# BAB II

# TINJAUAN PUSTAKA

## II.1. Sistem Informasi

### II.1.1. Sistem

Sistem adalah sekumpulan komponen yang saling berhubungan, bekerja sama untuk mencapai suatu tujuan dengan menerima input serta menghasilkan ouput dalam proses trantformasi yang teratur.

Sistem adalah sebuah tatanan yang terdiri atas sejumlah komponen fungsional (dengan tugas/fungsi khusus) yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses/pekerjaan tertentu. Sebagai contoh, sistem kendaraan terdiri dari : komponen starter, komponen pengapian, komponen penggerak, komponen pengerem, komponen kelistrikan-spedometer, lampu dan lain-lain. Komponen-komponen tersebut diatas memiliki tujuanyang sama yaitu untuk membuat kendaraan tersebut bisa dikendarai dengan nyaman dan aman. Contoh lain yaitu pergutuan tinggi, yang terdiri dari dosen, mahasiswa, kurikulum, dan lain-lain. Sistem ini bertujuan untuk menghasilkan mahasiswa-mahasiswa yang memiliki kemampuan dibidang ilmunya. (Jogiyanto : 2005 : 2)

### II.1.2. Konsep Dasar Sistem

Terdapat dua kelompok pendekatan didalam mendefenisikan sistem, yaitu menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan sistem yang lebih menekankan pada prosedur mendefenisikan sistem sebagai berikut.

Suatu sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatana atau menyelesaikan suatu sasaran tertentu.

Pendekatan sistem yang merupakan jaringan kerja dari prosedur lebih menekankan urut-urutan operasi didalam sistem, prosedur didefenisikan oleh Richard F. Neuschel sebagai berikut.

Suatu prosedur adalah suatu urut-urutan operasi klerikal (tulis menulis), biasanya melibatkan beberapa orang didalam satu atau lebih department, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi.

Pendekatan sistem yang menekankan pada komponen akan lebih mudah didalam mempelajari suatu sistem untuk tujuan analisis dan perancangan suatu sistem. Untuk menganalisis dan merencanakan suatu sistem, analis dan perancangan sistem haus dimengrti terlebih dahulu mengenai komponen-komponen atau elemen-elemen atau subsistem-subsistem dari sistem tersebut.

(Jogiyanto ; 2005 : 1)

### II.1.3. Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti yang menerimanya. Sumber dari informasi adalah data. Data merupakan bentuk jamak dari bentuk tunggal datung atau data-item. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Kejadian-kejadian (*event)* adalah suatu yang terjadi pada saat yang tertentu. Didalam dunia bisnis, kejadian-kejadian nyata yang sering terjadi adalah perubahan dari suatu nilai yang disebut dengan transaksi. Misalnya penjualan adalah transaksi perubahan nilai barang menjadi nilai uang atau nilai piutang dagang. Kesatuan nyata (*fact* dan *entity*) adalah berupa suatu objek nyata seperti tempat, benda dan orang yang betul-betul adan dan terjadi.

Berkaitannya dengan penyedia informasi bagi manajemen dalam mengambil suatu keputusan, yang diperoleh harus berkulalitas, maka kualitas dari informasi tergantung :

1. Akurat

   Akurat berarti bahwa informasi harus bebas dari kesalahan-kesalahan dan tidak biasa (menyesatkan) dan jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasikemungkinan banyak terjadi gangguan yang dapat merubah atau merusak informasi tersebut.

2. Relevansi

   Berarti informasi tersebut mempunyai manfaat untuk pemakaiannya. Relvansi informasi untuk tiap-tiap orang yang satu dengan yang lainnya berbeda.

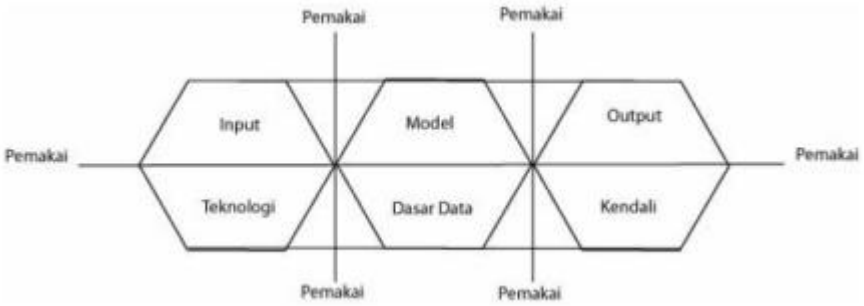
3. Tepat Waktu

   Tapat waktu berarti bahwa informasi yang datang pada penerimaan tidak boleh terlambat, informasi yang sudah usang tidak akan mempunyai nilai lagi. Karena informasi merupakan landasan dalam pengambilan keputusan. Dewasa ini mahalnya nilai informasi disebabkan harus ceptanya informasi itu didapat, sehingga diperlakukan teknologi-teknologi yang muktahir untuk mendapatkan, mengelolah dan mengirimkannya.

   Sedangkan sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manejerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak dari luar tertentu dangan laporan-laporan yang diperlukan (Robert A. Leitch dan K. Roscoe Davis)

   Dan menurut Jhon Burch dan Gary Grudnitski mengemukakan bahwa sistem informasi terdiri dari komponen-komponen yang disebutnya dengan blok bangunan (*building blocky*), yaitu blok masukan (*input block*), blok model (*model block*), blok keluaran (*output block*), blok

teknologi (*technology block*), blok basis data (*database block*) dan blok kendali (*control block*). Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lainnya membentuk satu kesatuan untuk mencapai sasarannya.




**Gambar II.1. Blok Sistem Informasi**

*(Sumber : Jogiyanto : 2005 : 12)*

Berdasarkan defenisi diatas dapat disimpulkan bahwa sistem informasi adalah sekumpulan prosedur organisasi yang pada saat dilaksanakan akan memberika informasi bagi pengambilan keputusan atau untuk pengendalian berinteraksi untuk menghasilkan informasi yang menunjang pengambilan keputusan operasi, manajerial, strategik.

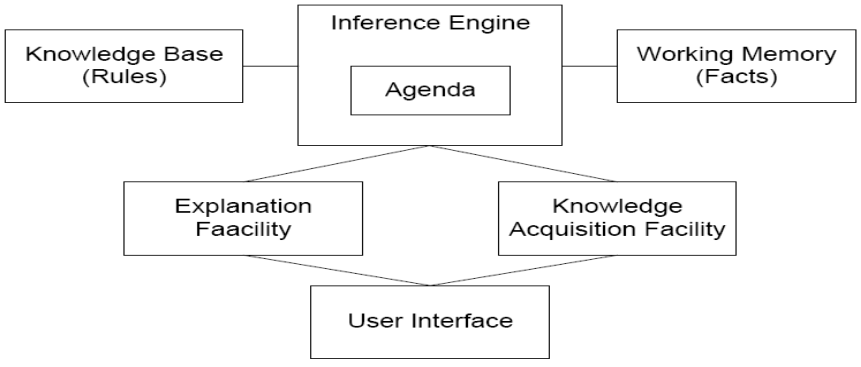
## II.2. Sistem Bisnis Cerdas

### II.2.1. Sistem Pakar

Menurut Rosnelly (2011:3), pakar atau ahli(*expert)* didefenisikan sebafai seseorang yang memiliki pengetahuan atau keahlian khusus yang tidak dimiliki oleh kebanyakan orang. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan kebanyakan orang. Dengan kata lain, dapat memecahkan suatu masalah dengan lebih efisien namun bukan berarti lebih murah. Pengetahuan yang dimuat kedalam sistem pakar dapat berasal dari seorang pakar atau pun pengetahuan yang berasal dari buku, jurnal, majalah, dan dokumentasi yang dipublikasikan lainnya, serta orang yang memilikipengetahuan meskipun bukan ahli. Istilah siste pakar(*expert system)*, sering disinonimkan dengan sistem berbasis pengetahuan(*knowledge-based-system)* atau sistem pakar berbasis pengetahuan(*knowledge based expert system)*.

### II.2.2. Karakteristik Sistem Paka

Adapun struktur sistem pakar dapat dilihat pada gambar II.




**Gambar II.2. Struktur Sistem Pakar**

*(Sumber : Rosnelly : 2011 : 12)*

Komponen yang terdapat dalam struktur sistem pakar ini adalah *knowladge base (rules), inference engine, working memory, explanation fasility, knowladge acquisition, dan user interface.*

1. *Knowladge base* (Basis pengetahuan)

   Basis pengetahuan mengandung pengetahuan untuk pemahaman formulasi, dan menyelesaikan masalah. Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupak informasi tentang objek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui. Pada struktur sistem pakar diatas, knowladge base disini untuk menyimpan pengetahuan dari pakar berupa rule / aturan (if <kondisi> then <aksi> atau dapat juga disebut condition-action rules).

2. *Inference engine* (Mesin inferensi)

   Mesin inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan *control struktur* (struktur kontrol) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi disini adalah *processor* pada sistem pakar yang mencocokkan bagian kondisi dari *rule* yang tersimpan didalam *knowladge base* dengan fakta tersimpan di *working memory.*

3. *Working memory*

   Berguna untuk menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai global database dari fakta yang digunakan oleh rule-rule yang ada.

4. *Expalanation facility*

   Menyediakan kebenaran dari solusi yang dihasilkan kepada user (*reasoning chain).*

5. *Knowladge acquisition facility*

   Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi keprogram komputer, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User interface*

   Mekanisme i=untuk memberi kesempatan kepada user dan sistem pakar untuk berkomunikasi. Antar muka menerima informasi dari pemakai dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikan kedalam bentuk yang dapat dimengerti oleh pemakai.

### II.1.3. Pakar

Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu.

Seorang pakar memiliki kemampuan kepakaran, yaitu :

1. Dapat mengenali dan merumuskan suatu masalah.
2. Menyelesaikan masalah dengan cepat dan tepat.
3. Menjelaskan solusi dari suatu masalah.
4. Restrukturisasi pengetahuan.
5. Belajar dari pengelaman.
6. Memehami batas kemampuan (Rosnelly : 2011 : 10).

### II.1.4. Kelebihan Sistem pakar

Sistem pakar memiliki bebrapa fitur menarik yang merupakan kelebihannya, seperti :

1. Meningkatkan ketersediaan (*increased availability).* Kepakaran atau kehalian menjadi tersedia dalam siste komputer. Dapat diakatakan bahwa sistem pakar merupakan produksi kepakaran secara massal.
2. Mengurangi biaya (*reduced cost).* Biaya yang diperlukan untuk menyediakan keahlian per satu orang *user* menjadi berkurang.
3. Mengurangi bahaya (*reduced danger).* Sistem pakar dapat digunakan dilingkungan yang mugkin berbahaya bagi manusia.

4. Permanen (*permanence)*. Sistem pakar dan pengetahuan yang terdapat didalamnya bersifat lebih permanen dari pada manusia yang dapat merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.
5. Keahlian multiple (*multiple expertise)*. Pengetahuan dari beberapa pakar dapat dimuat kedalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian atau pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar.
6. Meningkatkan kehandalan ( *increased reliability)*. Sistem pakar meningkatkan kepercayaan dengan memberikan hasil yang benar sebagai alternatif pendapat dari seorang pakar atau sebagai penengah jika terjadi konflik antara beberapa pakar.
7. Penjelasan (*explanation)*. Sistem pakar dapat menjelaskan detail proses penalaran (*reasoning)* yang dilakukan hingga mecapai suatu kesimpulan.
8. Respon yang cepat (*fast response)*. Repson yang cepat atau *real time* diperlukan pada beberapa aplkasi. Meskipun bergantung pada *hardware* dan *software* yang digunakan, namun sistem pakar relatif memberikan respon yang lebih cepat dibandingkan seorang pakar.
9. Stabil, tidak emosional, dan memberikan respn yang lengkap setiap saat. Karakteristik ini diperlukan pada situasi *real time* dan

keadaan darurat ketika seorang pakar mungkn tidak berada pada kondisi puncak disebabkan oleh stres atau kelelahan.

10. Pembimbing pintar (*intelligent tutor).* Sistem pakar dapat berperan sebagai *intelligent tutor* dengan memberikan kesempatan pada user untuk menjalankan contoh program dan menjelaskan proses *reasoning* yang dilakukan.
11. Basis data cerdas. Sistem pakar dapat digunakan untuk mengakses basis data secara cerdas (Rosnelly : 2011 : 5).

### II.2.5. Kekurangan Sistem Pakar

Selain manfaat, ada juga beberapa kekurangan yang ada pada sistem pakar diantaranya :

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan memeliharanya.
3. Sistem pakar tidak 100 % bernilai benar.

## II.3. *Certanty Factor*

Faktor ketidak pastian merupakan cara dari penggabungan kepercayaan dan ketidak percayaan dalam bilangan tunggal. Dalam teori kepastian, data-data kualitatiif direpresentasikan sebagai derajat kepastian, (*degree of belief* ). Dalam menggambarkan derajat keyakinan, teori kepastian menggunakan nilai yang disebut certainty factor (CF) untuk mengasumsikan derajat keyakinan seorang

pakar terhadap suatu data. *Certainty Factor* menerapkan konsep keyakinan (*belief*) dan ketidak pastian (*disbelief*).

CF(H,E)=MB(H,E)-MD(H,E)

$$MB(H,E) = \frac{Max[P(H \square E),P(H)] - P(H)}{Max[1,0] - P(H)} \quad (1)$$

$$MD(H,E) = \frac{Min[P(H \square E),P(H)] - P(H)}{Min[1,0] - P(H)} \quad (2)$$

Keterangan:

CF = *Certainty Factor* dalam hipotesis (H) dipengaruhi oleh fakta E

MB = *Measure of belief* merupakan ukuran kenaikan dari kepercayaan hipotesis (H) yang dipengaruhi oleh fakta B.

MD = *Measure of Increased Disbelief* merupakan ukuran kenaikan dari ketidak percayaan hipotesis (H) yang dipengaruhi oleh fakta E.

E = Evidence (peristiwa/fakta)

H = Hipotesa (dugaan)

P(H □ E)= probabilitas (H) benar karena fakta E

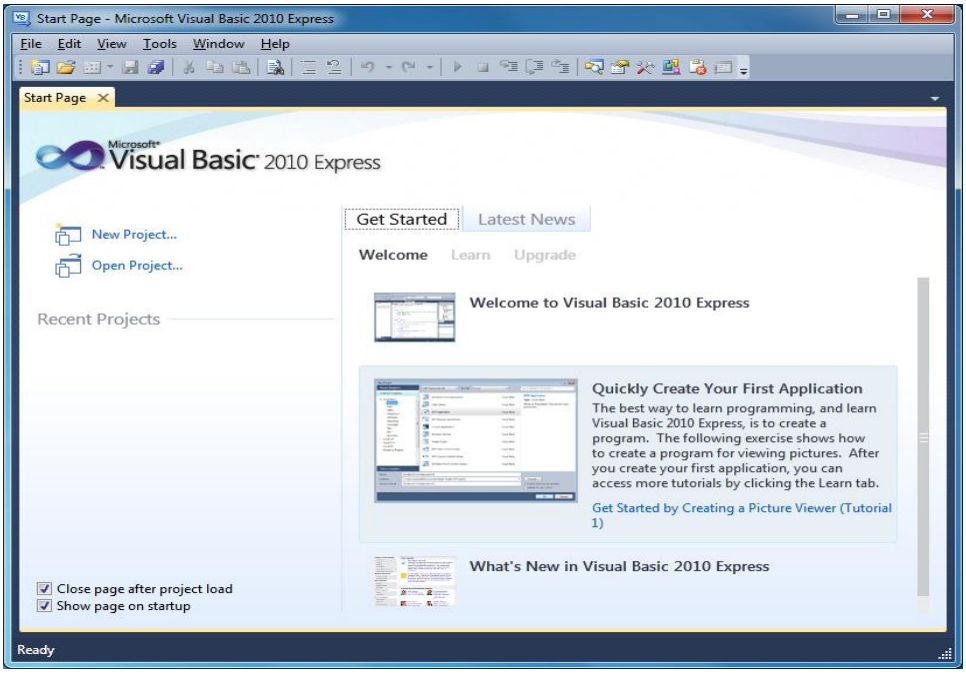
(Reski Mai Candra dan Weni Rahim : 2014 : 19).

**II.4. Pemograman Visual Basic 2010**

Visual basic 2010 merupakan salah satu bagian dari produk pemograman terbaru yang dikeluarkan oleh *microsoft*, yaitu *microsoft* visual studio 2010. Visual studio merupakan prosuk pemograman andalan dari *microsoft corporation*, dimana didalamnya berisi beberapa jenis IDE pemograman seperti visual basic, visual C++, visual web depelover, 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 standart yang disertakan adalah Microsoft SQL Server 2008 express.

Visual basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu visual basic 2008. Beberapa pengembangannya yang terdapat didalamnya antara lain dukungan terhadap library terbaru dari *microsoft*.




**Gambar II.2. Tampilan Microsoft Visual Basic 2010**
*(Sumber : Wahana komputer : 2011)*

## II.5. SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang *database*. SQL Server adalah DBMS (Database Management *System*) yang dibuatoleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM danOracle. SQl Server 2008 dibuat pada saat kemajuan dalam bidang *hardware* sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 membawa beberapa terobosan dalam bidang pengolahan danpenyimpanan data.

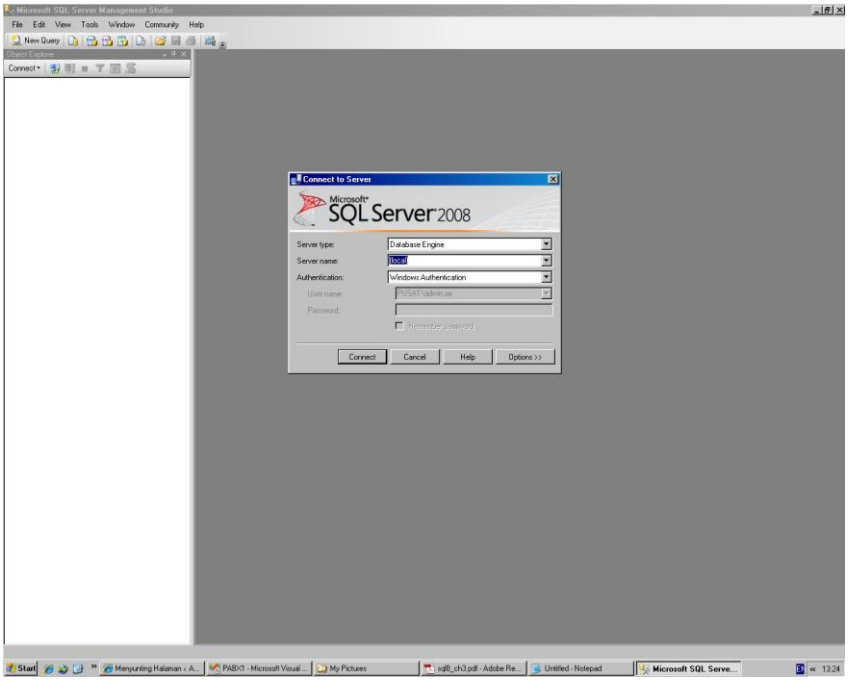
Microsoft merilis SQL Server 2008 dalam beberapa versi yang disesuaikan dengan segment-segment pasar yang dituju.Versi-versi tersebut adalah sebagai berikut. Menurut cara pemrosesan data pada prosesor maka Microsoft mengelompokkan produk ini berdasarkan 2 jenis yaitu :

- Versi 32-bit(x86), yang biasanya digunakan untuk komputer dengan single prosesor (Pentium 4) atau lebih tepatnya prosesor 32 bit dan sistem operasi Windows XP.
- Versi 64-bit(x64), yang biasanya digunakan untuk komputer dengan lebih dari satu prosesor (Misalnya Core 2 Duo) dan system operasi 64 bit seperti Windows XP 64, Vista, dan Windows 7.

Sedangkan secara keseluruhan terdapat versi versiseperti berikut ini:

- Versi Compact, ini adalah versi “Tipis” dari semua versi yang ada. Versi ini seperti versi desktop pada SQL Server 2000. Versi ini juga digunakan pada handled drvice seperti Pocket PC, PDA, SmartPhone, Tablet PC.

- Versi Express, ini adalah versi “Ringan” dari semua versi yang ada (tetapi versi ini berbeda dengan versi compact) dan paling cocok untuk latihan para pengembang aplikasi. Versi ini memuat Express Manager standar, integrasi dengan CLR dan XML (Wenny Widya dan Iskandar Zulkarnaen).




## II.6. Kamus Data

Kamus data (KD) atau *data dictionary* (DD) atau disebut juga dengan istilah *sistem data dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan KD,

analisis sistem dapat mendefenisikan data yang mengalir disistem dengan lengkap. KD dibuat pada tahap perancangan sistem. Pada tahap analisis, KD dapat digunakan sebagai alat komunikasi antara analisis sistem dengan pemakai sistem tentang data yang mengalir disistem, yaitu tentang data yang masuk kesistem dan tentang informasi yang dibutuhkan oleh pemakai sistem. Pada tahap perancangan sistem, KD digunakan untuk merancang *input*, merancang laporan-laporan dan database. KD dibuat berdasarkan arus data yang ada di DAD, arus data di DAD sifatnya adalah global, hanya ditunjukkan nama arus datanya saja (Jogiyanto : 2005 :725)

## II.7. *Unified Modeling Language* (UML)

*Unified modeling language (UML)* Adalah sebuah bahasa yang diterima dan digunakan oleh *software* developer dan *software analist analyst* sebagai suatu bahasa yang cocok untuk mempresentasikan grafik dari suatu relasi antar entitas-entitas *software* (Gornik 2003). Dengan menggunakan UML, tim pengembang *software* akan memiliki banyak keuntungan, seperti memudahkan komunikasi dengan sesama anggota tim tentang *software* apa yang akan dibuat, memudahkan integrasi kedalam area pengerjaan *software* karena bahasa ini berbasiskan meta-models bisa mendefenisikan proses-proses untuk mengkontruksikan konsep-konsep yang ada. UML juga menggunakan formatur input dan output yang sudah mempunyai bentuk standar yaitu XML metadata interchange (XMI), menggunakan aplikasi dan pemodelan data yang universal, mempresentaskan dari

tahap analisis keimplementasi lalu ke *deyploment* yang terpadu, dan mendeskripsikan keutuhan tentang spesifikasi *software* (Edgar winata dan Johan setiawan : 2013).

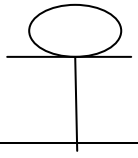
Peralatan utama UML adalah diagram-diagram yang digunakan untuk membantu manusia dalam memvisualisasikan proses pengembangan sebuah sistem perangkat lunak, sama seperti penggunaan denah dalam pembuatan bangunan.

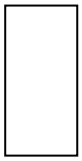
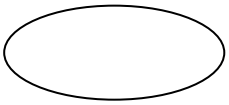
**1. *Use Case Diagram***

*Use case* diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi – fungsi tersebut.

Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

**Tabel II.1 Simbol-simbol *use case***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *Actor* | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi denga *use case.* |
| - - - - - - > | *Include* | Menspesifikasikan bahwa *use case* sumber secara explisit. |

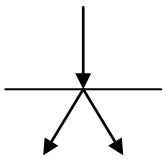
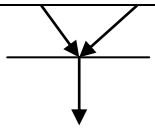
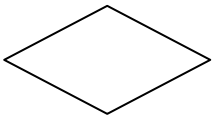
| | | |
|---|---|---|
| <- - - - - - - - | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan |
| ——————— | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya |
|  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas |
|  | *Use case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |

### *2. Activity Diagram (*Diagram aktivitas)

Teknik untuk menjelaskan busines *process, procedural logic*, dan *work flow.* Bisa dipakai untuk menjelaskan teks *use case* dalam notasi grafis dengan menggunakan notasi yang mirip *flow chart*, meskipun terdapat sedikit perbedaan notasi (Edgar winata dan Johan setiawan : 2013).

**Tabel II.2. Simbol-simbol *Activity Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | Start Point | Diletakkan pada pojok kiri dan merupakan awal aktivitas |
|  | End point | Akhir aktivitas |

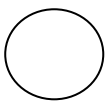
| | | |
|---|---|---|
|  | Activities | Menggambarkan suatu proses kegiatan |
|  | Fork | Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan parallel jadi satu |
|  | Join | Digunakan untuk menunjukkan adanya dekomposisi |
|  | Decision point | Menggambarkan pilihan untuk pengambilan keputusan, true, false |
| | Swimlane | Pembagian activity diagram untuk menunjukkan siapa melakukan apa |

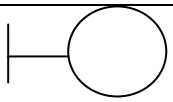
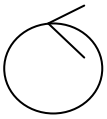
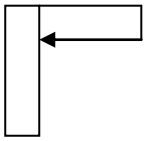
(*Sumber : Edgar winata dan Johan setiawan : 2013*)

**3. *Sequence Diagram* (Diagram Urutan)**

Menjelaskan interaksi objek-objek yang saling berkolaborasi, mirip dengan activity diagram yaitu meggambarkan alur kejadian sebuah aktivitas tetapi lebih detail dalam menggambarkan aliran data termasuk data atau *behaviour* yang dikirimkan atau diterima namun kurang mampu menjelaskan detail diri dari sebuah algoritma (Edgar winata dan Johan setiawan : 2013).

**Gambar II.3. Simbol-Simbol *Squence Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *Entity clas* | Merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal |

| | | |
|---|---|---|
| | | sistem dan menjadilandasan untuk menyususun basis data |
|  | *Boundary Class* | Berisi kumpulan kelasyang menjadi interface atau interaksi antara satu atau lebih aktor dengan sistem |
|  | *Control Class* | Suatu objek yang berisi logoka aplikasi yang tidak memiliki tanggung jawab kepada entitas. |
|  | *Message* | Simbol mengirim pesan antar class |
|  | *Recursive* | Menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri |
|  | *Activation* | Meawakili sebuah eksekusi operasi dari objek, penjang kotak ini berbanding lurus dengan durasi aktivitas sebuah opersi |
|  | *Lifeline* | Garis titik-titik yang terhubung objek, sepanjang lifeline terdapat activation |

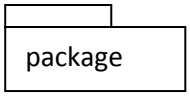
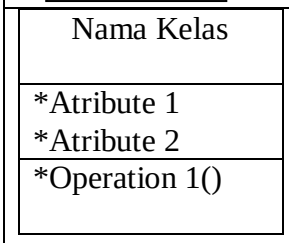
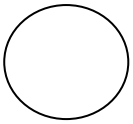
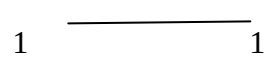
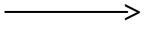
***(****Sumber : Edgar winata dan Johan setiawan : 2013)*

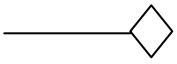
**4. *Class Diagram* (Diagram kelas)**

*Class diagram* merupakan diagram paling umum yang dijumpai dalam pemodelan bebasis UML. Didalam *class diagram* terdapat *class* dan

*interface* besarta atribut-atribut dan operasinya, realasi yang terjadi antar objek, *constraint* terhadap objek-objek yang saling berhubungan dan *inheritance* untuk organisasi class yang lebih baik. *Class diagram* juga terdapat *static view* dari elemen pembangun sistem. Pada intinya class diagram mampu membantu proses pembutan sistem dengan memanfaatkan konsep *forward* ataupun *reverse engineering* (Edgar winata dan Johan setiawan : 2013).

**Tabel II.4. Simbol-Simbol *Class Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *Package* | Merupakan bungkusan dari satu atau lebih kelas |
|  | *Operasi* | Kelas pada struktur sistem |
|  | *Interface* | Sama dengan konsep *interface* dalam pemograman berorientasi objek |
|  | *Asosiasi* | Relasi antar kelas dengan makna umum |
|  | *Asosiasi* berarah | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain |
| | | |

| | | |
|---|---|---|
|  | *Generalisasi* | Relasi antar kelas dengan makna *generalisasi-spesialisasi* |
| - - - - - - - - - -> | *Kebergantungan* | Relasi antar kelas dengan makna kebergantungan antar kelas |
|  | *Agregasi* | Relasi antar kelas dengan makna semua bagian |

***(****Sumber : Edgar winata dan Johan setiawan : 2013*)

## II.8. Normalisasi

Menurut Martin (1975), Normalisasi diartikan sebagai suatu teknik yang menstrukurkan/ mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan (Edy Sutanta ; 2011 : 174).

Proses normalisasi menghasilkan relasi yang optimal, yaitu (Martin, 1975) : (Sutanta ; 2011 : 175).

1. Memiliki struktur *record* yang konsisten secara logik;
2. Memiliki struktur *record* yang mudah untuk dimengerti;

3. Memiliki struktur *record* yang sederhana dalam pemeliharaan;
4. Memiliki struktur *record* yang mudah ditampilkan kembali untuk memenuhi kebutuhan pengguna;
5. Minimalisasi kerangkapan data guna meningkatkan kinerja sistem.

Secara berturut-turut masing-masing level normal tersebut dibahas berikut ini, dimulai dari bentuk tidak normal (Sutanta : 2011 : 176-179).

1. Relasi bentuk tidak normal (*Un Normalized Form* / UNF)

   Relasi-relasi yang dirancang tanpa mengindahkan batasan dalam defisi basis data dan karakteristik *Relational Database Management System* (RDBM) menghasilkan relasi *Un Normalized Form* (UNF). Bentuk ini harus di hindari dalam perancangan relasi dalam basis data. Relasi *Un Normalized Form* (UNF) mempunyai kriteria sebagai berikut.

   a. Jika relasi mempunyai bentuk *non flat file* (dapat terjadi akibat data disimpan sesuai dengan kedatangannya, tidak memiliki struktur tertentu, terjadi duplikasi atau tidak lengkap)
   b. Jika relasi membuat *set atribut* berulang *(non single values)*
   c. Jika relasi membuat *atribut non atomic value*

2. Relasi bentuk normal pertama (*First Norm Form* / 1NF)

   Relasi disebut juga *First Norm Form* (1NF) jika memenuhi kriteria sebagai berikut.

   a. Jika seluruh atribut dalam relasi bernilai *atomic ( atomic value)*
   b. Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*

c. Jika relasi tidak memuat set atribut berulang

d. Jika semua record mempunyai sejumlah atribut yang sama.

Permasalahan dalam *First Norm Form* (1NF) adalah sebagai berikut.

a. Tidak dapat menyisipkan informasi parsial

b. Terhapusnya informasi ketika menghapus sebuah *record*

3. Bentuk normal kedua *(Second Normal Form /* 2NF)

Relasi disebut sebagai *Second Normal Form (*2NF) jika memenuhi kriteria sebagai berikut

a. Jika memenuhi kriteria *First Norm Form (*1NF)

b. Jika semua atribut nonkunci *Functional Dependence* (FD) pada *Primary Key* (PK)

Permasalahan dalam *Second Normal Form* / 2NF adalah sebagai berikut:

a. Kerangkapan data *(data redundancy)*

b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data *(data inconsistency)*

c. Proses pembaharuan data tidak efisien

Kriteria tersebut mengidentifikasikan bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Norm Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Norm Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

a. Identifikasikan *Functional Dependence* (FD) relasi *First Norm Form* (1NF)
b. Berdasarkan informasi tersebut, dekomposisi relasi *First Norm Form* (1NF) menjadi relasi-relasi baru sesuai *Functional Dependence* nya. Jika menggunakan diagram maka simpul-simpul yang berada pada puncak diagram ketergantungan data bertindak *Primary Key* (PK) pada relasi baru

4. Bentuk normal ketiga (*Third Norm Form* / 3NF)

   Suatu relasi disebut sebagai *Third Norm Form* jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Second Normal Form* (2NF)
   b. Jika setiap atribut nonkunci tidak (TDF) *(Non Transitive Dependeny)* terhadap *Primary Key* (PK)

   Permasalahan dalam *Third Norm Form* (3NF) adalah keberadaan penentu yang tidak merupakan bagian dari *Primary Key* (PK) menghasilkan duplikasi rinci data pada atribut yang berfungi sebagai *Foreign Key* (FK) (duplikasi berbeda dengan keterangan data).

   Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Norm Form* (3NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

   a. Identifikasi TDF relasi *Second Normal Form* (2NF)

b. Berdasarkan informasi tersebut, dekomposisi relasi *Second Normal Form* (2NF) menjadi relasi-relasi baru sesuai TDF-nya.

5. Bentuk normal *Boyce-Cood (Boyce-Codd Norm Form* / BCNF)

   Bentuk normal *Boyce-Codd Norm Form* (BCNF) dikemukakan oleh R.F. Boyce dan E.F. Codd. Suatu relasi disebut sebagai *Boyce-Codd Norm Form* (BCNF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Third Norm Form* (3NF)
   b. Jika semua atribut penentu (determinan) merupakan CK

6. Bentuk normal keempat (*Forth Norm Form* / 4NF)

   Relasi disebut sebagi *Forth Norm Form* (4NF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Boyce-Codd Norm Form*.
   b. Jika setiap atribut didalamnya tidak mengalami ketergantungan pada banyak nilai.

7. Bentuk normal kelima (*Fifth Norm Form* / 5NF)

   Suatu relasi memenuhi kriteria *Fifth Norm Form* (5NF) jika kerelasian antar data dalam relasi tersebut tidak dapat direkonstruksi dari struktur relasi yang sederhana.

8. Bentuk normal kunci domain (*Domain Key Norm Form* / DKNF)

Relasi disebut sebagai *Domain Key Norm Form* (DKNF) jika setiap batasan dapat disimpulkan secara sederhana dengan mengetahui sekumpulan nama atribut dan domainnya selama menggunkan sekumpulan atribut pada kuncinya.