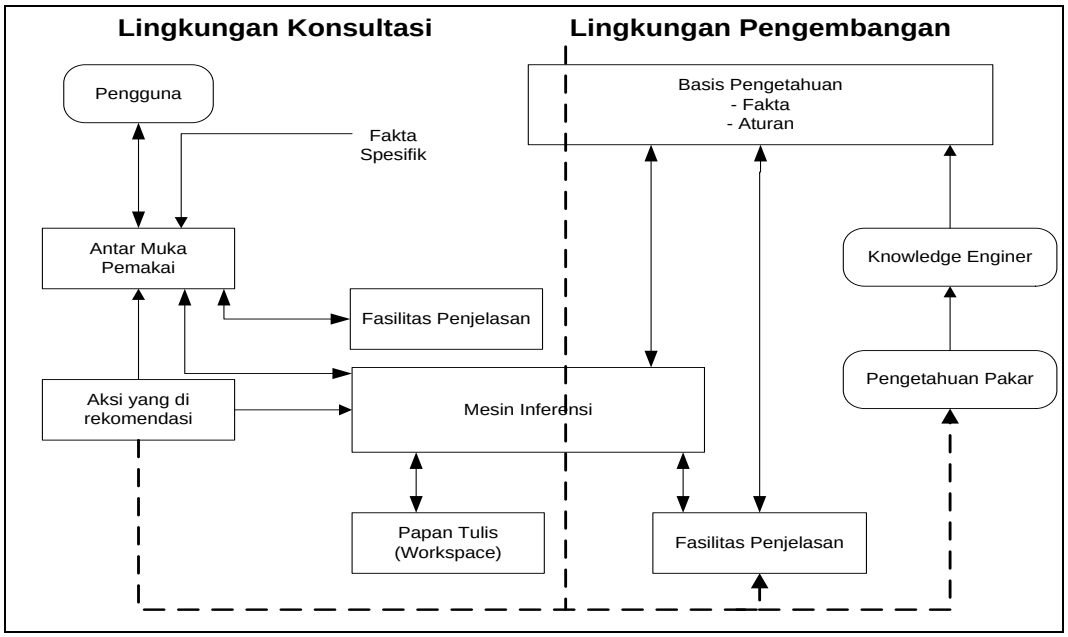
6. Fasilitas penjelasan

   Fasilitas penjelasan adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar. Komponen ini menggambarkan penalaran sistem kepada pemakai dengan cara menjawab pertanyaan-pertanyaan.

7. Perbaikan pengetahuan

   Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dan kinerjanya.

Komponen-komponen sistem pakar dalam kedua bagian tersebut dapat dilihat pada gambar 2.1 berikut ini:




**Gambar II.5 Komponen sistem pakar**
**(sumber: Kusrini S.kom (Aplikasi Sistem Pakar, 2008)**

### II.2.8 Keuntungan Sistem Pakar

Sistem pakar merupakan paket perangkat lunak atau paket program komputer yang ditujukan sebagai penyedia nasehat dan sarana bantu dalam menyelesaikan masalah di bidang-bidang spesialisasi tertentu. Ada beberapa keunggulan dari sistem pakar, diantaranya dapat (Kusrini, 2008):

1. Menghimpun data dalam jumlah yang sangat besar.
2. Menyimpan data tersebut untuk jangka waktu yang panjang dalam suatu bentuk tertentu.
3. Mengerjakan perhitungan secara cepat dan tepat dan tanpa jemu mencari kembali data yang tersimpan dengan kecepatan tinggi.

Ada banyak keuntungan yang dapat diperoleh dengan adanya sistem pakar antara lain sebagai berikut :

1. Memungkinkan orang awam dapat mengerjakan pekerjaan para ahli.
2. Dapat melakukan proses berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian para pakar.
4. Meningkat output dan produktivitas.
5. Meningkatkan kualitas.
6. Mampu mengambil dan melestarikan keahlian para pakar.
7. Mampu beroperasi dalam lingkungan yang berbahaya.
8. Memiliki kemampuan untuk mengakses pengetahuan.
9. Memiliki reliabilitas.
10. Meningkatkan kapabilitas sistem komputer.

11. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
12. Sebagai media pelengkap dalam pelatihan.

### II.2.9 Representasi pengetahuan

Representasi pengetahuan merupakan metode yang digunakan untuk mengkodekan pengetahuan dalam sebuah sistem pakar yang berbasis pengetahuan. Perepresentasian dimaksudkan untuk manangkap sifat-sifat penting problema dan membuat informasi itu dapat diakses oleh prosedure pemecahan problema (Kusrini, 2008).

Bahasa representasi harus dapat membuat seorang perogram mampu mengekspresikan pengetahuan yang diperlukan untuk mendapatkan solusi problema, dapat diterjemahkan ke dalam bahasa pemrograman dan dapat disimpan.Harus dirancang agar fakta-fakta dan pengetahuan lain yang terkandungdi dalamnya dapat digunakan untuk penalaran.

### II.2.10 Model Representasi Pengetahuan

Pengetahuan dapat direpresentasikan dalam bentuk yang sederhana atau kompleks, tergantung dari masalahnya. Beberapa model representasi pengetahuan yang penting, adalah:

1. Logika (*logic*)

   Logika merupakan suatu pengkajian ilmiah tentang serangkaian penalaran, sistem kaidah, dan prosedur yang membantu proses penalaran. Logika

merupakan representasi pengetahuan yang paling tua. Bentuk logika komputasional ada 2 macam, yaitu:

a. Logika Proporsional atau Kalkulus

   Logika proporsional merupakan logika simbolik untuk memanipulasi proposisi. Proposisi merupakan pernyataan yang dapat bernilai benar atau salah yang dihubungkan dengan operator logika diantaranya operator *And* (dan), *Or* (atau), *Not* (tidak), Impilikasi (*if..then*), Bikondisional (*if and only if*). Contohnya: Jika hujan turun sekarang maka saya tidak akan ke pasar, dapat dituliskan dalam bentuk: ( p => q)

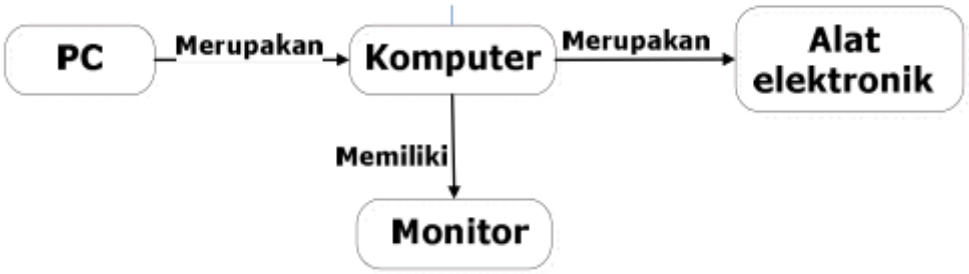
b. Logika Predikat

   Logika predikat adalah suatu logika yang seluruhnya menggunakan konsep dan kaidah proposional yang sama dengan rinci. Suatu proposisi atau premis dibagi menjadi dua bagian, yaitu: argumen (objek) dan predikat (keterangan). Predikat adalah keterangan yang membuat *argument* dan predikat. Contohnya: Mobil berada dalam garasi, dapat dinyatakan menjadi Di dalam (mobil,garasi). Di dalam = keterangan, mobil = argumen, garasi = argumen.

2. Jaringan semantik (*semantic nets*)

   Representasi jaringan semantic merupakan penggambaran grafis dari pengetahuan yang memperlihatkan hubungan hirarkis dari objek-objek yang terdiri atas simpul (*node*) dan penghubung (*link*). Contohnya : Merepresentasikan pernyataan bahwa semua komputer merupakan alat elektornik, semua PC merupakan komputer, dan semua komputer memiliki

monitor. Dari pernyataan tersebut dapat diketahui bahwa semua PC memiliki monitor dan hanya sebagian alat elektronik yang memiliki monitor, hal ini dapat dilihat pada gambar 2.2 berikut ini:




**Gambar II.6 Representasi jaringan semantic**
**(Sumber : Kusrini, 2008)**

3. *Object*-Atributte-*Value* (OAV)

   *Object* dapat berupa bentuk fisik atau konsep, *Attribute* adalah karakteristik atau sifat dari object tersebut, *Value* (nilai) - besaran spesifik dari attribute tersebut yang berupa numeric, string atau boolean. Contoh: Object: mangga ; Attribute: berbiji ; Value: tungggal.

4. Bingkai (*frame*)

   Bingkai berupa ruang (*slots*) yang berisi atribut untuk mendeskripsikan pengetahuan yang berupa kejadian. Binkai memuat deskripsi sebuah objek dengan menggunakan tabulasi informasi yang berhubungan dengan objek.

5. Kaidah produksi (*production rule*).

   Kaidah menyediakan cara formal untuk merepresentasikan rekomendasi, arahan, atau strategi. Kaidah produksi dituliskan dalam bentuk jika-maka (*if-then*) yang menghubungkan anteseden dengan konsekuensi yang

diakibatkannya. Berbagai struktur kaidah *if-then* ynag menghubungkan obyek atau atribut adalah sebagai berikut :

JIKA premis MAKA konklusi

JIKA masukan MAKA keluaran

JIKA kondisi MAKA tindakan

JIKA anteseden MAKA konsekuen

JIKA data MAKA hasil

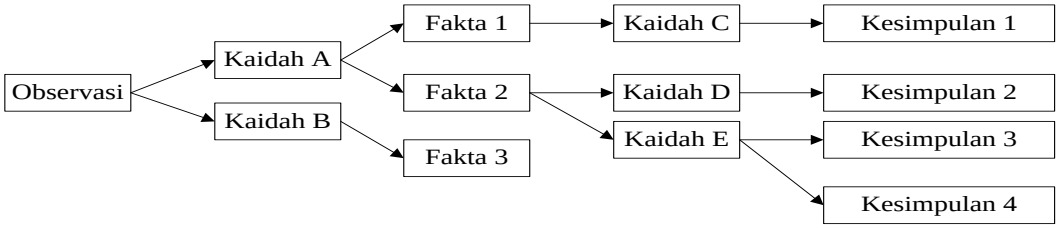
JIKA tindakan MAKA tujuan

### II.2.11 Inferensi

Inferensi merupakan proses untuk menghasilkan informasi dari fakta yang diketahui atau diasumsikan. Inferensi adalah konklusi logis (*logical conclusion*) atau implikasi berdasarkan informasi yang tersedia dalam hal ini akan digunakan metode inferensi dalam pengambilan kesimpulan. Ada dua metode inferensi yang penting dalam sistem pakar untuk menarik kesimpulan, yaitu (Kusrini, 2008*)*:

1. Forward chaining

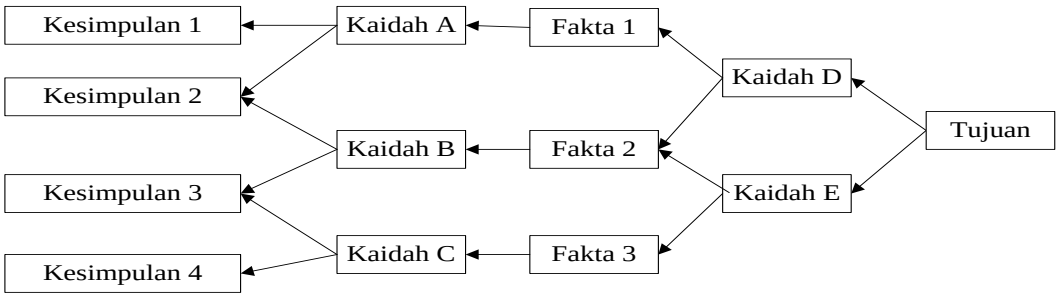
   Strategi dari sistem ini adalah dimulai dari inputan beberapa fakta, kemudian menurunkan beberapa fakta dari aturan-aturan yang cocok pada knowledge base dan melanjutkan prosesnya sampai jawaban sesuai. Forward chaining dapat dikatakan sebagai penelusuran deduktif.\




**Gambar II.7 Diagram Pelacakan ke Depan**
**(Sumber : Kusrini, 2008*)***

2. Backward chaining

   Strategi penarikan keputusan yang didasarkan dari hipotesa atau dugaan yang didapat dari informasi yang ada. Ciri dari strategi ini adalah pertanyaan user. Memperoleh fakta biasanya diajukan dalam bentuk "YA" atau "TIDAK", proses ini berdampak dengan diterima atau tidaknya hipotesis.




**Gambar II.8 Diagram Pelacakan ke Belakang**
**(Sumber : Kusrini, 2008)**

Contoh penalaran menggunakan metode runut balik :

Aturan 1 :

Mesin hidupnya tersendat-sendat setelah dipanaskan dengan CF 0,63

JIKA Torak, cincin torak dan lubang silinder aus/rusak

DAN Terlalu banyak mengisi oli pelumas

DAN Tekanan pompa oli terlalu tinggi

Aturan 2 :

Mesin hidupnya tersendat-sendat setelah dipanaskan dengan CF 0,63

JIKA Torak, cincin torak dan lubang silinder aus/rusak

ATAU Terlalu banyak mengisi oli pelum

ATAU Tekanan pompa oli terlalu tinggi

### II.2.12 Pengertian Metode Ketidakpastian (*Uncertainty*)

Ketidakpastian dapat dianggap sebagai suatu kekurangan informasi yang memadai untuk membuat suatu keputusan. Ketidakpastian merupakan suatu permasalahan karena menghalangi dalam membuat suatu keputusan yang terbaik bahkan dapat menghasilkan suatu keputusan yang buruk. Dalam dunia medis, ketidakpastian dapat menghalangi pemeriksaan yang terbaik untuk pasien dan dapat menghasilkan terapi yang keliru.

Beberapa teori ketidakpastian antara lain probabilitas klasik, probabilitas Bayes, teori Hartley yang berdasar pada himpunan klasik, teori Shanon yang didasarkan pada peluang, Teori *Dempster-Shafer* dan teori *Fuzzy Zadeh*.

### II.2.13 Metode Dempster-Shafer

Metode *Dempster-Shafer* adalah sutu teori matematika untuk pembuktikan bedasarkan belief function dan plausible reasoning ( fungsi kepercayaan dan pemikiran yang masuk akal ). Yang digunakan untuk mengkombinasikan potongan informasi yang terpisah ( bukti ) untuk mengkallkulasi kemungkinan dari suatu peristiwa.

Metode *Dempster-Shafer* pertama kali diperkenalkan oleh Dempster, yang melakukan percoban model ketidakpastian dengan range probabilitas sebagai

probabilitas tunggal. Kemudian pada tahun 1976 Shafer mempublikasikan teori Dempster tersebut pada sebuah buku yang berjudul *Mathematical Theory of Evident.* Secara umum Metode *Dempster-Shafer* ditulis dalam suatu interval : **[*Belief,Plausibility*].**

*Belief* (Bel) adalah ukuran kekuatan *evidence* dalam mendukung suatu himpunan proposisi. Jika bernilai 0 (nol) maka mengindikasikan bahwa tidak ada *evidence,* dan jika bernilai 1 menunjukkan adanya kepastian. Menurut Giarratano dan Riley fungsi *belief* dapat diformulasikan sebagai :

$$Bel(X) = \sum_{y \subseteq y} m(Y) \text{.....(2.1)}$$

sedangkan *Plausibility* (Pls) dinotasikan sebagai :

$$Pls(X) = 1 - Bel(X') = 1 \sum_{y \subseteq x'} m(X') \text{.....(2.2)}$$

Bel(X) = *Belief* (X)

Pls(X) = *Plausibility* (X)

m(X) = *mass function* dari (X)

m(Y) = *mass function* dari (Y)

*Plausibility* juga bernilai 0 sampai 1, jika kita yakin akan X' maka dapat dikatakan *Belief* (X') = 1 sehingga dari rumus di atas nilai Pls (X) = 0. Beberapa kemungkinan range antara *Belief* dan *Plausibility* adalah :

*Tabel II.1 Range Belief dan Plausibility*

| Kemungkinan | Keterangan |
|---|---|
| [1,1] | Semua Benar |
| [0,0] | Semua Salah |
| [0,1] | Ketidakpastian |
| [Bel,1] where 0 < Bel < 1 | Cenderung Mendukung |
| [0,Pls] where 0 < Pls < 1 | Cenderung Menolak |
| [Bel,Pls] where 0 < Bel ≤ Pls < 1 | Cenderung Mendukung dan Menolak |

Pada teori Dempster - Shafer juga dikenal adanya *frame of discernment* yang dinotasikan dengan . FOD ini merupakan semesta pembicaraan dari sekumpulan hipotesis sehingga sering disebut dengan *environment* , dimana:

$$\Theta = \{01,02,.....0n\}.....(2.3)$$

$\Theta$ = FOD atau environment

$$\Theta = \{01,02,.....04\}.....(2.3)$$

$01...0n$ = Elemen/unsur bagian dalam environment

*Environment* mengandung elemen-elemen yang menggambarkan kemungkinan sebagai jawaban dan hanya ada satu yang akan sesuai dengan jawaban yang dibutuhkan. Kemungkinan ini dalam teori *Dempster-Shafer* disebut dengan *power set* dan dinotasikan dengan P($^0$), setiap elemen dalam *power set* ini memiliki nilai interval antara 0 sampai 1.

### II.2.14 Perancangan Sistem

Perancangan sistem adalah diagram yang menggambarkan sistem yang sedang berjalan dan sistem baru yang akan digunakan dengan menggunakan komputer. Dalam tahap-tahap ini dilakukan pemecahan masalah secara logika dengan menggunakan alat bantu, yaitu Data Flow Diagram (DFD), Entity Relatiob Diagram (ERD) dan Normalisasi.

### II.2.15. Basis Data (*DataBase)*

Basis data (database) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. *Database* merupakan salah satu komponen yang penting dalam sistem informasi, karena merupakan basis dalam penyediaan informasi bagi para pemakai. Penerapan database dalam sistem informasi disebut *database system* (Abdul Kadir, 2005).

Sistem database adalah suatu sistem informasi yang mengintegrasi kumpulan dari data yang saling berhubungan satu dengan yang lainnya dan membuatnya tersedia untuk beberapa aplikasi yang bermacam-macam didalam suatu organisasi. (Jogiyanto HM ,1999).

Jenjang Data atau Hirarki Data terdiri dari :

1. *Database*

   Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

2. *File*

   File terdiri dari record yang menggambarkan satu kesatuan data yang sejenis. Misalnya : file mata kuliah berisi data tentang semua mata kuliah yang ada.

3. *Record*

   Kumpulan dari *field* membentuk suatu *record*. *Record* menggambarkan suatu unit data yang tertentu, Kumpulan dari record membentuk suatu file. Misalnya : file penyakit, tiap-tiap record dapat mewakili data tiap-tiap penyakit.

4. *Field*

   Suatu field menggambarkan suatu atribut dari record yang menunjukan suatu item dari data, kumpulan dari field membentuk suatu record. Ada 3 hal yang penting dalam suatu field, yaitu :

   a. Nama dari *field* (*field name*)

      Field harus diberi nama untuk membedakan field yang satu dengan field yang lainnya.

   b. Representasi dari field (*field representation*)

      Reprentasi dari field menunjukkan tipe data dari field *(field type)* serta lebar dari field (*field width*). Field dapat bertipe numerik ataupun huruf (pada paket dBASE ataupun FOXBASE+, tipe *field* dapat berupa numerik, karakter atau huruf, tanggal dan memo). Lebar dari field menunjukan ruang maksimum dari *field* yang dapat diisi dengan karakter-karakter data.

   c. Nilai dari *field* (*field value*)

      Nilai dari *field* menunjukan isi dari *field* untuk masing-masing record.

5. *Character / Byte*

   Karakter merupakan bagian data yang terkecil, dapat berupa characters numeric, huruf ataupun karakter-karakter khusus (*special characters*) yang membentuk suatu item data.

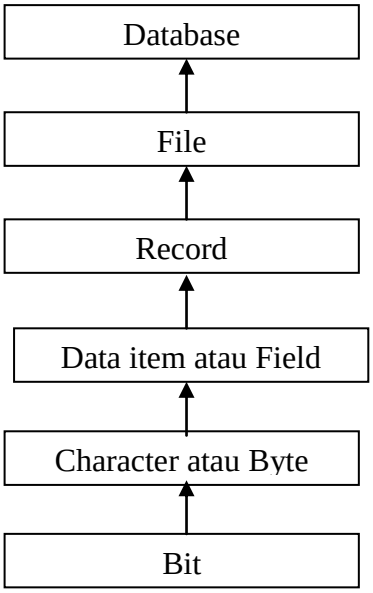
6. *Bit*

   *Bit* merupakan bagian terkecil dari data secara keseluruhan yaitu berupa karakter ASCII nol atau satu yang merupakan komponen pembentuk byte

### II.2.16. Database Management Sistem (DBMS)

*Database Management Sistem* atau disingkat DBMS adalah perangkat lunak (*Software*) yang berfungsi untuk mengelola database, mulai dari membuat database itu sendiri, sampai dengan proses-proses yang berlaku. Salah satu jenis DBMS yang sangat terkenal saat ini adalah *Relational DBMS* (RDBMS), yang merepresentasikan data dalam bentuk tabel-tabel yang saling berhubungan. Sebuah tabel disusun dalam bentuk baris (record) dan kolom (field) (Abdul Kadir, 2005).

Perkembangan perangkat lunak RDBMS ini telah banyak ditemukan, misalnya *MySQL, Oracle, Sybase, dBase, MS. SQL, Microsoft Access (MS. Access*) dan lain-lain. Pada praktek nantinya, anda menggunakan perangkat lunak MS Access versi 2002, karena perangkat lunak ini sudah termasuk kedalam paket aplikasi office dari Microsoft XP. Dan yang terpenting MS. Access (Abdul Kadir, 2005).




**Gambar II.9 Struktur Databases**
**(Analisa Sistem Informasi, Jogiyanto HM)**

Ada tiga kelompok perintah yang digunakan dalam mengelola dan mengorganisasikan data dalam RDBMS, yaitu :

***a. Data Definition Language***

Merupakan perintah-perintah yang digunakan oleh seorang Database Administrator untuk mendefinisikan struktur dari database, baik membuat tabel baru, menentukan struktur penyimpanan tabel, model relasi antar tabel, validasi data, dan lain sebagainya.

***b. Data Manipulation Language (DML)***

Perintah-perintah yang digunakan untuk memanipulasi dan mengambil data pada suatu database. Manipulasi yang dapat dilakukan terhadap data adalah :

1. Penambahan data.
2. Penyisipan data.
3. Penghapusan data.

4. Pengubahan data.

DML merupakan bahasa yang memudahkan pengguna dalam mengakses database. Ada dua jenis DML :

1. ***Prosedural***, mengharuskan pengguna untuk menentukan spesifikasi data apa yang dibutuhkan dan bagaimana cara mendapatkannya. Contoh paket bahasanya adalah dBase III, FoxBase, FoxPro.
2. ***Non Prosedural***, pengguna hanya menentukan data apa yang dibutuhkan tanpa harus tahu bagaimana cara mendapatkannya. Contoh paket bahasanya diberi nama *Structural Query Language (SQL).*

***c. Data Control Language***

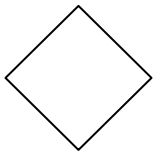
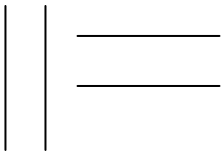
Bagian ini berkenaan dengan cara mengendalikan data, seperti siapa saja yang bisa melihat isi data, bagaimana data bisa digunakan oleh banyak user, dan lain-lain.

### II.2.17. Entitiy Relationship Diagram ( ERD )

*Entity Relationship Diagram ( ERD )* adalah sebuah data yang berbentuk hubungan yang menggunakan *primary key* ( Kunci penghubung ) untuk menyatukan data dari satu tabel ke tabel lain.

Berikut adalah daftar simbol – simbol yang digunakan dalan pembuatan *Entity Relationship Diagram ( ERD )*

Tabel II.2 : Simbol – simbol dalam ERD

| NO | SIMBOL | KETERANGAN |
|---|---|---|
| 1 |  | **Simbol table**<br>Simbol digunakan untuk mewakili suatu tabel |
| 2 |  | **Simbol kondisi**<br>Simbol digunakan untuk suatu kondisi di dalam table |
| 3 |  | **Simbol garis relasi**<br>Simbol digunakan untuk menunjukan arus relasi dari table |

*( Sumber : Hanif Al Fatta , 2007 )*

### II.2.19 Pengenalan Bahasa Pemogramnan Visual Basic 2010

.NET Platform merupakan satu set kumpulan teknologi yang memungkinkan teknologi Internet ditransformasikan ke dalam platform *distributed computing* dengan skalabilitas dan kompatibilitas tinggi. Secara teknikal, .NET Platform menyediakan konsep pemrograman dengan library dan modul-modul baru yang konsisten, terlepas dari jenis bahasa pemrograman yang digunakan. .NET Platform menyediakan hal-hal berikut bagi para developer :

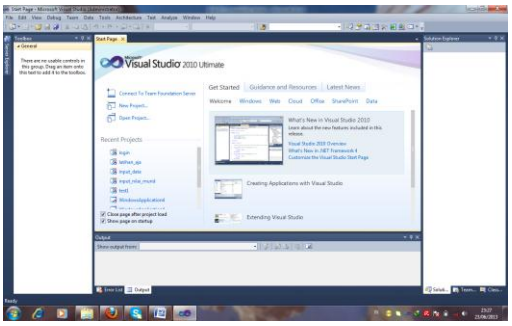
a. Language independent, dengan programming model yang konsisten di semua tier aplikasi yang dibangun.
b. Interoperability dan kompatibilitas antar aplikasi.

c. Kemudahan migrasi dari teknologi yang ada saat ini.

d. Dukungan penuh terhadap berbagai teknologi standar yang digunakan dalam platform internet, antara lain HTTP, XML, SOAP dan HTML.

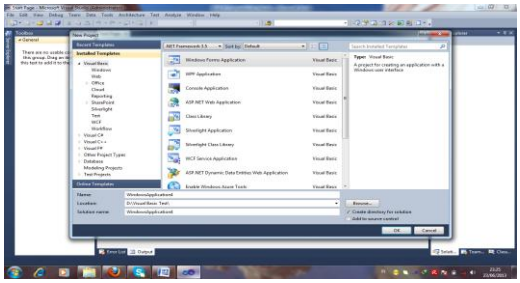
### II.2.20 Membuat Project Baru

Mulailah dengan membuka Visual Studio .NET, maka akan tampil Start Page yang menampilkan beberapa project terakhir yang anda akses. Pada bagian kiri terdapat beberapa baris hyperlink yang menghubungkan anda dengan beberapa informasi penting. Beberapa link memerlukan koneksi internet untuk mengaksesnya, seperti Online Community dan web Hosting.




**Gambar II.10.Tampilan awal VB 2010**

Untuk membuat Project baru pilih New Project untuk menampilkan dialog New Project seperti gambar II.7 di bawah ini :



**Gambar II.11. dialog New Project**

Dalam dialog New Project anda dapat memilih jenis aplikasi yang akan dibuat termasuk bahasa pemrograman digunakan. Jenis aplikasi yang dapat dibuat adalah :

a. **Windows Application** : adalah aplikasi yang paling umum dibuat, menggunakan interface windows. Biasanya Windows Application merupakan interface aplikasi sedangakan logic aplikasi terdapat di dalam Class Library. Windows Application dapat berisi form, class, XML file, maupun file VB Script dan Jscript.
b. **Class Library** : merupakan fondasi dasar untuk membuat komponen yang menjalankan fungsi tertentu. Class merupakan fondasi dasar untuk membentuk obyek dalam pemrograman berorientasi obyek. Class Library tidak memiliki interface tertentu seperti form, tetapi dapat diakses oleh aplikasi lain untuk menjalakan berbagai fungsi yang terdapat di dalamnya. Class Library dapat disamakan dengan teknologi ActiveX DLL (.dll) dan ActiveX EXE dalam pemrograman VB6.
c. **Windows Control Library** : tidak puas dengan built in control yang disediakan VS .NET ? Anda dapat berkreasi membuat kontrol sendiri dan memasukkan berbagai fungsi yang anda inginkan di dalam kontrol tersebut. Fasilitas untuk membuat kontrol tersebut adalah Windows Control Library. Kontrol ini sama dengan ActiveX Control (.ocx) dalam pemrograman VB6.
d. **ASP .NET Web Application** : adalah project yang digunakan untuk membuat aplikasi web. Teknologi yang digunakan adalah ASP .NET yang memiliki berbagai kelebihan dibandingkan ASP klasik. Perubahan utamanya adalah

dapat diprogram menggunakan berbagai bahasa .NET seperti VB, C++, C# maupun J#. ASP .NET juga menyediakan berbagai kontrol yang bersifat event drivent programming sehingga lebih menghemat waktu pembuatan aplikasi.

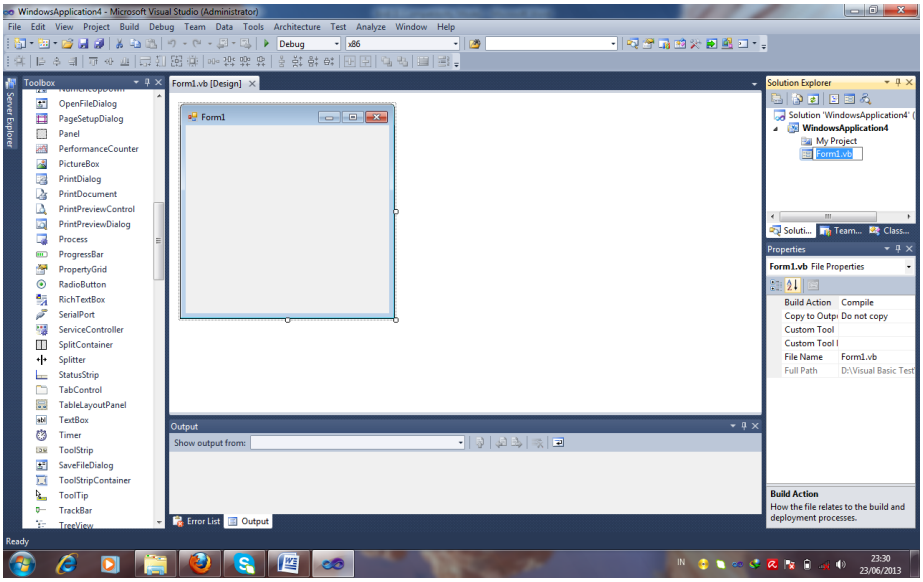
e. **ASP .NET Web Service** : Web service merupakan salah satu ide utama dalam .NET. Anda dapat membuat web service dan meletakkannya di web server untuk diakses berbagai aplikasi. Sebuah web service dapat diakses oleh aplikasi windows, web, console, maupun mobile device. Web service hampir sama dengan Class Library, perbedaan utamanya adalah web service tersebut diletakkan di web server sehingga dapat diakses dengan lebih mudah dan tidak terbatas pada aplikasi berbasis windows saja.
f. **Console Application** : merupakan aplikasi dengan tampilan text mode atau DOS. Aplikasi jenis ini biasa digunakan sebagai monitoring service atau remote application dimana sumber daya komputer dan bandwith sangat terbatas.
g. Windows **Service** : adalah aplikasi yang berjalan sebagai service di windows, yang di load bersamaan dengan proses start up windows. Aplikasi ini berjalan di background dan biasanya tidak memiliki interface. Penerapan aplikasi ini misalnya untuk pembuatan scanning antivirus, server FTP, dan remote server.
h. **Web Control Library** : Hampir sama dengan Windows Control Library tetapi digunakan untuk aplikasi web.

### II.2.21 VB .Net IDE

Gambar di bawah menjelaskan garis besar IDE yang biasa digunakan. Di bagian atas terdapat toolbar yang sudah tidak asing lagi, mencakup berbagai

fasilitas editing seperti cut, copy, paste, dan tombol Start. Di bagian kanan terdapat Solution Explorer yang menampilkan berbagai obyek dalam aplikasi seperti form, class dan component.

Di bawah Solution Explorer terdapat Properties Window yang berisi properti obyek yang sedang aktif di bagian designer. Anda dapat mengatur properti obyek di bagian ini baik dari segi tampilan maupun perilaku obyek tersebut dalam aplikasi. Selain menetapkan properti di bagian ini dapat pula mengeset properti secara run time dengan menggunakan coding. Bagian yang sering digunakan adalah Toolbox yang terdapat di sisi kiri, yang pada gambar di atas sengaja dihide untuk menghemat tempat. Gambar di bawah menampilkan Toolbox yang berisi berbagai macam obyek untuk ditempatkan di form. Form pada VB 2010 dapat dilihat pada gambar II.12 dibawah ini.




**Gambar II.12. Lembar Kerja VB 2010**

Toolbox Windows Form berisi obyek untuk mendesain form seperti TextBox, Label, CheckBox, dll. Kontrol yang mungkin paling sering anda gunakan adalah TextBox, Label, dan Button. Anda dapat mengklik ganda atau dengan drag-drop untuk mel etakkan suatu kontrol ke form.