BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pakar

II.5.1. Pengertian Sistem Pakar

Sistem pakar merupakan salah satu bidang teknik kecerdasan buatan (*artificial intelegence*) yang cukup diminati karena penerapannya diberbagai bidang baik bidang ilmu pengetahuan maupun bisnis yang terbukti sangat membantu dalam mengambil keputusan dan sangat luas penerapanya.Ini merupakan bagian *software* spesialisasi tingkat tinggi yang berusaha menduplikasi fungsi seorang pakar dalam satu bidang keahlian.

Definisi sistem pakar yang lain yaitu suatu sistem komputer yang dirancang agar dapat melakukan penalaran seperti layaknya seorang pakar pada suatu bidang keahlian tertentu. Kecerdasan buatan dapat dilihat sebagai upaya untuk aspek model pemikiran manusia pada komputer.Hal ini juga kadang-kadang didefinisikan sebagai sebuah usaha komputer untuk memecahkan setiap masalah yang manusia dapat selesaikan dengan lebih cepat.

Ada beberapa definisi sistem pakar/ *expert system*:

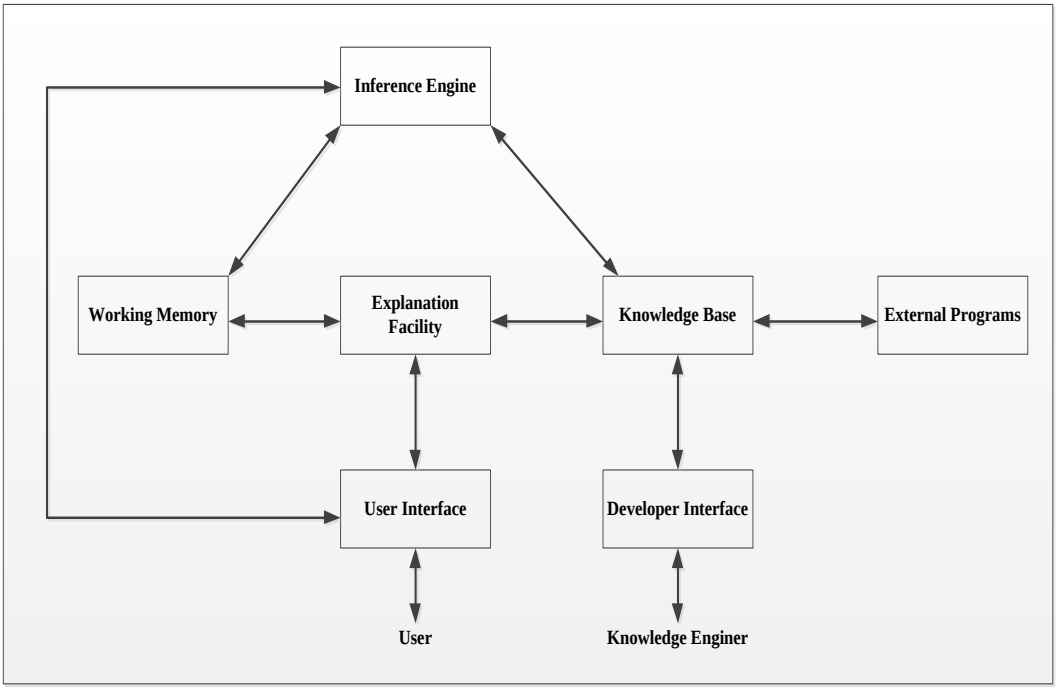
1. Menurut Arhami (2005 : 2), sistem pakar adalah suatu sistem komputer yang menyamai (*emulates*) kemampuan pengambilan keputusan dari seorang pakar.
2. Menurut Sutojo, et al. (2011 : 13), sistem pakar adalah suatu sistem yang dirancang untuk dapat menirukan keahlian seorang pakar dalam menjawab pertanyaan dan memecahkan suatu masalah.

3. Menurut Desiani dan Arhami (2006 : 227), sistem pakar merupakan sistem yang berbasis pengetahuan, yaitu meniru penalaran dari seorang pakar dalam bidang tertentu.

Dapat ditarik sebuah kesimpulan bahwa sistem pakar adalah sistem yang didesain dan diimplementasikan dengan bantuan bahasa pemrograman tertentu untuk dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli (pakar).Diharapkan dengan sistem ini, orang awam dapat menyelesaikan masalah tertentu baik yang mudah ataupun rumit sekalipun tanpa bantuan para ahli dalam bidang tersebut.

II.5.2. Arsitektur Sistem Pakar

Arsitektur sistem pakar dapat di lihat pada Gambar II.1.di bawah ini :



Gambar II.1.Arsitektur Sistem Pakar

(Sumber : Andreas Handojo, et al, 2004:34)

Adapun penjelasan dari Gambar II.1.di atas adalah sebagai berikut :

1. *Knowledge base* adalah representasi pengetahuan dari seorang ataubeberapa pakar yang diperlukan untuk memahami, memformulasikan danmemecahkan masalah. Dalam hal ini digunakan untuk memecahkanmasalah-masalah yang terjadi pada sistem. *Knowledge base* ini terdiri daridua elemen dasar, yaitu fakta dan *rule*.
2. *Inference engine* merupakan otak dari sistem pakar yang mengandungmekanisme fungsi berpikir dan pola-pola penalaran sistem yangdigunakan oleh seorang pakar. Mekanisme ini yang menganalisis suatumasalah tertentu dan kemudian mencari solusi atau kesimpulan yangterbaik.
3. *Working Memory* merupakan tempat penyimpanan fakta-fakta yangdiketahui dari hasil menjawab pertanyaan.
4. *User/developer interface*. Semua *software* pengembangan sistem pakarmemberikan *interface* yang berbeda bagi *user* dan *developer*. *User* akanberhadapan dengan tampilan yang sederhana dan mudah sedangkan*developer* akan berhadapan dengan editor dan *source code* waktumengembangkan program.
5. *Explanation facility* memberikan penjelasan saat mana *user* mengetahuiapakah sistem yang diberikan sebuah solusi.
6. *External programs*. Berbagai program seperti *database, spreadsheets, algorithms*, dan lainnya yang berfungsi untuk mendukung sistem (Andreas Handojo, et.al , 2004 : 34).

II.5.3. Ciri-Ciri Sistem Pakar

Menururt Sutojo, et al. (2011:162), ciri-ciri sistem pakar adalah sebagai berikut :

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

II.5.4. Manfaat Sistem Pakar

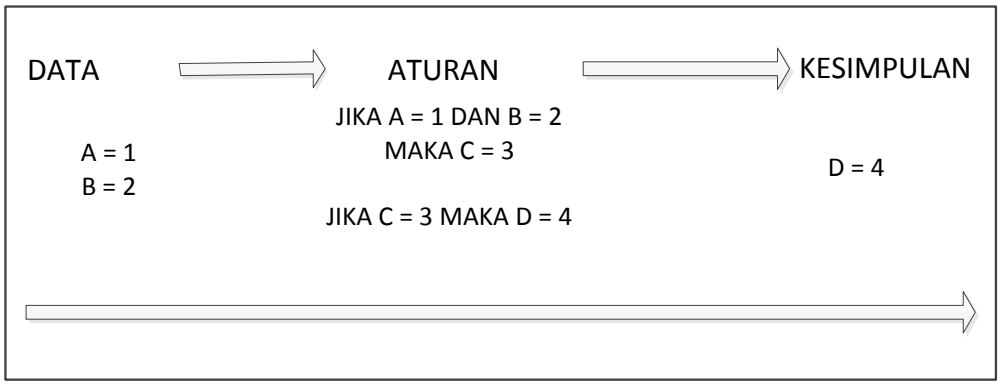
Menurut Sutojo, et al. (2011:160), sistem pakar memiliki banyak manfaat, diantaranya adalah sebagai berikut :

1. Meningkatkan produktivitas, karena sistem pakar dapat berkerja lebih cepat daripada manusia.
2. Membuat seseorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kulitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.

5. Memudahkan akses pengetahuan seorang pakar.
6. Sistem pakar tidak pernah menjadi bosan dan kelelahan atau sakit.
7. Meningkatkan kapabilitas sistem komputer.
8. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti.
9. Bisa digunakan sebagai media pelengkap dalam pelatihan.
10. Meningkatkan kemampuan untuk menyelesaikan suatu permasalahan.

II.2. Metode *Forward Chaining*

Forward Chaining adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencocokkan fakta – fakta tersebut dengan bagian *IF* dari aturan *IF-THEN* (Sutojo, et al., 2011 : 171). Bila fakta yang cocok dengan bagian *IF*, maka aturan tersebut dieksekusi.Bila sebuah aturan dieksekusi, maka sebuah fakta baru (bagian *THEN*) ditambahkan ke dalam database.Setiap kali pencocokkan, dimulai dari aturan teratas.Setiap aturan hanya boleh dieksekusi sekali saja. Proses pencocokkan berhenti bila tidak ada lagi aturan yang bisa dieksekusi.



Gambar II.2. Metode *Forward Chaining*

(Sumber : Kusrini, 2006:36)

Beberapa kelebihan dari metode *Forward Chaining* ini adalah sebagai berikut :

1. Dapat menghasilkan informasi baru dari jumlah data yang relatif sedikit.
2. Merupakan pendekatan yang baik untuk masalah terntentu seperti perencanaan, pengawasan, pengaturan dan interpretasi.
3. Dapat bekerja baik dengan permasalahan yang membutuhkan informasi lebih dulu baru kemudian menarik kesimpulan.

II.3. Virus Komputer

Perkembangan virus komputer sudah terjadi sekitar tahun 1983.Menurut catatan, pada tahun tersebut telah ada virus yang bernama *Elk Cloner* yang menyerang *Apple* dengan metode penyebaran melalui *Floppy Disk* (S'to, 2010:122).Virus merupakan sebuah program yang mampu merusak sistem komputer khususnya pada sistem operasi *windows*. Komputer yang sudah terinfeksi oleh virus akan memperlambat kinerja komputer ataupun melakukan pencurian data dari komputer target untuk dikirimkan pada komputer si pemilik virus.

Virus juga mampu bermutasi, berevolusi dengan berganti bentuk secara dinamis yang membuatnya lebih tahan terhadap pemberantasan oleh antivirus. Untuk menghindari serangan dari antivirus, virus akan memanfaatkan fitur keamanan yang terdapat pada sistem operasi ataupun akan melakukan enkripsi terhadap dirinya sendiri.

II.4.1. Karakteristik Virus Komputer

Karakteristik Virus umumnya mempunyai struktur yang hampir sama, dan dapat dibedakan menjadi beberapa kode, yaitu : kode penanda virus, kode penggandaan virus, kode pertahanan dan penyembunyian deteksi, kode pemicu, dan kode manipulasi (Fatkhiyah, M.Isa, 2012:163). Kode penanda virus, setiap virus pasti mempunyai identitas masing-masing.Bisa dibuat dengan karakter atau jumlah byte tertentu sebagai marker sesuai dengan keinginan si pembuat. Contoh, virus A mempunyai penanda X, dan virus B mempunyai penanda Y, maka virus-virus tersebut akan dikenali antivirus sesuai penandanya. Kode penggandaan virus, suatu program tidak dapat dikatakan virus jika tidak dapat menggandakan dirinya.

Banyak cara atau jurus untuk menggandakan diri yang digunakan oleh virus-virus sekarang ini. Kode pertahanan dan penyembunyian deteksi, kode ini diperlukan virus untuk mengecoh antivirus, bisa dengan mengenkripsi file virus tersebut, menyembunyikan proses kerja pada komputer target, ataupun menampilkan pesan pengalihan ketika *user* mencoba menjalankan program antivirus. Kode pemicu, setiap virus mempunyai program atau kode untuk mengaktifkan program utamanya. Program atau kode ini bias dipicu dengan bermacam-macam cara, contohnya virus diaktifkan ketika *user* membuka file tertentu pada jendela *explorer*. Atau dengan memakai nama file yang sedang populer dan menarik perhatian *user* untuk menjalankan file tersebut. Kode manipulasi, kode ini berguna untuk menghapus file, menjalankan aplikasi tertentu untuk mencuri dan mengirimkan data ke sebuah *email*.Batasan manipulasi

terserah kepada pembuatnya, karena hal inilah maka virus dikategorikan dalam program yang bersifat merusak.

II.4.2. Jenis-Jenis Virus Komputer

Saat ini jenis-jenis virus sudah menyebar hampir keseluruh penjuru dunia tidak terkecuali Indonesia.Bahkan Indonesia merupakan salah satu pembuat virus (*Virues Maker*) yang sangat terkenal, karena virus-virus yang berasal dari Indonesia umumnya mampu menggandakan dan mempertahankan diri.Jenis virus saat ini yang paling banyak beredar di Indonesia adalah virus *Worm*.Selain itu masih banyak jenis virus lainnya yang mengancam pada pengguna teknologi, khususnya komputer. Berdasarkan file yang di infeksi, virus dapat dibedakan menjadi (S'to, 2010:126-127):

1. Virus *Boot Sector*

   Sebuah komputer terinfeksi oleh *boot sector* virus jika komputer tersebut di-*boot* atau di-*re-boot* dari *floppy disk* yang telah terinfeksi oleh virus jenis ini. *Boot sector* virus cenderung tidak menyebar melalui jaringan komputer, dan biasanya menyebar akibat ketidaksengajaan penggunaan *floppy disk* yang telah terinfeksi.

2. Virus File

   Virus jenis ini akan menginfeksi file-file yang bias dieksekusi sehingga ketika program yang terinfeksi dijalankan, virus juga akan ikut dijalankan. Contoh dari file yang bisa dieksekusi ini adalah *.com, *.exe, dll.

3. Virus *Macro*

Virus ini memanfaatkan perintah-perintah yang bisa dijalankan oleh sebuah dokumen atau yang sering dikenal dengan perintah *macro*. Sebagai contoh, dokumen seperti *Microsoft Excel, Word,* dan *Access* merupakan file favaorit tempat virus ini bermukim.

4. Virus *Source Code*

   Virus jenis ini sangat unik karena metode penyebarannya dilakukan melalui sumber kode program yang ditemukan. Virus akan menambahkan kode virus kedalam sumber kode yang ditemukan dan ketika target mengkompilasi sumber kode tersebut, secara otomatis virus akan menyatu dengan programnya.

5. Virus *Network*

   Virus ini memanfaatkan jaringan yang ada untuk menyebarkan dirinya.Karena penggunaan jaringan yang sudah sangat umum, metode ini sangatlah efektif untuk mendapatkan target yang sebanyak-banyaknya.

II.4. *Unfied Modeling Languange* (UML)

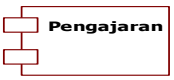
Unified Modeling Language (UML) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek (Adi Nugroho, 2010:6).Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan

mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UMLjuga merupakan salah satu alat yang sangat handal di dunia pengembangan system yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

Ada beberapa pengklasifikasi (*Classifer*) dan notasi dalam UML (Adi Nugroho, 2010:16) yang ditunjukkan pada Tabel II.1. dibawah ini.

Tabel II.1. Pangklaisfikasi dan Notasi UML

Pengklasifikasi	Kegunaan	Notasi
Actor	Menggambarkan semua objek diluar sistem yang berinteraksi dengan sistem yang dikembangkan.	
Use Case	Menggambarkan fungsionalistis yang dimiliki sistem.	
Kelas (*Class*)	Menggambarkan konsep dasar pemodelan sistem.	
Subsistem (*Subsystem*)	Menggambarkan paket spesifikasi serta implementasi.	
Komponen (*Component*)	Menggambarkan bagian-bagian fisik sistem/perangkat lunak yang dikembangkan.	
Antarmuka (*Interface*)	Menggambarkan antarmuka pengiriman pesan (*message*) antar pengklasifikasi.	

Simpul (*Node*)	Menggambarkan sumber daya komputasional yang digunakan oleh sistem.	

(Sumber : Adi Nugroho, 2010:16)

Relasi-relasi antar pengkalsifikasi yang dikenali UML adalah asosiasi, generalisasi, aliran, dan berbagai jenis kebergantungan termasuk didalamnya realisasi dan penggunaan.Untuk lebih jelasnya dapat dilihat pada Tabel II.2.berikut ini :

Tabel II.2. Relasi-relasi dalam UML

Relasi	Fungsi	Notasi
Asosiasi (*Association*)	Mendeskripsikan hubugan antar instance suatu kelas.	———
Kerbergantungan (*Dependency*)	Relasi antar dua elemen model.	- - - - - - - - >
Aliran (*Flow*)	Relasi antar dua versi suatu model.	- - - - - - - - >
Generalisasi (*Generalization*)	Relasi antar pengkalisifikasi yang memiliki deskripsi yang bersifat lebih umum dengan berbagai pengklasifikasi yang lebih spesifik, digunakan dalam struktur pewarisan.	———
Realisasi (*Realization*)	Relasi antara spesifikasi dan implementasinya.	- - - - - - - - ▸
Penggunaan (*Usage*)	Situasi di mana salah satu elemen membutuhkan elemen yang lainnya agar dapat berfungsi dengan baik.	- - - - - - - - >

(Sumber : Adi Nugroho, 2010:23)

II.5.1. *Use Case Diagram*

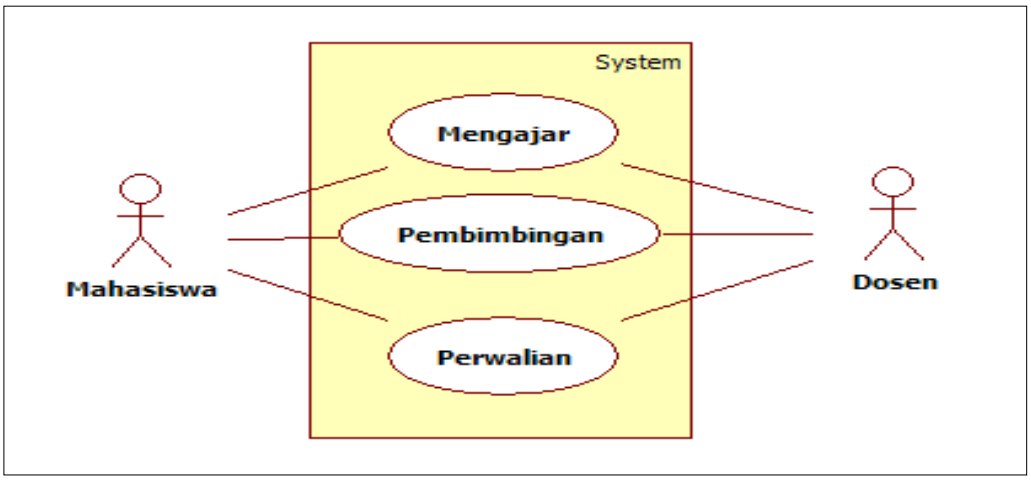
Use Case Diagram adalah fungsionalitas atau persyaratan – persyaratan sistem yang harus dipenuhi oleh sistem yang akan dikembangkan tersebut menurut pandangan pemakai sistem (Sholiq, 2010:21). *Use case* diagram juga dapat diartikan sebagai urutan langkah-langkah yang secara tindakan saling terkait

(skenario), baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk *ellips/oval*.

Untuk membentuk suatu *use case diagram*, maka diperlukan beberapa kompenen. Adapun beberapa komponen pembentuk *use case diagram* adalah sebagai berikut :

1. Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam sistem.
2. *Use case*, aktivitas/sarana yang disiapkan oleh bisnis/sistem.
3. Hubungan (*Link*), aktor mana saja yang terlibat dalam *use case* ini.

Adapun gambar *Use Case Diagram* dapat di lihat pada Gambar II.3.sebagai berikut :



Gambar II.3. Contoh *Use Case Diagram*

(Sumber : Adi Nugroho, 2010:34)

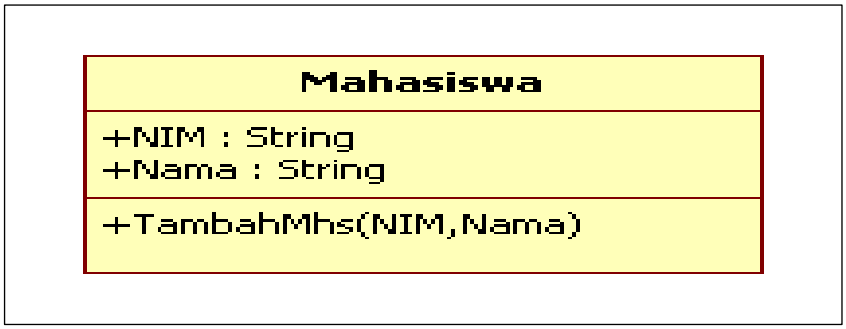
II.5.2. *Class Diagram*

Class Diagram menunjukkan interaksi antar kelas – kelas dalam sistem.Sebuah kelas mengandung informasi dan tingkah laku (*behavior*) yang berkaitan dengan informasi tersebut.*Class diagram* sesungguhnya merupakan deskripsi dari konsep yang datang dari ranah aplikasi atau solusi aplikasi (Adi

Nugroho : 2010 :11).Oleh karena itu pengertian kelas sangat penting sebelum merancang diagram kelas. Kelas diagram sebagai satu set objek yang memiliki atribut dan prilaku yang sama.

Kelas kadang-kadang disebut kelas objek (*object class*). Secara alami, objek yang berupa buku analisis disain dan buku pemrograman terstruktur kita kelompokan dalam satu kelas, yaitu kelas buku.kedua objek memiliki atribut dan prilaku yang serupa. Contohnya, kedua objek mungkin memilki atribut yang serupa seperti nomor ISBN, judul, tanggal penerbitan, edisi, dan sebagainya. Demikian juga, kedua objek memilki perilaku yang sama misalnya membuka dan menutup.

Adapun contoh*classdiagram* dapat di lihat pada Gambar II.4. sebagai berikut :



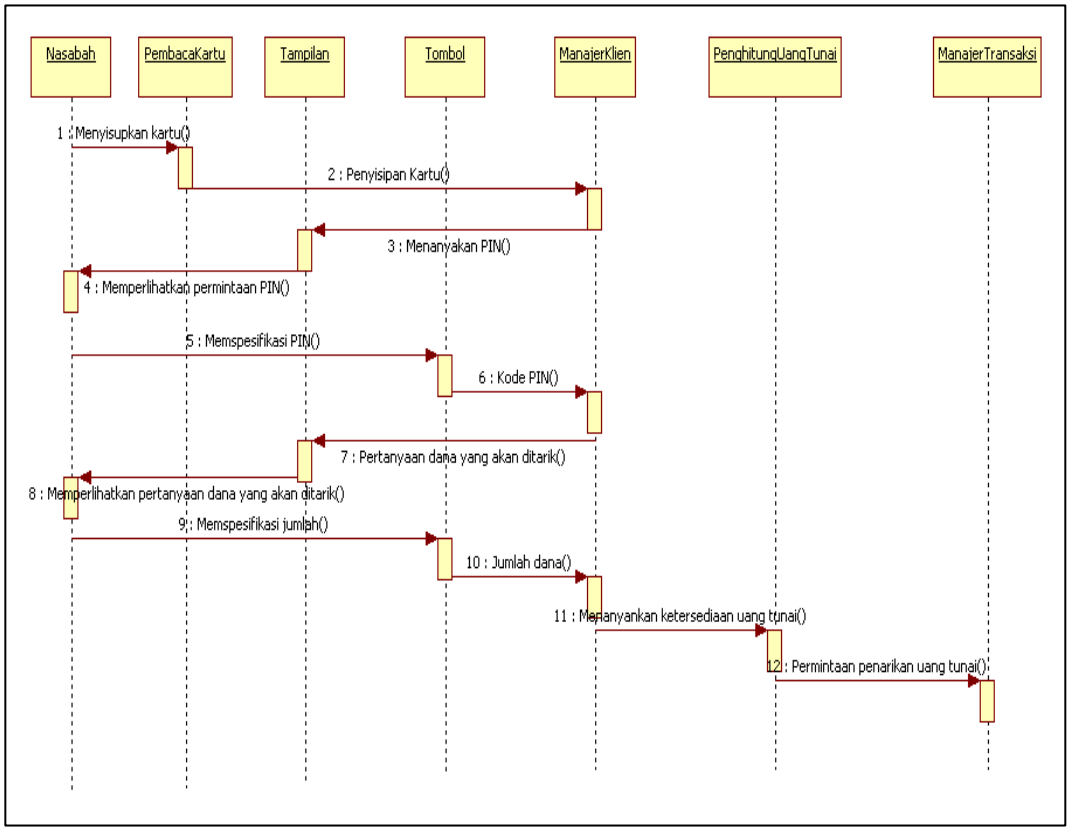
Gambar II.4. Contoh *Class Diagram*

(Sumber : Abdurohman, et,al:2010:19)

II.5.3. *Sequence Diagram*

Sequence Diagram memeperlihatkan interaksi sebagai diagram dua mantra (dimensi). Mantra vertical adalah sumbu waktu; waktu bertambah dari atas ke bawah sedangkan mantra horizontal memperlihatkan pengklasifikasi yang mempresentasikan objek-objek mandiri yang terlibat dalam kolaborasi.*Diagram*

sequence merupakan diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu (Sulistyorini, 2009:24). *Sequence Diagram* digunakan untukmenggambarkan perilaku pada sebuah skenario.Diagram ini menunjukkan sejumlah contoh objek dan *messege* (pesan) yang diletakkan diantara objek-objek ini dalam *use case*.



Gambar II. 5. Contoh *Sequence Diagram*

(Sumber : Adi Nugroh, 2010:109)

Ada beberapa komponen yang terdapat pada *sequence diagram*, yaitu :

1. Objek*/Participant*

   Objek diletakkan di dekat bagian atas diagram dengan urutan dari kiri ke kanan. Objek ini diatur dalam urutan guna menyederhanakan diagram. Setiap *participant* terhubung dengan garis titik-titik yang disebut dengan *lifeline*.

Sepanjang *lifeline* ada kotak yang disebut *activation*. *Activation* mewakili sebuah eksekusi operasi dari *participant*. Panjang kotak ini berbanding lurus dengan durasi *activation*.

2. *Messege*

   Sebuah *messege* bergerak dari satu *participan*t ke *participant* yang lain dan dari satu *lifeline* ke *lifeline*. Sebuah *participant* bisa mengirim sebuah *messege* kepada dirinya sendiri. Jika sebuah *participant* mengirimkan sebuah *messege synchronous*, maka jawaban atas *messege* tersebut akan ditunggu sebelum diproses dengan urusannya. Namun jika *messege synchronous*yang dikirimkan, maka jawaban atas *messege* tersebut tidak perlu ditunggu.

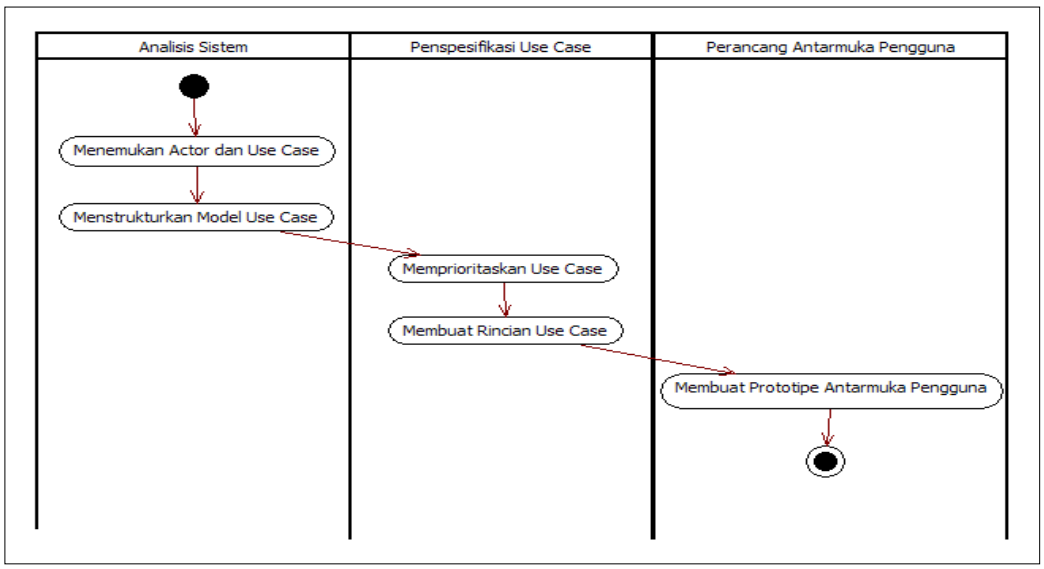
3. *Time*

   Time adalah diagram yang mewakili waktu pada arah *vertical*. Waktu dimulai dari atas ke bawah. *Messege* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibanding *messege* yang lebih dekat ke bawah.

II.5.4. *Activity Diagram*

Activity Diagram mendefinisikan dari mana *workflow* dimulai, di mana *workflow* berakhir, aktivitas apa saja yang terjadi di dalam*workflow*, dan apa saja yang dilakukan saat sebuah aktivitas terjadi (Sholiq, 2010:22). Aktivitas adalah tugas yang dilakukan selama dalama *workflow*.*Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa.

Adapun contoh*activitydiagram* dapat di lihat pada Gambar II.6. sebagai berikut :



Gambar II.6. Contoh *Activity Diagram*

(Sumber : Adi Nugroho, 2010:141)

II.5. *Microsoft Visual Basic .Net*

Visual Basic merupakan salah satu bahasa pemograman yang andal dan banyak digunakan oleh pengembang untuk membangun berbagai macam aplikasi *windows* (Wahana Komputer, 2010:2). Net mempertahankan kemudahan dan kesederhanaan dari VB versi sebelumnya ditambah dengan kemampuan berorientasi obyek yang mengikuti keandalan C++. Kemampuan berorientasi obyek diantaranya mendukung abstraksi, *enkapsulasi*, *inheritance*, *constructors*, *polymorphism*, dan *overloading*.

Bahasa pemrograman *Visual Basic* tidaklah hanya identik dengan *Visual Basic* saja. Sistem Pemrograman *Visual Basic* dalam bentuk Edisi Aplikasi, telah dimasukkan ke dalam *Microsoft Excel*, *Microsoft Access*, dan banyak aplikasi Windows lainnya juga menggunakan bahasa yang sama. *Visual Basic Scripting Edition (VBScript)* adalah sebuah bahasa skrip yang digunakan secara lebih umum

dan merupakan bagian dari bahasa *Visual Basic*. Dengan mempelajari *Visual Basic*, maka Anda akan dibawa ke area-area yang telah disebutkan tadi.

*Service-service*atau pelayanan yang terdapat pada *Microsoft Visual Basic .NET*, antara lain:

1. Sebuah model pemrograman yang memungkin *developer* membangun XML *Web Service* dan aplikasinya.
2. Sekumpulan XML *Web Service* seperti *Microsoft .NETMy Service* yang membantu pengembang menghasilkan aplikasi yang mudah dan terpadu.
3. Sekumpulan *server*, termasuk *Windows* 2000 dan 2003, *SQL Server* 2005 64 Bit, yang memadukan,menjalankan, dan mengoperasikan, serta menangani XML *Web Services* dan aplikasinya.
4. *Tool* seperti *Visual Basic .NET* untuk membangun XML *Web Service* dan aplikasi untuk *window* dan *web.*
5. Peranti lunak klien seperti *Windows* XP dan *Windows Vista*.

II.6. *Microsoft SQL Server*

Pada dasarnya pengertian dari *SQL Server* itu sendiri adalah bahasa yang dipergunakan untuk mengakses data dalam basis data relasi.Bahasa ini secara defakto adalah bahasa standar yang digunakan dalam manajemen basis data relasional.Saat ini hampir semua *server* basis data yang ada mendukung bahasa ini dalam manajemen datanya.*SQL server* 2008 R2 merupakan salah satu produk dari *Relational Database Management System* (RDBMS).

SQL Server adalah hasil kerja sama antara *Microsoft* dengan *Sybase* untuk memproduksi sebuah *software* penyimpan data (*database*) yang bekerja pada

sistem operasi OS/2. Sistem Operasi merupakan OS/2 merupakan sistem operasi baru dari hasil kerja sama antara *Microsoft* dengan IBM.

SQL Server memiliki beberapa komponen dan layanan seperti *Relational Database Engine*, *Analysis Services*, *Data Transformation Services*, *Notification Sevices*, *Reporting Services dan Service Broker* (Santoso, 2005:18).

II.7. Basis Data

II.8.1. Pengertian Basis Data

Basis data adalah suatu susunan/kumpulan data operasional lengkap dari suatu organisasi/perusahaan yang diorganisasikan/dikelola dan disimpan secara terintegrasi dengan menggunakan metode tertentu menggunakan komputer shingga mampu menyediakan informasi optimal yang diperlukan pemakainya. Sistem basis data adalah suatu sistem menyusun dan mengelola *record-record* menggunakan komputer untuk menyimpan dan merekam serta memelihara data operasional lengkap sebuah organisasi/perusahaan sehingga pemakai mampu menyediakan informasi yang optimal yang diperlukan pemakai untuk proses mengambil keputusan (Marlinda, 2004:1).

II.8.2. Komponen Dasar Sistem Basis Data

Sistem basis data terdiri dari beberapa komponen yang saling berhungan satu sama lain. Adapun komponen-komponen dasar pada sistem basis data adalah sebagai berikut (Marlinda, 2004:2):

1. Data

Data di dalam sebuah basis data dapat disimpan secara terintegrasi dan dapa pula dipakai secara bersama-sama. Terdapat tiga jenis data, yaitu :

a. Data Operasional
b. Data Masukkan
c. Data Keluaran

2. *Hardware* (Perangkat Keras)

   Terdiri dari semua peralatan komputer yang digunakan untuk pengelolaan sistem basis data.

3. *Software*(Perangkat Lunak)

   Berfungsi sebagai perantara antara pemakai dengan data fisik pada basis data.Perangkat lunak dalam basis data berupa *Database Management System* (DBMS) atau program aplikasi prosedur.

4. *User* atau Pemakai

   Pemakai basis data dibagi atas tiga klasifikasi, yaitu :

   a. *Database Administrator* (DBA)

      Database Administartor (DBA) adalah orang atau tim yang bertugas untuk mengelola sistem basis data secara keseluruhan. Adapun tugas dari DBA adalah sebagai berikut :

      1) Mengontrol DBMS dan *software-software*.
      2) Memonitor siapa saja yang mengakses basis data.
      3) Mengatur pemakaian basis data.
      4) Memeriksa keamanan, *integrity, recovery* dan *concurrency*.

   b. *Programmer*

Programmer adalah orang atau tim yang bertugas membuat program aplikasi, misalnya untuk perbankan, administrasi dan lain-lain.

c. *End User*

End User adalah orang yang mengakses basis data melalui terminal dengan menggunakan *query language* atau program aplikasi yang telah dibuat oleh *programmer*.

II.8.3. Normalisasi

Normalisasi adalah proses pengelompokan atribut-atribut dari suatu relasi sehingga membentuk *well structure relation* (Marlinda, 2004:115). Normalisasi juga merupakan proses pengelompokan elemen data menjadi tabel-tabel yang menunjukkan entitas dan relasinya.Teori normalisasi secara umum merupakan satu set peraturan yang membenarkan perkara pangkalan data menganal pasti kes-kes perkumpulan data yang tidak memuaskan dan menentukan hubungan yang boleh ditukar menjadi bentuk yang lebih cekap. Untuk menggunakan normalisasi yang baik, pangkalan data mestilah mengetahui maksud data-data.Terdapat enam bentuk normal (Marlinda, 2004:122-123). Namun demikian, dalam pemaparan berikut hanya akan dijelaskan bentuk normal pertama, kedua dan ketiga.

1. Bentuk Tidak Normal (*Unnormalized Form*)

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan untuk mengikuti suatu format tertentu. Dapat saja data tidak lengkap atau terduplikasi. Data tersebut dikumpulkan apa adanya.

2. Bentuk Normal Pertama (1NF)

Suatu relasi 1NF jika dan hanya jika sifat dan setiap relasi atributnya bersifat atomik.Atomik bermaksud tidak berkepunyaan untuk berada dalam keadaan satu bagian. Ciri-ciri bentuk normal pertama, yaitu :

a. Setiap data dibentuk dalam *flat file*.
b. Tidak ada *set* atribut yang berulang atau bernilai ganda.
c. Tiap *field* hanya satu pengertian.

3. Bentuk Normal Kedua (2NF)

   Bentuk normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk normal pertama.Atribut bukan kunci haruslah bergantung secara fungsi pada *primary key*.Peraturan ini menentukan kebergantungan sepenuhnya.Beberapa sumber teks menjelaskan sebagai kebergantungan secara fungsi dan transitif.

4. Bentuk Normal Ketiga (3NF)

   Satu hubungan dikatakan dalam bentuk 3NF jika dan hanya jika ia dalam bentuk 2NF dan setiap atribut tanpa kekunci pula bergantungan secara tidak transitif dengan kekunci primer. Kebergantungan transitif mengenal pasti sama ada atribut-atribut tanpa kekunci dalam satu hubungan juga bergantung secara fungsi terhadap suatu lagi atribut tanpa kekunci, apabila kedua-duanya bergantung secara fungsi pada kekunci primer karena berada dalam bentuk 2NF. Kedua-dua atribut tanpa kunci didapati bergantung secara transitif pada kunci primer.

II.8.4. *Entity Relationship Diagram*

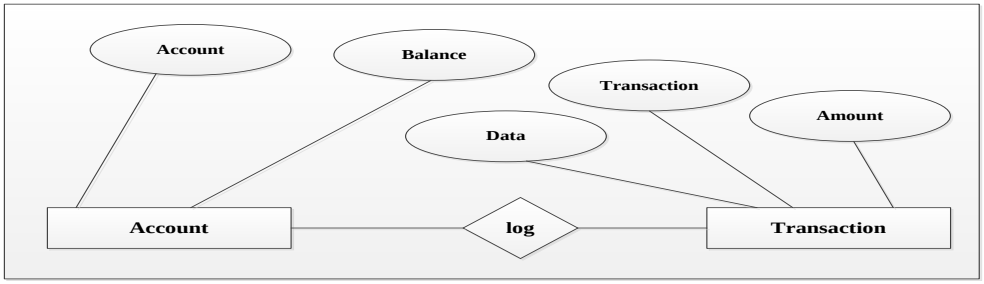
Entity relationship diagram merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data. Menurut Marlinda (2004:17), *modelentityrelationship* atau *entityrelationship diagram* adalah suatu penyajian data dengan menggunakan *entity* dan *relationship*. Diperkenalkan pada tahun 1976 oleh P.P. Chen. Ada beberapa alasan diperlukannya *entity relationship diagram*, yaitu :

1. Dapat menggambarkan hubungan antar entitas dengan jelas.
2. Dapat menggambarkan batasan jumlah entitas dan partisipasi antar entitas.
3. Mudah dimengerti oleh pemakai.
4. Mudah disajikan oleh perancang basis data.

Entity relationship diagram terbentuk karena didukung oleh beberapa komponen yang saling berhubungan satu sama lain. Komponen-komponen yang terdapat pada *entity relationship diagram* adalah sebagai berikut :

1. *Entity*, suatu yang dapat dibedakan dalam dunia nyata dimana informasi yang berkaitan dengannya dikumpulkan. Simbol *entity* adalah persegi panjang.
2. *Relationship*, merupakan hubungan yang terjadi antara satu atau lebih *entity*.
3. *Attribute*,merupakan karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang hal tersebut. Nilai *attribute* adalah suatu data yang aktual.
4. Indikator Tipe, ada dua, yakni : indikator tipe *asscociative object* dan indikator tipe supertipe.

5. *Cardinality Rasio,* menjelaskan hubungan batasan jumlah keterhubungan satu *entity* dengan *entity* lainnya atau banyaknya *entity* yang bersesuaian dengan *entity* yang lain melalui *relationship.*
6. Derajat *Relationship,* menyatakan jumlah entity yang berpatisipasi di dalam suatu *relationship*.
7. *Participantion Contraint,* menjelaskan apakah keberadaan suatu *entity* tergantung pada hubungannya dengan *entity* yang lain.
8. Representasi dari *Entity Set,* dipresentasikan dalam bentuk tabel dan nama yang *unique*. Setiap tabel terdiri dari sejumlah kolom dan diberi nama yang *unique*.



Gambar II.7. Contoh *Entity Relationship Diagram*

(Sumber : Marlinda, 2004:23)

II.8.5. Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem (Widyana, et al., 2013:40).Dengan kamus data sistem analis dapat mendefinisikan data yang mengalir pada sistem yang lengkap.Kamus data dibuat dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis kamus data digunakan sebagai alat

komunikasi antara sistem analis dengan user tentang data yang mengalir pada sistem tersebut serta informasi yang dibutuhkan oleh pemakai sistem.

Tabel II.3. Contoh Kamus Data

No.	Atribut	Type	Size	Keterangan
1.	no_periksa_umum	Integer	-	Nomor pemeriksaan umum
2.	Keluhan	Varchar	25	Keluhan

(Sumber : Widyana, et al., 2013:40)