BAB II

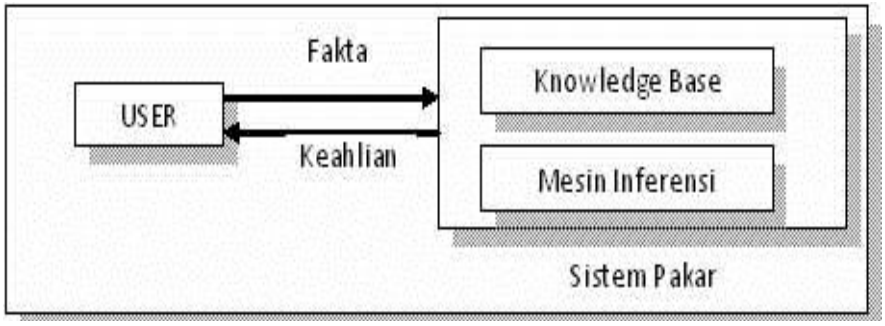
TINJAUAN PUSTAKA

II.1 Sistem Pakar

Sistem pakar adalah sistem komputer yang ditujukan untuk meniru semua aspek (emulates) kemampuan pengambilan kepurusan (decision making) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah.

Pakar atau ahli (expert) didefinisikan sebagai seorang yang memiliki pengetahuan atau keahlian yang tidak dimiliki oleh kebanyakan orang. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan oleh kebanyakan orang. Dengan kata lain, dapat memecahkan suatu masalah dengan lebih efisisen namun bukan berarti lebih murah. Pengetahuan yang dimuat ke dalam sistem pakar dapat berasal dari seorang pakar atau pun pengetahuan yang berasal dari buku, jurnal, majalah, dan dokumentasi yang dipublikasikan lainnya, serta orang yang memilki pengetahuan meskipun bukan ahli. Istilah sistem pakar (expert system), sering disinonimkan dengan sistem berbasis pengetahuan (knowledge-based system) atau sistem pakar berbasis pengetahuan (knowledge based expert system) (Rika, Rosnelly ; 2012:2-3).

Konsep dasar Sistem Pakar yaitu pengguna menyampaikan fakta atau informasi untuk sistem pakar dan kemudian menerima saran dari pakar atau jawaban ahlinya dapat dilihat pada gambar II.1 berikut ini :



Gambar II.1. Konsep Dasar Sistem Pakar
Sumber : Muhammad Arhami (2005 : 4)

Ada beberapa alasan mendasar mengapa sistem pakar dikembangkan untuk menggantikan seorang pakar, diantaranya :

1. Dapat menyediakan kepakaran setiap waktu dan di berbagai lokasi.
2. Secara otomatis mengerjakan tugas – tugas rutin yang membutuhkan seorang pakar.
3. Seorang pakar pensiun atau pergi.
4. Seorang pakar adalah mahal.
5. Kepakaran dibutuhkan juga pada lingkungan yang tidak bersahabat (*hostile environtment*).

Perbandingan sistem konvensional dan sistem pakar dapat dilihat pada Tabel II.1 berikut ini :

Tabel II.1. Perbandingan Sistem Konvensional Dan Sistem Pakar

Sistem Konvensional	Sistem Pakar
Informasi dan pemrosesan umumnya digabung dalam satu program sekuensial	Basis pengetahuan dari mekanisme pemrosesan (inferensi)
Program tidak pernah salah (kecuali programnya yang salah)	Program bisa saja melakukan kesalahan
Tidak menjelaskan mengapa input dibutuhkan atau bagaimana hasil yang diperoleh	Penjelasan (explanation) merupakan bagian dari sistem pakar
Membutuhkan semua input data	Tidak harus membutuhkan semua input data atau fakta

Perubahan pada program merepotkan	Perubahan pada kaidah dapat dilakukan dengan mudah
Sistem bekerja jika sudah lengkap	Sistem dapat bekerja hanya dengan kaidah yang sedikit
Eksekusi secara algoritmik (step by step)	Eksekusi dilakukan secara heuristic dan logis
Manipulasi efektif pada database yang besar	Manipulasi efekktif pada basis pengetahuan yang besar
Efisiensi adalah tujuan utama	Efektifitas adalah tujuan utama
Data kuantitatif	Data kualitatif
Representasi adalah numeric	Representasi pengetahuan dalam simbolik
Menangkap, menambah dan mendistribusikan data numerik atau informasi	Menangkap, menambah dan Mendistribusi pertimbangan (judgment) dan pengetahuan

Sumber : Muhammad Arhami (2005 : 8)

Tujuan dari sebuah sistem pakar adalah untuk mentransfer kepakaran yang dimiliki seorang pakar kedalam komputer, dan kemudian kepada orang lain (*nonexpert*). Aktifitas yang dilakukan untuk memindahkan kepakaran adalah :

1. *Knowledge Acquisition* (dari pakar atau sumber lainnya)
2. *Knowledge Representation* (kedalam komputer)
3. *Knowledge Inferencing*
4. *Knowledge Transfering*

II.1.1 Keuntungan Sistem Pakar

Ada beberapa keunggulan sistem pakar, diantaranya :

1. Menghimpun data 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.

Sementara kemampuan sistem pakar diantaranya adalah :

1. Menjawab berbagai pertanyaan yang menyangkut bidang keahliannya.
2. Bila diperlukan dapat menyajikan asumsi dan alur penalaran yang digunakan untuk sampai ke jawaban yang dikehendaki.
3. Menambah fakta kaidah dan alur penalaran sahih yang baru kedalam otaknya.

Selanjutnya ada banyak keuntungan bila menggunakan sistem pakar diantaranya adalah :

1. Menjadikan pengetahuan dan nasihat lebih mudah didapat.
2. Meningkatkan output dan produktivitas.
3. Menyimpan kemampuan dan keahlian seorang pakar.
4. Meningkatkan penyelesaian masalah – menerusi panduan pakar, penerangan, sistem pakar khas.
5. Meningkatkan reliabilitas.
6. Memberikan respons (jawaban) yang cepat.
7. Merupakan panduan yang intelligence (cerdas).
8. Dapat bekerja dengan informasi yang kurang lengkap dan mengandung ketidakpastian.
9. Intelligence database (basis data cerdas), bahwa sistem pakar dapat digunakan untuk mengakses basis data dengan cara cerdas.

Selain keuntungan – Keuntungan diatas, Sistem pakar seperti halnya sistem lainnya, juga memiliki kelemahan, diantaranya :

1. Masalah dalam mendapatkan pengetahuan dimana pengetahuan tidak selalu bisa didapatkan dengan mudah, karena kadangkala pakar dari masalah yang dibuat tidak ada, dan kalaupun ada pendekatan yang dimiliki oleh pakar berbeda - beda.
2. Untuk membuat suatu sistem pakar yang benar – benar berkualitas tinggi sangatlah sulit dan memerlukan biaya yang sangat besar untuk pengembangan dan pemeliharaanya.
3. Boleh jadi sistem tidak dapat membuat keputusan.
4. Sistem pakar tidaklah 100% menguntungkan, walaupun seorang tetap tidak sempurna atau tidak selalu benar. Oleh karena itu perlu diuji ulang secara teliti sebelum digunakan, dalam hal ini peran manusia tetap menjadi faktor dominan (Muhammad, Arhami ; 2005:7-11).

II.1.2 Struktur Sistem Pakar

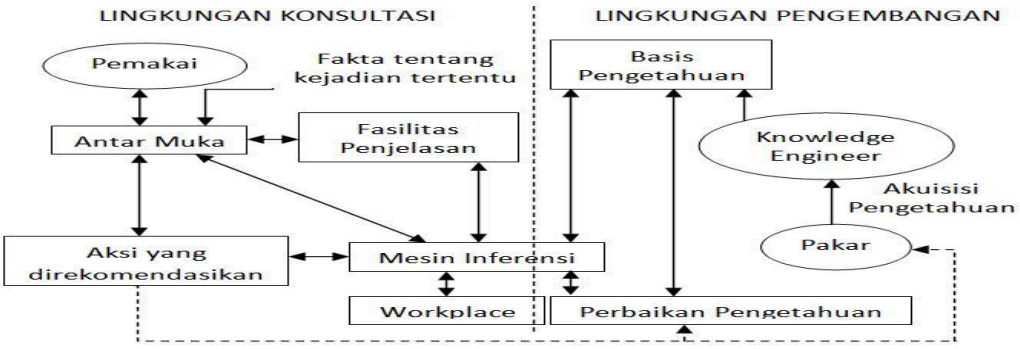
Komponen-komponen yang terdapat dalam struktur sistem pakar adalah sebagai berikut :

1. Basis Pengetahuan *(Knowledge Base*), basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi dan penyelesaian masalah. Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang obyek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

2. Akuisisi Pengetahuan, merupakan proses untuk mengumpulkan data pengetahuan terhadap suatu masalah dari sumber pengetahuan (berasal dari pakar atau media seperti majalah, buku, literatur, dll) kedalam komputer. Sumber pengetahuan tersebut dijadikan dokumentasi untuk diolah, dipelajari dan diorganisasikan menjadi basis pengetahuan.
3. Mesin Inferensi (*Inference Engine*), komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi adalah program kompoter yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace* dan untuk memformulasikan kesimpulan.
4. Antarmuka Pengguna (*User Interface),* merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi dengan penggunanya.Antarmuka menerima informasi dari pemakaia dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya kedalam bentuk yang dapat dimengerti oleh pemakai.
5. Area Kerja (*Workplace*), digunakan untuk merekam hasil – hasil antara dan kesimpulan yang dicapai. Ada 3 tipe keputusan yang dapat direkam, yaitu :
    1. Rencana : bagaimana menghadapi masalah
    2. Agenda : aksi – aksi yang potensial yang sedang menunggu untuk dieksekusi.
    3. Solusi : calon aksi yang akan dibangkitkan.

6. Fasilitas Penjelasan (*Explantion Justifier),* adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar. Komponen ini menggambarkan penelaran sistem kepada pemakai. Fasilitas penejelasan dapat menjelaskan perilaku sistem pakar dengan menjawab pertanyaan – pertanyaaan sebagai berikut :
    a. Mengapa pertanyaan tertentu ditanyakan oleh sistem pakar ?
    b. Bagaimana kesimpulan tertentu diperoleh ?
    c. Mengapa alternative tertentu ditolak ?
    d. Apa rencana untuk memperoleh penyelesaian ?
7. Pengguna *(user),* pada umunya pengguna sistem pakar bukanlah seorang pakar *(non-expert)* yang membutuhkan solusi, saran, atau pelatihan *(trainning)* dari berbagai permasalahan yang ada.
8. Perbaikan Pengetahuan (*Knowlegde Refining),* pakar memiliki kemampuan untuk menganilisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya. Kemampuan tersebut adalah penting dalam pembelajaran terkomputerisasi, sehingga program akan mampu menganalisis penyebab kesuksesan dan kegagalan yang dialaminya (Muhammad, Arhami ; 2005:13-22).

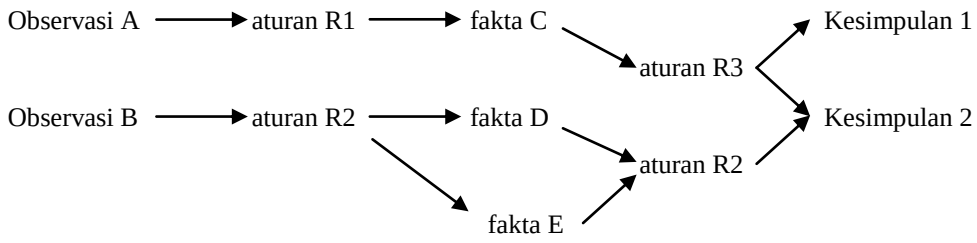
    Arsitektur dasar dari sistem pakar dapat dilihat pada gambar II.2 berikut ini:



Gambar II.2. Arsitektur Sistem Pakar
Sumber : Muhammad Arhami (2005 : 14)

II.1.3 Metode Pelacakan Kedepan *(Forward Chaining)*

Forward Chaining adalah pendekatan yang dimotori data (*data-driven*) dalam pendekatan ini pelacakan dimulai dari informasi masukan, dan selanjutnya mencoba menggambarkan kesimpulan. Pelacakan kedepan mencari fakta yang sesuai dengan bagian IF dari aturan IF-THEN.



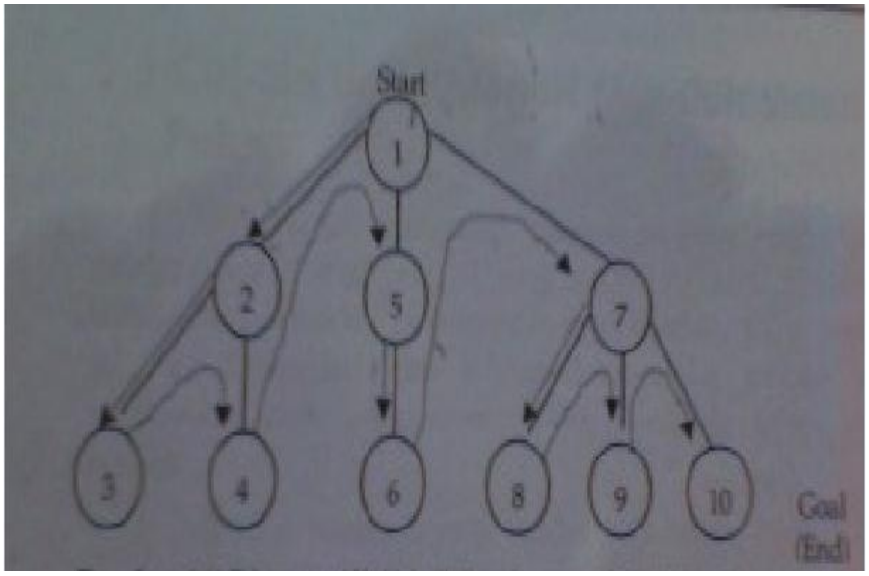
Gambar II.3. Proses Forward Chaining
Sumber : Muhammad Arhami (2005 : 20)

Metode inferensi tersebut dipengaruhi oleh tiga macam penelusuran, yaitu *Defth-first search*, *Breadth-first search* dan *Best-first search.*

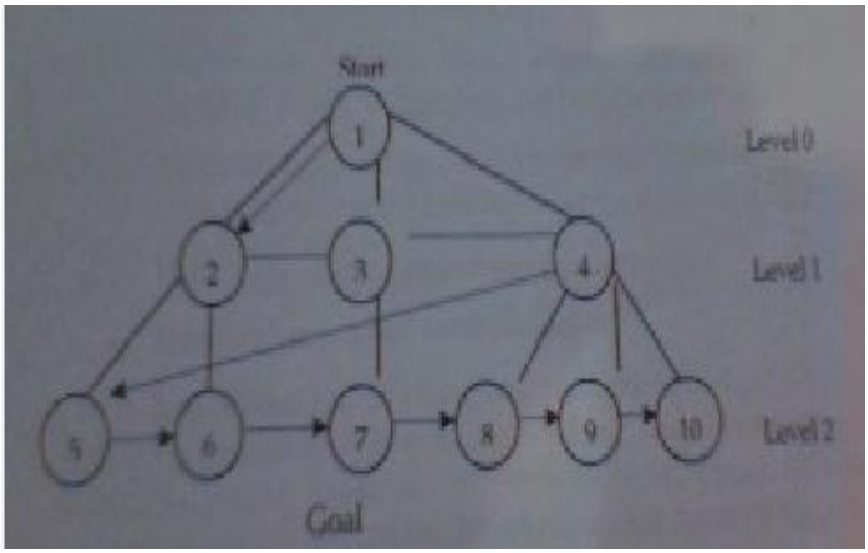
a. *Defth-first search,* melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ke tingkat dalam yang berurutan (Gambar II.4).

b. *Breadth-first search,* bergerak dari simpul akar, simpul yang ada pada setiap tingkat diuji sebelum pindah ke tingkat selanjutnya (Gambar II.5).

c. *Best-first search,* bekerja berdasarkan kombinasi kedua metode sebelumnya.



Gambar II.4. Diagram Alir Teknik Penelusuran Defth First Search
Sumber : Muhammad Arhami (2005 : 21)



Gambar II.5. Diagram Alir Teknik Penelusuran Breadth First Search
Sumber : Muhammad Arhami (2005 : 21)

II.2 Tumbuhan Liar

Pada awal peradaban manusia, semua yang ada di muka bumi ini tumbuh secara liar. Seiring dengan berjalannya waktu dan desakan kebutuhan, manusia mulai melakukan demostifikasi dan kegiatan budidaya terhadap berbagai spesies tumbuhan sehingga muncul istilah tumbuhan , tanaman, dan gulma.

Tumbuhan bermakana flora secara umum, sedangkan tanaman adalah setiap tumbuhan yang ditanam atau dibudidayakan karena manfaat dan kegunaannya yang besar bagi manusia. Meskipun demikian, bukan berarti tumbuhan yang tidak dibudidayakan tidak bermanfaat bagi manusia.

Pada dasarnya , semua tumbuhan yang berada di muka bumi ini berguna dan mempunyai manfaat. Karena Allah SWT menciptakan segala sesuatau di muka bumi ini tidak sia-sia.

Gulma secara sederhana dapat diartikan sebagai tumbuhan liar, tumbuhan pengganggu, atau tumbuhan yang tidak dikehendaki dan merugikan. Gulma dianggap merugikan karena bersaing dengan tanaman yang dibudidayakan dalam memperebutkan ruang tumbuh, unsur hara, air, dan udara. Defenisi lain gulma adalah tumbuhan yang tumbuh pada waktu, tempat, dan kondisi yang tidak diinginkan manusia. Dengan demikian, jagung yang tumbuh di pertanaman kedelai dapat dianggap sebagai gulma, karena jagung tidak ditanam secara sengaja sedangkan kedelai sebaliknya. Oleh karena itu, gulma dapat didefenisikan pula sebagai *a plant out of place* atau tumbuhan yang salah tempat.

Pengertian gulma bersifat relatif dan temporer. Manusialah yang karena kebutuhannya secara subjektif menjadikan tumbuhan menjadi gulma dan bukan gulma. Dengan kata lain, setiap orang bisa memiliki pandangan yang berbeda terhadap suatu tumbuhan dalam waktu yang sama. Contohnya, para ahli gulma menggolongkan alang-alang ke dalam gulma. Bahkan, mereka menggolongkan alang-alang ke dalam gulma ganas. Padahal bagi para herbalis, tumbuhan yang seolah-olah tak berguna tersebut begitu familiar sebagai penggempur aneka penyakit (Fauzi R. Kusuma & B. Muhammad Zaky;2005:3-5).

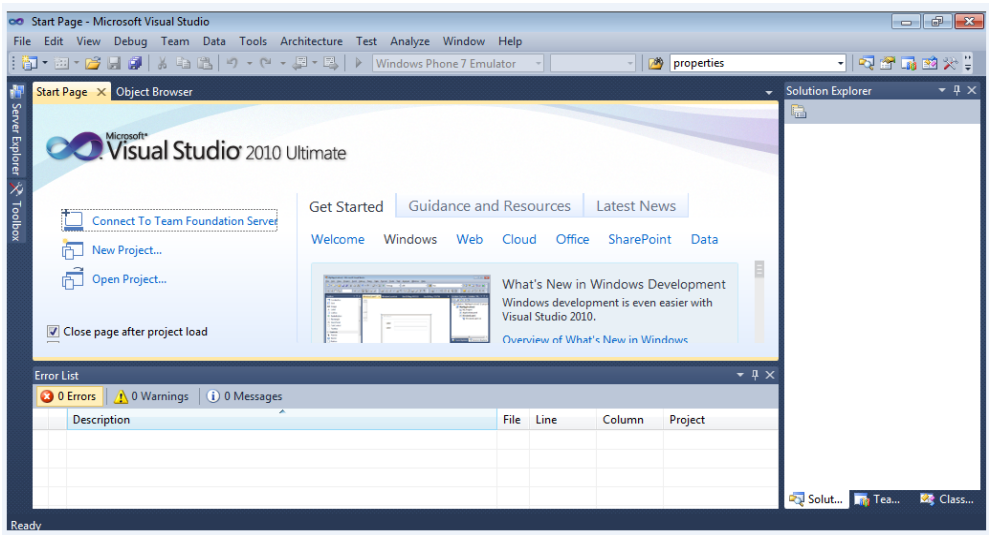
II.3 Microsoft Visual Studio 2010

Visual basic 2010 merupakan salah satu bagian dari produk pemograman terbaru yang dikeluarkan oleh Microsoft, yaitu Microsoft Visual Studio 2010. Visual Studio merupakan pruduk pemograman andalan dari Microsoft Corporation, di mana di dalamnya berisi beberapa jenis IDE pemograman seperti

Visual Basic, Visual C++, Visual Web Developer, Visual C#, dan Visual F#. Semua IDE pemograman tersebut sudah mendukung penuh implementasi .Net Framework terbaru, yaitu .Net Framework 4.0 yang merupakan pengembangan dari .Net Framework 3.5. Adapun database standar yang disertakan adalah Microsoft SQL Selver 2008 express.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu Visual Basic 2008. Beberapa pengembangan yanfg terdapat di dalamnya antara lain dukungan terhadap library terbaru Microsoft, yaitu .Net Framwork 4.0, dukungan terhadap pengembangan aplikasi menggunakan Microsoft SilverLight, dukungan terhadap aplikasi berbasis Cloud Computing, serta perluasan dukungan terhadap database-database, baik standalone maupun database selver.

Bahasa Visual Studio 2010 sendiri awalnya berasal dari bahasa pemograman yang sangat popular di kalngan programmer kompuet yaitu, bahasa BASIC, yang oleh Microsoft dadaptasi dalam program Microsoft Quick BASIC. Seiring dengan berkembangnya teknologi komputasi dan desain, Microsoft mengeluarkan produk yang dinamakan Microsoft Visual Studio dengan Visual Basic di dalamnya. Saat ini versi Microsoft Visual Studio yang beredar adalah versi 10 yang popular dengan nama Microsoft Visual Studio 2010, yang di dalamnya termasuk Microsoft Visual Basic 2010 (Wahana Komputer;2011:2-3).



Gambar II.6. Microsoft Visual Studio 2010
Sumber : Wahana Komputer (2011 : 3)

II.4 MySQL

MySQL tergolong sebagai DBMS (*Database Management System*). Perngkat lunak ini bermafaat untuk mengelola data dengan cara yang sangat fleksibel dan cepat. Berikut adalah sejumlah aktifitas yang terkait dengan data yang didukung oleh perangkat lunak tersebut.

a. Menyimpan data kedalam tabel.

b. Menghapus data dalam tabel.

c. Mengubah data dalam tabel.

d. Mengambil data yang tersimpan dalam tabel.

e. Memungkinkan untuk memilih data tertentu yang diambil.

f. Memungkinkan untuk melakukan pengaturan hak akses terhadap data.

MySQL banyak dipakai untuk kepentingan penanganan database karena selain handal juga bersifat *open source*. Konsekuensi dari *open source*, perangkat lunak ini dapat dipakai oleh siapa saja tanpa membayar dan *source code* nya boleh diunduh oleh siapa saja (Abdul Kadir, 2010; 10).

Gambar II.7. Logo MySQL
Sumber : Rulianto Kurniawan (2010 : 16)

MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial swedia yaitu MySQL AB. MySQL AB memegang penuh hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah :David Axmark, Allan Larson dan Michael "Monty" Widenius.

MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus – kasus dimana penggunaanya tidak cocok dengan penggunaan GPL (Achmad Solihin, 2010; 8)

II.5 UML

Notasi UML dibuat sebagai kolaborasi dari Grady Booch, DR.James Rumbaugh, Ivar Jacobson, Rebecca Wirfs-Brock, Peter Yourdon, dan lainya. Jacobson telah menulis tentang bagaimana mendapatkan persyaratan-persyaratan sistem dalam paket-paket transaksi yang disebut *use case*. Ia juga mengembangkan sebuah metode untuk perancangan sistem yang disebut (OOSE) yang berfokus pada analisis. Booch, Rumbaugh dan Jacobson biasa disebut dengan tiga sekawan (*tree amigos*). Ketiganya bekerja di *Rational Software Corporation* dan fokus pada standarisasi dan perbaikan ulang UML. Simbol-simbol UML mirip dengan Booch, notasi OMT, dan juga ada kemiripan dengan notasi lainya.

Penggabungan beberapa metode menjadi UML dimulai tahun 1993. Setiap orang dari tiga sekawan di rational mulai menggabungkan idenya dengan metode-metode lain yang pada saat itu ada. Akhir tahun 1995 *Unified Method* versi 0.8 diperkenalkan. *Unified Method* diperbaiki dan diubah menjadi UML pada tahun 1996, UML 1.0 disahkan dan diberikan pada *Object Technology Group* (OTG) pada tahun 1997, dan pada tahun itu juga beberapa perusahaan pengembang utama perangkat lunak mulai mengadopsinya. Pada tahun yang sama OMG merilis UML 1.1 sebagai standar industri (Sholiq ; 2010 : 18).

II.5.1 Diagram – Diagram Pada UML

UML Menyediakan beberapa diagram visual yang menunjukkan berbagai aspek dalam sebuah sistem. Banyaknya diagram tersebut dimaksudkan untuk

memberikan gambaran yang lebih terintegrasi terhadap sistem yang akan dibangun.

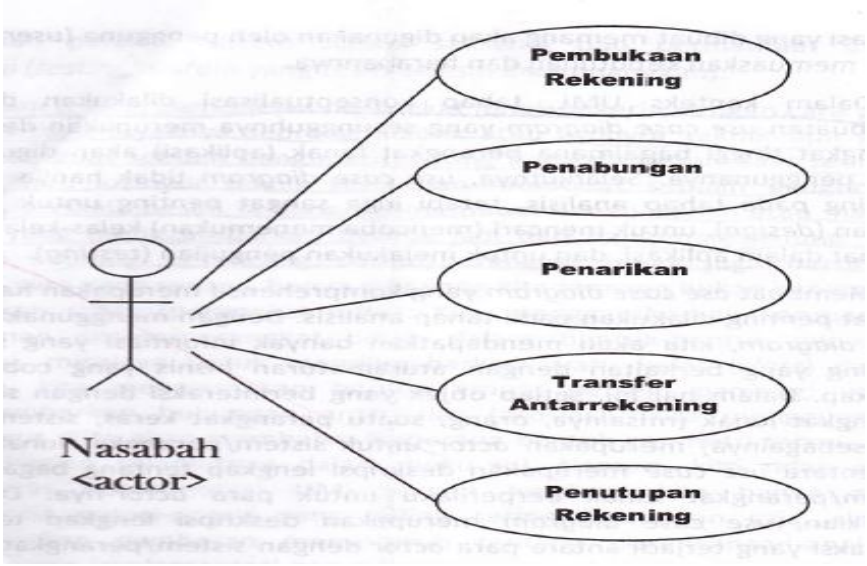
Beberapa diagram yang disediakan dalam UML antara lain :

1. Diagram Use Case

Dalam konteks UML, tahap konseptualisasi dilakukan dengan pembuatan *use case* diagram yang sesungguhnya merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunaka oleh penggunnya. Selanjutnya, *use case* diagram tidak hanya sangat penting pada tahap anlisis, tetapi juga sangat penting untuk perancangan (*design*), untuk mencari (mencoba menemukan) kelas-kelas yang terlibat dalam aplikasi, dan untuk melakukan pengujian (testing) (Adi Nugroho ; 2009 : 6-8).

Saat kita mengembangkan *use case* diagram, hal ini yang pertama kali kita lakukan adalah mengenal *actor* untuk sistem/aplikasi yang sedang kita kembangkan. Dalam hal ini, ada beberapa karakteristik untuk para *actor*, yaitu :

1. Actor ada diluar sistem yang sedang kita kembangkan
2. Acktor berinteraksi dengan sistem yang sedang kita kembangkan.



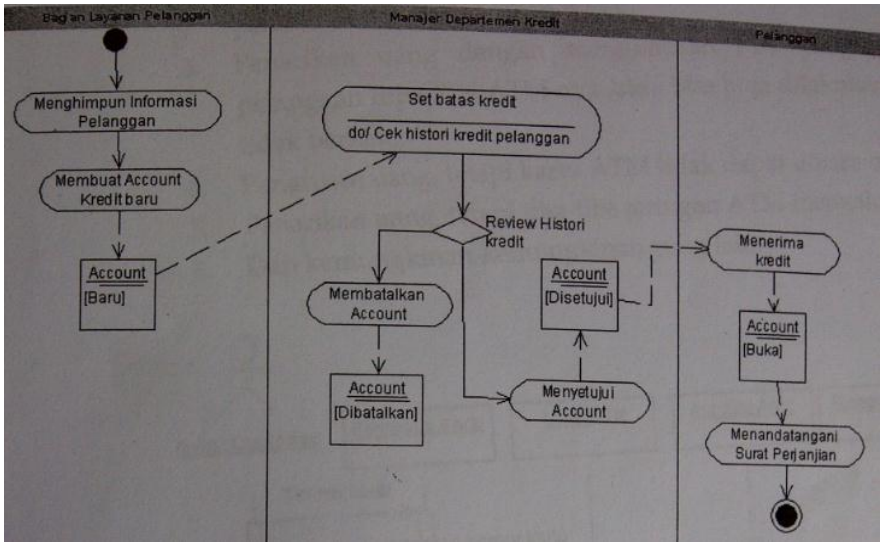
Gambar II.8. Diagram Use Case
Sumber : Adi Nugroho (2009 : 8)

2. Diagram Aktifitas

Diagram aktifitas menggambarkan aliran fungsionalitas sistem. Ada 2 kegunaan diagram aktifitas dalam permodelan dengan UML, Dua kegunaan tersebut adalah sebagai berikut :

a. Pada tahap permodelan bisnis, diagram aktifitas dapat dipergunakan untuk menunjukan alur kerja bisnis (*business workflow*).
b. Pada tahap permodelan sistem, diagram aktifitas dapat digunakan untuk menjelaskan yang terjadi dalam sebuah *use case.*

Diagram aktifitas mendefinisikan dari mana *workflow* dimulai, dimana *workflow* berakhir, aktifitas apa saja yang terjadi didalam *workflow*, dan apa saja yang dilakukan saat sebuah aktifitas terjadi. Aktifitas adalah tugas yang dilakukan selama *workflow*.



Gambar II.9. Diagram Aktifitas
Sumber : Sholic (2010 : 23)

3. Diagram Sekuensial

Diagram sekuensial (*sequence diagram*) digunakan untuk menunjukkan alur (*flows*) fungsionalitas yang melalui sebuah *use case* yang disusun dalam urutan waktu. Misalkan dalam *use case* "menarik uang " sebagaimana ditunjukan pada gambar II.11, mempunyai beberapa kemungkinan (scenario) yang bisa saja terjadi beberapa kemungkinannya sebagai berikut :

a. Penarikan uang secara normal yaitu menarik uang di ATM dengan sukses. Kemungkinan ini adalah skenario yang sering terjadi. Skenario sukses.
b. Penarikan uang, tetapi saldo tidak cukup. Bisa saja terjadi kan ? mau menarik uang 50 ribu tetapi saldo yang ada direkening tinggal

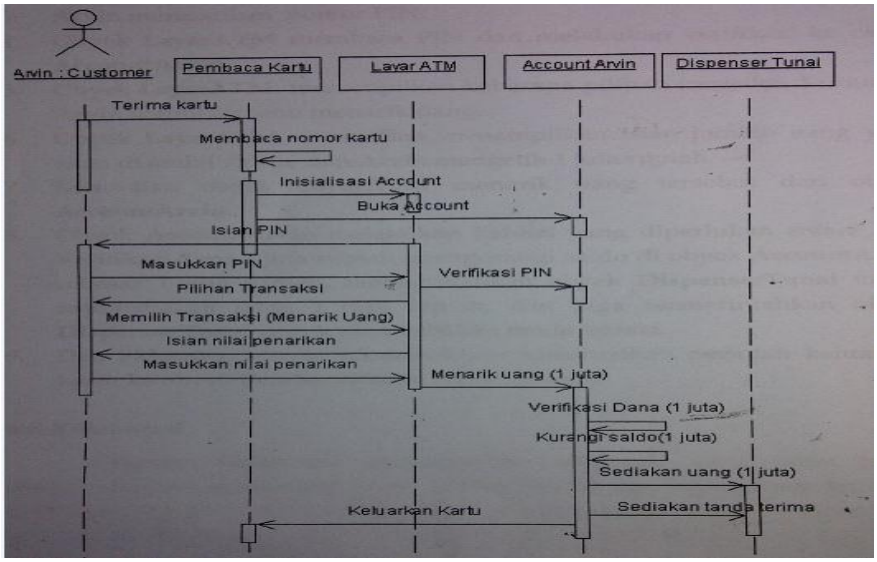
Rp.45 ribu. Barang kali pemilik ATM lupa saldonya tidak cukup untuk penarikan minimal dengan ATM.

c. Penarikan uang dengan menggunakan PIN yang salah. Bisa saja, pelanggan lupa PIN ATM-nya, atau bisa juga dilakukan orang lain yang tidak berhak.

d. Penarikan uang, tetapi kartu ATM tidak dapat dibaca oleh mesin ATM.

e. Penarikan uang, tetapi tiba – tiba jaringan ATM mengalami kerusakan.

f. Dan kemungkinan – kemungkinan yang lain.



Gambar II.10. Diagram Sekuensial Untuk Penarikan Uang
Sumber : Sholic (2010 : 24)

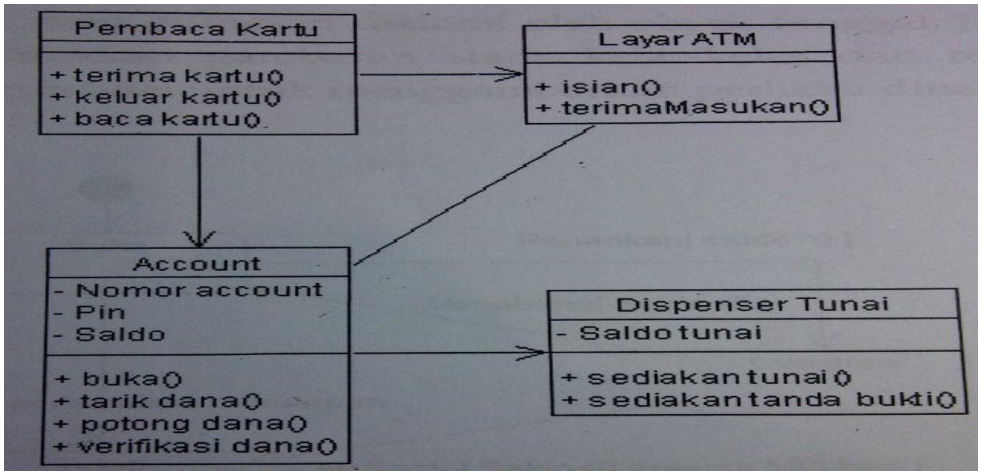
4. Diagram Kelas

Diagram Kelas menunjukkan interaksi antar kelas – kelas dalam sistem. Kelas juga dapat dianggap sebagai cetak biru dari obyek –

obyek didalam sistem. Nomor *AccountArvin* adalah sebuah obyek dari kelas Account. Sebuah kelas mengandung informasi (attribut) dan tingkah laku (behavior) yang berkaitan dengan informasi tersebut. Kelas Account mengandung attribut nomor PIN pengguna dan tingkah laku untuk mengecek PIN tersebut. Sebuah kelas pada diagram kelas dibuat untuk setiap tipe obyek pada diagram sekuensial atau diagram kolaborasi.

Gambar II.13 adalah diagram kelas yang menunjukkan hubungan antar kelas – kelas yang didapatkan dari obyek – obyek dalam diagram sekuensial atau diagram kolaborasi untuk *use case* penarikan uang.

Diagram ini terdiri atas empat kelas : Pembaca kartu, Account, Layar ATM, dan Dispenser Tunai. Kelas pembaca kartu didapatkan dari obyek pembaca kartu, kelas Account didapatkan dari obyek AccountArvin didiagram sekuensial atau diagram kolaborasi, kelas layar ATM didapatkan dari obyek layar ATM, dan kelas Dispenser Tunai dari obyek Dispenser Tunai (Sholic, 2010; 21-28).



Gambar II.11. Diagram Kelas Untuk Penarikan Uang Pada Sistem ATM
Sumber : Sholic (2010 : 28)