BAB II

TINJAUAN PUSTAKA

II.1. Teori Sistem

Sistem mempunyai beberapa pengertian, tergantung dari sudut mana kata tersebut didefenisikan. Menurut Kusrini (2010:5) Secara garis besar ada dua pendekatan yang dilakukan yaitu :

1. Pendekatan sistem yang lebih menekankan pada elemen-elemen atau kelompoknya, yang didalam hal ini sistem ini didefenisikan sebagai suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu aturan tertentu.
2. Pendekatan sistem sebagai jaringan kerja dari prosedur, yang lebih menekankan urutan operasi didalam sistem. Prosedur didefenisikan sebagai urutan operasi kerja (tulis-menulis), yang biasanya melibatkan beberapa orang didalam satu atau lebih departemen yang diterapkan untuk menjamin penanganan yang seragam dari transaksi bisnis yang terjadi.

Pendekatan sistem yang lebih menekankan pada elemen-elemen atau komponennya mendefisikan sistem sebagai sekumpulan elemen-elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Dengan demikian didalam suatu sistem, komponen-komponen ini tidak dapat berdiri sendiri, tetapi sebaliknya, saling berhubungan hingga berbentuk suatu kesatuan hingga tujuan sistem dapat tercapai.

II.1.1. Karakteristik sistem

Sistem mempunyai beberapa karakteristik atau sifat-sifat tertentu antara lain, dalam Kusrini (2010:6):

1. Komponen sistem (*Components*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling bekerja sama untuk membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupaya suatu subsistem atau bagian dari sistem. Setiap sistem tidak peduli berapapun kecilnya, selalu mengandung komponen-komponen atau subsistem.

2. Batasan sistem (*Boundary*)

   Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya.

3. Lingkungan luar sistem (*Environtment*)

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut

4. Penghubung sistem (*Interface*)

   Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Melalui perhubungan ini memungkinkan sumber-sumber daya mengalir dari subsistem yang lainnya.

5. Masukan sistem (*Input*)

   Merupakan energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*).

6. Keluaran sistem (*Output*)

Keluaran adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolah sistem (*Process*)

   Suatu sistem yang dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran.

8. Sasaran sistem (*Objective*)

   Suatu sistem pasti mempunyai tujuan atau sasaran. Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

II.1.2. Klasifikasi sistem

Sistem dapat diklasifikasikan dari beberapa sudut pandangan, diantaranya sebagai berikut, dalam Kusrini (2010:7) :

1. Sistem abstrak dan sistem fisik.

   Sistem abstrak adalah sistem yang berisi gagasan atau konsep. Misalnya, sistem teologi yang berisi gagasan tentang hubungan manusia dan Tuhan. Sistem fisik merupakan sistem yang secara fisik dapat dilihat. Misalnya sistem komputer, sistem sekolah, sistem akuntansi, dan sistem transportasi.

2. Sistem alamiah dan sistem buatan manusia.

   Sistem alamiah adalah sistem yang terjadi karena alam (tidak dibuat manusia). Misalnya, sistem tata surya. Sistem buatan manusia adalah sistem yang dibuat oleh manusia. Misalnya, sistem komputer dan sistem mobil.

3. Sistem tertentu dan sistem tak tentu.

Sistem tertentu adalah sistem yang operasinya dapat diprediksi secara tepat. Misalnya, sistem komputer. Sistem tak tentu adalah sistem yang tak dapat diramal dengan pasti karena mengandung unsur probabilitas. Misalnya, sistem arisan dan sistem sediaan.

4. Sistem tertutup dan sistem terbuka

   Sistem tertutup adalah sistem yang tidak bertukar materi, informasi, atau energi dengan lingkungan. Misalnya, reaksi kimia dalam tabung terisolasi. Sistem terbuka adalah sistem yang berhubungan dengan lingkungan dan dipengaruhi oleh lingkungan.

II.2 Sistem Pendukung Keputusan

Sistem pendukung keputusan (Inggris: *decision support systems* disingkat DSS) adalah bagian dari sistem informasi berbasis komputer (termasuk sistem berbasis pengetahuan (manajemen pengetahuan)) yang dipakai untuk mendukung pengambilan keputusan dalam suatu organisasi atau perusahaan.

II.2.1 Pengertian Sistem Pendukung Keputusan

Konsep awal sistem pendukung keputusan dikenalkan pertama kali oleh Scott Morton pada awal tahun 1970-an. Ia mendefinisikan DSS sebagai sistem berbasis komputer interaktif, yang membantu para pengambil keputusan untuk menggunakan data dan berbagai model untuk memecahkan masalah-masalah tidak terstruktur, dalam Turban, dkk (2005).

Sistem pendukung keputusan atau *Decision Support System* menunjukkan sebuah sistem yang dimaksudkan untuk mendukung para pengambil keputusan manajerial dalam situasi keputusan semiterstruktur, dalam Turban, dkk (2005).

Alter (dalam Kusrini, 2007) mendefinisikan sistem pendukung keputusan merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan dan manipulasi data. Sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi yang semiterstruktur dan

situasi tidak terstruktur, dimana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat.

Berdasarkan penjelasan diatas, dapat disimpulkan bahwa sistem pendukung keputusan merupakan sistem yang mampu memberikan penilaian terhadap alternatif guna untuk membantu para manajer dalam pengambilan keputusan.

II.2.2 Karakteristik Sistem Pendukung Keputusan

Menurut Turban, dkk (2005), karakteristrik dan kapabilitas Sistem Pendukung Keputusan adalah sebagai berikut :

1. Dukungan untuk pengambilan keputusan, terutama pada situasi semiterstruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi.
2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok. Masalah yang kurang terstruktur sering memerlukan keterlibatan individu dari departemen dan tingkat organisasional yang berbeda atau bahkan dari organisasi lain.
4. Dukungan untuk keputusan independen dan atau sekuensial. Keputusan dapat dibuat satu kali, beberapa kali atau berulang (dalam interval yang sama).
5. Dukungan di semua fase proses pengambilan keputusan: intelegensi, desain, pilihan, dan implementasi.
6. Dukungan diberbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu. Pengambilan keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara cepat dan dapat mengadaptasikan DSS untuk memenuhi perubahan tersebut.

II.2.3 Ciri-Ciri Sistem Pendukung Keputusan

Kriteria atau ciri-ciri sistem pendukung keputusan adalah sebagai berikut, dalam Dicky Nofriansyah, S.Kom, M.Kom (2014:2) :

1. Banyak pilihan/alternatif
2. Ada kendala atau surat
3. Mengikuti suatu pola atau model tingkah laku, baik yang terstruktur maupun tidak terstruktur.
4. Banyak input/Variabel
5. Ada faktor resiko, dibutuhkan kecepatan, ketepatan dan keakuratan .

II.2.4 Fase Dalam Sistem Pendukung Keputusan

Tiga fase dalam pengambilan keputusan yaitu, dalam Dicky Nofriansyah, S.Kom, M.Kom (2014:2) :

1. *Inteligence*

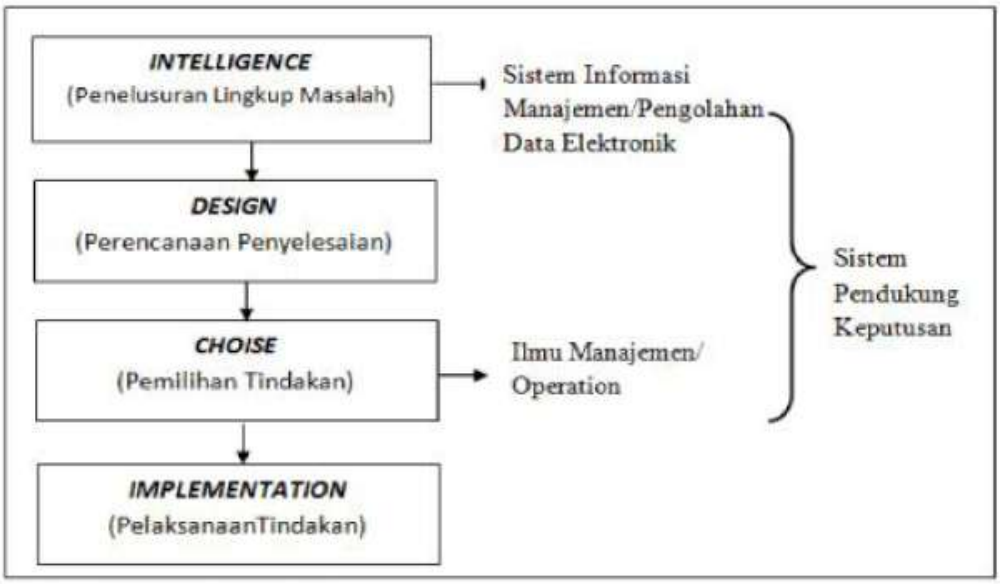
   Tahap ini merupakan proses penelusuran dan pendeteksian dari ruang lingkup problematika secara proses pengenalan masalah. Data masukan diperoleh dan diuji dalam rangka mengidentifikasi masalah.

2. *Design*

   Tahap ini merupakan proses menemukan, mengembangkan dan menganalisis alternatif tindakan yang bisa dilakukan. Tahap melakukan pengujian kelayakan solusi.

3. *Choice*

   Pada tahai ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan kedalam proses pengambilan keputusan.



Gambar II.1 Fase Proses Pengambilan Keputusan

Sumber (Dicky Nofriansyah, S.Kom, M.Kom ; 2014)

II.2.5 Komponen Sistem Pendukung Keputusan

Secara garis besar sistem pendukung keputusan dibangun oleh tiga komponen utama yaitu, dalam Dicky Nofriansyah, S.Kom, M.Kom (2014:3) :

1. Sub Sistem Data *(Database)*

   Sub sistem data merupakan komponen sistem pendukung keputusan yang berguna sebagai penyedia data bagi sistem. Data tersebut disimpan untuk diorganisasikan dalam sebuah basis data yang diorganisasikan oleh suatu sistem yang disebut dengan Sistem Manajemen Sistem Basis Data (*Database Management System)*

2. Sub sistem Model

   Model adalah suatu tiruan dari alam nyata. Kendala yang sering diahadapi dalam merancang model adalah bawah model yang dirancang tidak mampu mencerminkan seluruh variabel alam nyata, sehingga keputusan yang diambil tidak sesuai dengna kebutuhan. Oleh karena itu, dalam menyimpan berbagai model harus diperhatikan dan harus diajaga fleksibilitasnya.

Hal lain yang harus diperhatikan adalah pada setiap model yang disimpan hendaknya ditambahkan rincian, keterangan dan penjelasan yang komprehensif mengenai model yang dibuat.

3. Sub sistem dialog (*User System Interface)*

   Sub sistem dialog adalah fasilitas yang mampu mengintegrasikan sistem yang terpasang dengan pengguna secara interaktif, yang dikenal dengan sub sistem dialog. Melalui sub sistem dialog sistem diimplementasikan sehingga pengguna dapat berkomunikasi dengan sistem yang dibuat.

II.2.6 Tujuan Sistem Pendukung Keputusan

Tujuan dari sistem pendukung keputusan adalah sebagai berikut, dalam Dicky Nofriansyah, S.Kom, M.Kom (2014:4) :

1. Membantu dalam pengambilan keputusan atas masalah yang terstruktur
2. Memberikan dukungan atas pertimbangan manager dan bukannya dimaksudkan untuk menggantikan fungsi manager.
3. Meningkatkan efektifitas keputusan yang diambil lebih dari perbaikan efesiensinya.
4. Kecepatan komputasi komputer memungkinkan para pengambil keputusan untuk banyak melakukan komputasi secara cepat dengan biaya yang sangat rendah.
5. Peningkatan produktifitas membangun suatu kelompok pengambil keputusan, terutama para pakar bisa sangat mahal. Sistem pendukung keputusan komputerisasi mengurangi ukuran kelompok dan memungkinkan para anggotanya untuk berada diberbagai lokasi yang berbeda-beda (menghemat biaya perjalanan). Selain itu produktifitas staf pendukung (misalnya analisis keuangan dan hokum) bisa ditingkatkan. Produktivitas juga ditingkatkan menggunakan peralatan optimalisasi yang menjalankan sebuah bisnis.

II.3. Peramalan

Menurut Edi Windarto (2013:79), Peramalan adalah kegiatan memperkirakan atau memprediksi apa yang terjadi pada waktu yang akan datang, sedangkan rencana merupakan penentuan apa yang akan dilakukan pada waktu yang akan datang. Peramalan menjadi sangat penting karena penyusunan suatu rencana diantaranya didasarkan pada suatu proyeksi atau Peramalan.

Peramalan adalah suatu untuk memperkirakan keadaan dimasa yang akan datang melalui pengujian keadaan dimasa lalu. Dalam kehidupan sosial segala sesuatu itu serba tidak pasti, sukar diperkirakan secara tepat. Dalam hal ini perlu diadakan peramalan. Peramalan yang dibuat selalu diupayakan agar dapat meminimumkan pengaruh ketidak pastian ini terhadap sebuah permasalahan. Dengan kata lain peramalan bertujuan mendapatkan peramalan yang bisa meminimumkan kesalahan meramal (*forecat error*) yang biasanya diukur dengan *mean square error, mean absolute error*, dan sebagainya.

Kegunaan peramalan terlihat pada saat pengambilan keputusan. Keputusan yang baik adalah keputusan yang didasarkan atas pertimbangan – pertimbangan yang akan terjadi pada waktu keputusan itu dilaksanakan.

Keberhasilan dari suatu peramalan sangat ditentukan oleh:

1. Pengetahuan teknik tentang pengumpulan informasi (data) masa lalu, data ataupun informasi tersebut bersifat kuantitatif
2. Teknik dan metode yang tetap dan sesuai dengan pola data yang telah dikumpulkan.

Gambaran perkembangan pada masa lalu yang akan datang diperoleh dari hasil analisa data yang didapat dari penelitian yang telah dilakukan. Perkembangan pada masa depan merupakan perkiraan apa yang akan terjadi, sehingga dapat dikatakan bahwa peramalan selalu

diperlukan didalam penelitian. Ketepatan penelitian merupakan hal yang penting, walaupun demikian perlu diketahui bahwa sesuatu ramalan selalu ada unsur kesalahannya, sehingga yang perlu diperhatikan adalah usaha untuk memperkecil kesalahan dari ramalan tersebut.

II.4 Metode Rata – rata Bergerak Tunggal (*Single Moving Averages)*

Menurut Edi Windarto (2013:80), Menetukan ramalan dengan metode *single moving averages* cukup mudah dilakukan. Bila akan menerapkan 4 bulan rata-rata bergerak maka maka ramalan pada bulan Mei dihitung sebesar rata-rata dari 4 bulan sebelumnya, yaitu bulan Januari, Februari, Maret, April. Persamaan Matematis dari teknik ini adalah :

$$F_{t+1} = \frac{X_1 + X_2 + ...X_T}{T} \qquad (1)$$

Keterangan :

F_{t+1} : Ramalan untuk periode ke t + 1

X_T : Nilai riil periode ke t

T : jangka waktu rata-rata bergerak.

Metode single moving average memiliki karakteristik khusus, yaitu:

1. Untuk menentukan ramalan pada periode yang akan datang memerlukan data historis selam jangka waktu tertentu.
2. Semakin panjang jangka waktu *moving averages*, efek pelicinan semakin terlihat dalam ramalan atau menghasilkan *moving average* yang semakin halus. Artinya pada *moving averages* yang jangka waktunya lebih panjang, perbedaan ramalan terkecil dengan ramalan terbesar menjadi lebih kecil.

II.4.1. Menghitung Kesalahan Ramalan

Hasil proyeksi yang akurat adalah *forecast* yang bisa meminimalkan kesalahan meramal (*forecast error*). Besarnya *forecast error* dihitung dengan mengurangi data riil dengan besarnya ramalan.

Error (E) = X_t - F_t

Keterangan :

X_t = data riil periode ke-t

F_t = ramalan periode ke-t

Dalam menghitung *forecast error* digunakan :

1. *Mean Absolute Error*

   Mean Absolute Error adalah rata-rata absolute dari kesalahan meramal, tanpa menghiraukan tanda positif maupun negatif.

$$MAE = \frac{\sum \left| X_t - F_t \right|^2}{n}$$

2. *Mean Squared Error*

   Mean Squared Error adalah kuadrat rata-rata kesalahan meramal.

$$MSE = \frac{\sum \left| X_t - F_t \right|^2}{n}$$

   Metode ini mudah menghitungnya dan sederhana, tetapi mempunyai kelemahan-kelemahan antara lain :

   1. Perlu data histories yang cukup,
   2. Data tiap periode diberi *weight* (bobot) sama,
   3. Kalau fluktuasi data tidak random, tidak menghasilkan forecasting yang baik.

3. Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error merupakan nilai tengah kesalahan persentase absolute dari suatu peramalan.

$$MAPE = \frac{\sum |APE|}{n}$$

4. Percentage Error (PE)

Percentage Error merupakan Kesalahan persentase dari suatu peramalan,

$$PE = \left\{ \frac{X_t - F_t}{n} \right\} x100$$

dimana :

x_t =nilai data ke periode ke-t

f_t =nilai ramalan periode ke-t

n =banyaknya data

II.5. *Unified Modeling Language* (*UML)*

Unified Modeling Language (*UML)* adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma "berorientasi objek". Pemodelan *(Modeling)* sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajaran dan dipahami. Dalam hal ini sasaran model sesungguhnya adalah abstraksi segala sesuatu yang ada diplanet bumi menjadi gambaran-gambaran umum yang lebih mudah dipahami dan dipelajari. Adapun tujuan pemodelan (dalam rangka pengembangan sistem/perangkat lunak aplikasi) sebagai sarana analisis, pemasahaman visualisasi dan komunikasi antar anggota tim pengembang, dalam Adi Nugroho (2010:6).

II.5.1. Pengenalan UML

UML sebagai *sebuah* bahasa yang memberikan *vocabulary* dan tatanan penulisan kata-kata dalam *'MS* Word' untuk kegunaan komunikasi. Sebuah bahasa model adalah sebuah bahasa yang mempunyai *vocabulary* dan konsep tatanan / aturan penulisan serta secara fisik mempresentasikan dari sebuah sistem. Seperti halnya *UML* adalah sebuah bahasa *standard* untuk pengembangan sebuah *software* yang dapat menyampaikan bagaimana membuat dan membentuk model-model, tetapi tidak menyampaikan apa dan kapan model yang seharusnya dibuat yang merupakan salah satu proses implementasi pengembangan *software.*

UML tidak hanya merupakan sebuah bahasa *pemograman visual* saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemograman, seperti *JAVA, C++, Visual Basic*, atau bahkan dihubungkan secara langsung ke dalam sebuah *object-oriented database*.

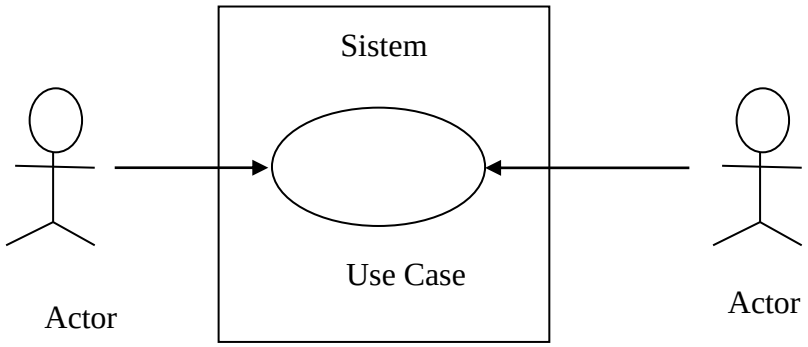
Begitu juga mengenai pendokumentasian dapat dilakukan seperti; *requirements, arsitektur, design, source code, project plan, tests,* dan *prototypes*. Untuk dapat memahami *UML* membutuhkan bentuk konsep dari sebuah bahasa model, dan mempelajari 3 (tiga) elemen utama dari *UML* seperti *building block*, aturan-aturan yang menyatakan bagaimana *building block* diletakkan secara bersamaan, dan beberapa mekanisme umum (*common*).

Alasan mengapa *UML* digunakan adalah, pertama, *scalability* dimana objek lebih mudah dipakai untuk menggambarkan sistem yang besar dan komplek. Kedua, *dynamic modeling*, dapat dipakai untuk pemodelan sistem dinamis dan *real time.* Sebagaimana dalam tulisan pertama, penulis menjelaskan konsep mengenai obyek, *OOA&D (Obyek Oriented Analyst/ Design)* dan pengenalan *UML*, maka dalam tulisan kedua ini lebih ditekankan pada cara bagaimana *UML* digunakan dalam merancang sebuah pengembangan software yang disertai gambar atau contoh dari sebuah aplikasi, dalam Adi Nugroho (2010).

Adapun jenis-jenis dari tipe diagram *UML* adalah sebagai berikut :

1. *Use Case Diagram*

Use *Case* adalah deskripsi fungsi dari sebuah sistem dari *perspektif* pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Model *use case* adalah bagian dari *requirement*. *Use case* adalah alat bantu terbaik guna menstimulasi pengguna potensial untuk mengatakan tentang suatu sistem dari sudut pandangnya. Dengan demikian diharapkan akan bisa dibangun suatu sistem yang bisa membantu pengguna, perlu diingat bahwa *use case* mewakili pandangan diluar sistem. Diagram *Use Case* menunjukkan 3 aspek dari sistem yaitu : *actor*, *use case dan system/sub system boundary*. *Actor* mewakili peran orang, *system* yang lain atau alat ketika berkomunikasi dengan *use case*. Berikut gambar notasi *use case :*



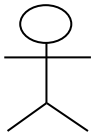
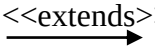
Gambar II.2. Notasi *Use Case* Diagram

(Sumber : Adi Nugroho ; 2010)

Berikut ini gambar *table* simbol-simbol dari *Use Case* Diagram adalah:

Tabel II.1 Simbol-simbol *Use Case Diagram*

Nama Komponen	**Deskripsi**	**Gambar**

Use Case	Menunjukkan proses yang terjadi pada sistem	
Actor	Menunjukkan user yang akan menggunakan sistem.	
Association	Sebuah garis yang berfungsi menghubungkan *Actor* dengan *Use Case*.	
Extend	Perluasan dari *Use Case* lain jika kondisi atau syarat terpenuhi.	
Include	Menjelaskan bahwa *Use Case* termasuk dalam *Use Case* lain.	

(Sumber : Adi Nugroho ; 2010)

2. *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akanmenghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok :

a. Nama (dan *stereotype*)

b. *Atribut*

c. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.

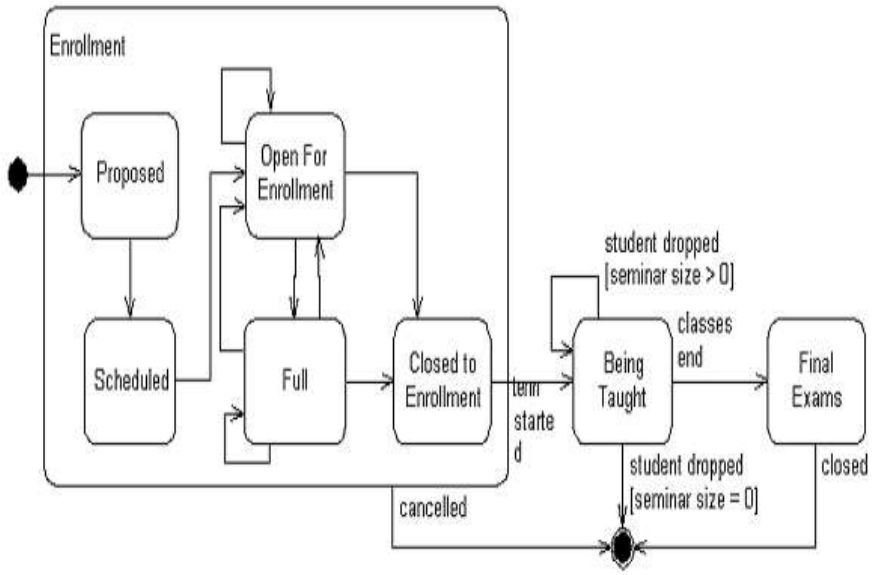
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.

c. *Public*, dapat dipanggil oleh siapa saja

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class abstrak* yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*.

3. *Statechart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).



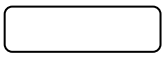
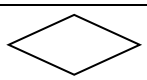
Gambar II.3 *Statechart Diagram*

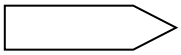
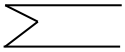
(Sumber : Adi Nugroho ; 2010)

4. *Activity Diagram*

Activity Diagram adalah teknik *untuk* mendiskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity Diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa. Berikut adalah simbol yang ada pada *activity diagram.*

Tabel II.2. Simbol Yang Ada Pada *Activity Diagram*

SIMBOL	KETERANGAN
	Titik Awal
	Titik Akhir
	*Activity*
	Pilihan untuk pengambilan keputusan.

	*Fork* : digunakan untuk menunjukkan kegiatan yang dilakukan secara 18cenario atau untuk menggabungkan dua kegiatan 18cenario menjadi satu.
	*Rake* ; menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran Akhir (*Flow Final*)

(Sumber : Adi Nugroho ; 2010)

5. *Sequence Diagram*

   Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display* , dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu.

6. *Collaboration Diagram*

   Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram* , tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*

7. *Component Diagram*

   Component Diagram mengandung *component, interface* dan *relationship.* Notasi *component Diagram* dapat dilihat seperti gambar di bawah ini :

Gambar II.4. Notasi *Component Diagram*

(Sumber : Adi Nugroho ; 2010)

8. *Deployment Diagram*

   Deployment Diagram menunjukkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware.* Sistem terdiri dari node-node dimana setiap node diwakili untuk sebuah kubus. Garis yang menghubungkan antara 2 kubus menunjukkan hubungan di antara kedua node tersebut. Tipe node bisa berupa *device* yang berwujud *hardware* dan bisa juga *processor*.

II.6. Pengertian Basis Data (*Database*)

Basis data merupakan kumpulan dari data-data yang saling terkait dan saling berhubungan satu dengan yang lainya. Basis data adalah kumpulankumpulan *file* yang saling berkaitan.

Database diartikan sebagai refresentasi fakta dunia nyata yang mewakili sebuah objek, misalnya manusia, hewan, barang, peristiwa, konsep dan lain sebagainya yang direkam dalam bentuk huruf, teks, symbol, angka, suara, gambar dan lainnya. Sedangkan basis data dapat diartikan sebagai tempat berkumpul, sarang atu gudang untuk menyimpan sesuatu. Dengan demikian basis data/database dapat diartikan sebagai tempat berkumpul, menyimpan data-data suatu benda atau kejadian yang saling berhubungan, dalam Wahana Komputer (2010:140).

Basis Data adalah kumpulan data yang saling berelasi. Data sendiri merupakan fakta mengenai obyek, orang, dan lain-lain. Data dinyatakan dengan nilai (angka, deretan karakter, atau simbol), dalam Kusrini (2010, p2).

Basis data dapat didefinisikan dalam berbagai susut pandang seperti berikut:

1. Himpunan kelompok data yang saling berhubungan yang diorganisasi sedemikian rupa sehingga kelak dapat dimanfaatkan dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (*redundancy*) yang tidak perlu, untuk memenuhi kebutuhan.
3. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpan elektronik.

II.6.1. Tujuan Basis Data

Basis data bertujuan untuk mengatur data sehingga diperoleh kemudahan, ketepatan dan kecepatan dalam pengambilan kembali. Untuk mencapat tujuan, syarat basisdata yang baik adalah sebagai berikut, dalam Kusrini (2010, p2) :

1. Tidak adanya redudansi dan inkonsistensi data

   Redudansis terjadi jika suatu informasi disimpan dibeberapa tempat. Misalnya ada data mahasiswa yang memuat nim, nama, alamat dan atribut lainya, sementara kita punya data lain tentang data KHS mahasiswa yang isinya terdapat NIM, nama, mata kuliah dan nilai. Pada kedua data tersebut kita temukan atribut nama.
2. Kesulitan pengaksesan data

   Basis data memiliki fasilitas untuk melakukan pencarian informasi dengan menggunakan query ataupun dari tool yang melibatkan tabelnya. Dengan fasilitas ini, bisa segera langsung melihat data dari software DBMNnya.
3. Multiple user

   Basis data memungkinkan penggunaan data secara bersama-sama oleh banyak pengguna pada saat yang bersamaan atau pada saat yang berbeda. Dengan meletakkan basis data pada

bagian server yang bisa diakses dari banyak client, sudah menyediakan akses kesemua pengguna dari komputer client ke sumber informasi yaitu basis data.

II.6.2. Manafaat/Kelebihan Basis Data

Banyak manfaat yang diperoleh dengan menggunakan basis data, Manafaat/Kelebihan Basis Data dan kelebihan basis data diantaranya adalah, dalam Kusrini (2010, p5) :

1. Kecepatan dan kemudahan

   Dengan menggunakan basis data pengambilan informasi dapat dilakukan dengan cepat dan mudah. Basis data memilki kemampuan dalam mengelompokkan, mengurutkan bahkan perhitungan dengan metematika. Dengan perancangan yang benar maka penyajian informasi dapat dilakukan dengan cepat dan mudah.

2. Kebersamaan pemakai (*sharability)*

   Sebuah basis data dapat digunakan oleh banyak user dan banyak aplikasi. Untuk data yang diperlukan oleh banyak bagian/orang, tidak perlu dilakukan pencacatan dimasing-masing bagian/orang, tetapi cukup dengan satu basis data untuk dipakai bersama.

3. Pemusatan kontrol data

   Karena cukup satu basis data untuk banyak keperluan, pengontrolan terhadap data juga cukup dilakukan disatu tempat saja.

4. Efesiensi ruang penyimpanan

   Dengan pemakaian bersama, tidak perlu menyediakan tempat penyimpanan diberbagai tempat tetapi cukup satu saja, sehingga ini dapat menghemat ruang penyimpanan yang dimiliki oleh sebuah organisasi.

5. Keakuratan (*Accuracy)*

Penerapan secara tepat acuan tipe data, domain data, keunikan data, hubungan antar data, dan lain-lain, dapat menekan ketidakakuratan dalam pemasukan / penyimpanan data.

6. Ketersediaan *(Availability)*

   Dengan basis data, semua data dapat dibackup, memilah-milah data mana yang masih diperlukan yang perlu disimpan ke tempat lain. Hal ini mengingat pertumbuhan transaksi sebuah organisasi dari lain waktu ke waktu membutuhkan penyimpanan yang semakin besar.

7. Keamanan *(Security)*

   Kebanyakan DBMS dilengkapi dengan fasilitas manajemen pengguna. Pengguna diberi hak akses yang berbeda-beda sesuai dengan kepentingan dan posisinya. Basis data bisa diberikan password untuk membatasi orang yang diaksesnya.

8. Kemudahan dalam pembuatan program aplikasi baru

   Penggunaan basis data merupakan bagian dari perkembangan teknologi. Dengan adanya basis data pembuatan aplikasi bisa memanfaatkan kemmapuan dari DBMS. Sehingga membuat aplikasi tidak perlu mengurusi penyimpanan data, tetapi cukup mengatur interface untuk pengguna.

9. Pemakaian secara langsung

   Basis data memiliki fasilitas yang lengkap untuk melihat datnya secara langsung dengan tools yang disediakan oleh DBMS.

10. Kebebasan data

    Perubahan dapat dilakukan pada level DBMS tanpa harus membongkar kembali program aplikasinya.

11. User View

    Basis data menyediakan pandangan yang berbeda-beda untuk tiap-tiap penguna.

II.6.3. Operasi Dasar Database

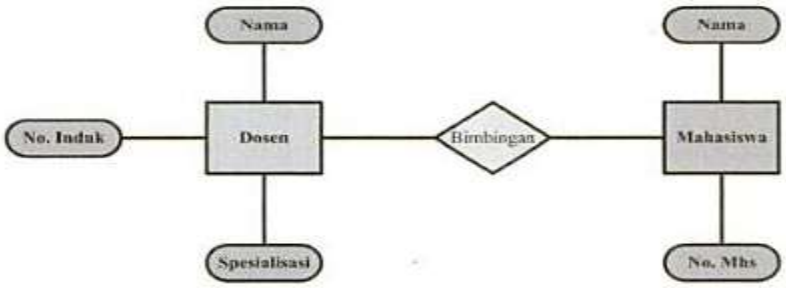
Menurut Kusrini (2010, p9), Beberapa operasi dasar basis data yaitu :

1. Pembuatan basis data
2. Penghapusan basis data
3. Pembuatan file/tabel
4. Penghapusan file/tabel
5. Pengubahan tabel
6. Penambahan/pengisian
7. Pengambilan data
8. Penghapusan data

II.6.4. Pemodelan Basis Data

Menurut Samiaji Sarosa (2010:4), Model diperlukan untuk mendapatkan penyederhanaan dari kenyataan dan memungkinkan desainer program program aplikasi bereksperimen dengan berbagai macam variable sebelum diaplikasikan ke system yang berjalan. Untuk merancang suatu aplikasi basis data alat yang biasa digunakan adalah *Entity Relationship Diagram (ERD).* ERD didasarkan dari artikel yang dipublikasikan oleh Peter Phin Shan Chen.

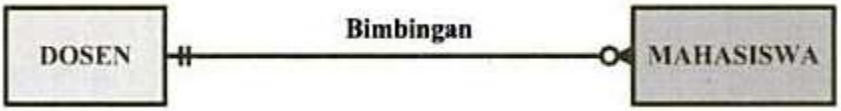
Ada beberapa case tool menamakan notasi ERD yang digunakan sebagain chen ERD. *Entit Relationship Model* adalah abstraksi konseptual yang mewakili struktur dari suatu basis data.



Gambar II.5. Diagram Dengan Notasi Chen ERD

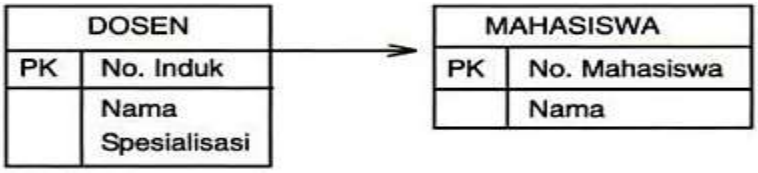
(Sumber : Adi Nugroho ; 2010)

Dalam perkembangannya banyak diciptakan notasi ERD yang berbeda-beda seperti telihat gambar dibawah ini.



Gambar II.6. Diagram Dengan Notasi Crows Foot

(Sumber : Adi Nugroho ; 2010)



Gambar II.7. Diagram Dengan Notasi Relational

(Sumber : Adi Nugroho ; 2010)

II.6.5. Normalisasi

Normalisasi adalah teknik yang dirancang untuk merancang tabel basis data relasional untuk meminimalkan duplikasi data dan menghindarkan basis data tersebut anomali. Suatu basis data dikatakan tidak normal jika terjadi 3 (tiga) anomali berikut, dalam Samiaji Sarosa (2010:5) :

1. *Insertion Anomaly*

Anomali yang terjadi jika ada data yang tidak bisa disisipkan kedalam table.

2. *Update/Modification anomaly*

   Anomali yang terjadi jika ada perubahan pada suatu item data maka harus mengubah lebih dari satu baris data.

Langkah-langkah normalisasi sampai pada bentuk 3NF adalah sebagai berikut :

1. *First Nornal Form (1NF)*

   Untuk menjadi 1NF suatu table harus memenuhi dua syarat. Syarat pertama tidak ada kelompok data atau *field* yang berulang. Syarat kedua harus ada *primary key (PK)* atau kunci unik, atau kunci yang membedakan satu bari dengan baris yang lain dalam satu table. Pada dasarnya sebuah table selamat tidak ada kolom yang sama merupakan bentuk table dengan 1NF.

2. *Second Nornal Form (2NF)*

   Untuk menjadi 2NF suatu table harus berada dalam kondisi 1NF dan tidak memilik *partial dependencies. Partial dependencies* adalah suatu kondisi jika atribut non kunci (Non PK) tergantung sebagian tetapi bukan seluruhnya pada PK.

3. *Third Nornal Form (3NF)*

   Untuk menjadi 3NF suatu table harus berada dalam kondisi 2NF dan tidak memilik *transitive dependencies. Transitive dependencies* adalah suatu kondisi dengan adanya ketergantunga fungsional antara 2 atau lebih atribut non kunci (Non PK).

II.7. Visual Basic 2010

Visual Basic adalah bahasa pemrogram klasik, legendaris yang paling banyak dipakai oleh programmer didunia. Pemograman ini dipakai oleh jutaan programmer dan tercatat sebagai program yang paling disukai oleh mayoritas orang, dalam Edi Winarno ST, M.Eng dkk (2010:1).

Visual Studio 2010 pada dasarnya adalah sebuah bahasa pemrograman komputer. Dimana pengertian dari bahasa pemrograman itu adalah perintah-perintah atau instruksi yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. Visual Studio 2010 selain disebut dengan bahasa pemrograman, juga sering disebut sebagai sarana (tool) untuk menghasilkan program-progam aplikasi berbasiskan windows.

Beberapa kemampuan atau manfaat dari Visual Studio 2010 diantaranya seperti :

1. Untuk membuat program aplikasi berbasiskan windows.
2. Untuk membuat objek-objek pembantu program seperti, misalnya : kontrol ActiveX, file Help, aplikasi Internet dan sebagainya.
3. Menguji program (debugging) dan menghasilkan program berakhiran EXE yang bersifat executable atau dapat langsung dijalankan.

II.7.1. Antar Muka Visual Basic 2010

Saat menjalankan Visual Basic 2010 pertama kali muncul jendela *chose default environtment settings.* Disini bisa memilih apakah ingin memilih antar muka di Visual Studio. Untuk *programmer* Visual Basic lebih baik memilih *Visual Basic Development Centre.*

Gambar II.8 Form Chose Default Environtment Settings

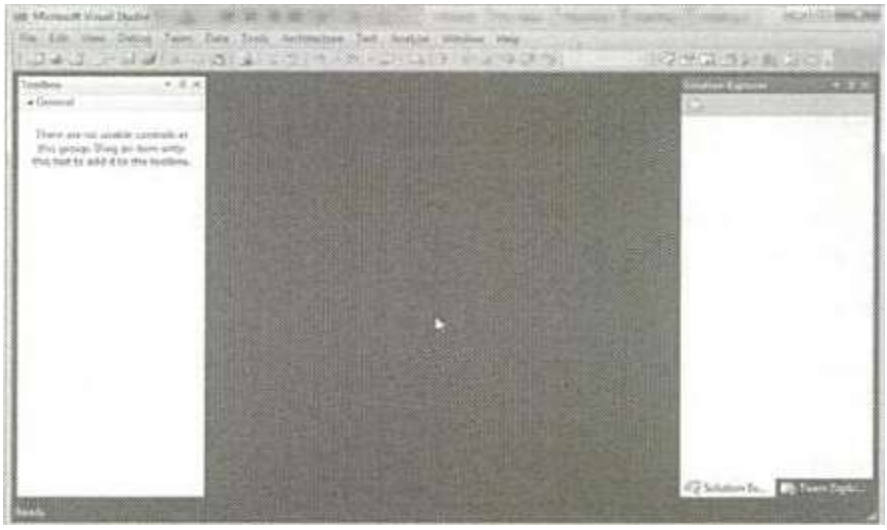
(Sumber Edi Winarno, dkk ; 2010)

Dibagian awal visual basic, bisa memilih *Start Page*. *Start Page* adalah halaman yang mencantumkan informasi-informasi seputar program dan juga informasi RSS dari sumber tertentu. Jika tidak ingin menampilkan hal ini hilangkan tanda centang pada *Show Page On Startup.*

Gambar II.9. Start Page Visual Basic 2010

(Sumber Edi Winarno, dkk ; 2010)

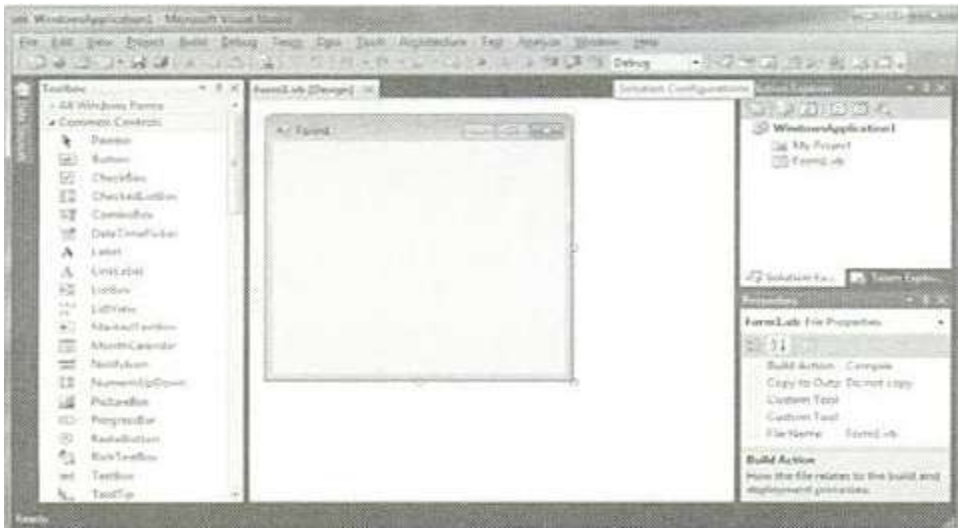
Jika *start page* ditutup terlihat tampilan sebagai berikut :



Gambar II.10. Tampilan IDE (*Integrated Development Environtment)* setelah *Start Page* ditutup

(Sumber Edi Winarno, dkk ; 2010)

Jika ada sebuah form yang terlihat, tampilan lengkap IDE seperti gambar berikut ini.



Gambar II.11. Tampilan lengkap IDE

(Sumber Edi Winarno, dkk ; 2010)

Komponen-komponen dari IDE adalah :

1. Dibagian kiri terdapat toolbox yang menampilkan semua objek tool yang bisa dimasukkan kedalam form untuk membuat program.
2. Dibagian tengah terdapat tempat meletakkan form dan kode, baik disaat desain ataupun pada saat program dijalankan.
3. Dibagian kanan terdapat solution explorer yang merupakan explorer untuk melihat file-file disebuah objek.
4. Dikanan bawah terdapat propertis untuk melihat properti dari nilai-nilai pada objek yang dipilih dibagian tengah. (Edi Winarno ST, M.Eng dkk, 2010:1)

II.8. SQL Server 2008 *Express Edition*

SQL Server 2008 *Express Edition* sebuah terobosan baru dalam bidang database, SQL Server adalah sebuah DBMS (*Database Management System)* yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQL Server 2008 *Express Edition* dibuat pada saat kemajuan dalam bidang hardware semakin pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 *Express Edition* membawa terobosan dalam bidang pengolahan dan penyimpanan data, dalam Wahana Komputer (2010:2).

II.8.1 Kebutuhan *Hardware*

Adapun *hardware* yang diperlukan untuk instalasi SQL Server 2008 *Express Edition* minimal adalah sebagai berikut:

1. Processcor minimal 1 GHz
2. Memori minimal 512 MB
3. Sistem Operasi Windows

Biar dapat diinstal pada sistem komputer dengan memoti 512 MB, tetapi disarankan menggunakan memori 1 GB. Sedangkan untuk jaringannya diperlukan adalah :

1. *Sharer Memory*
2. TCP/IP
3. *Named Pipes*
4. *Virtual Interface Adapter (VIA)*(Wahana Komputer, 2010:2).

II.8.2 Versi SQL Server 2008 *Express Edition*

Microsoft merilis SQL Server 2008 *Express Edition* dalam beberapa versi yang disesuaikan dengan segmen-segmen pasar yang dituju. Versi-versi tersebut adalag sebagai berikut :

a. Menurut cara pemrosesan data pada prosesor makan Microsoft mengelompokkan produk ini berdasarkan dua jenis yaitu :
    1. Versi 32 Bit (x86), yang biasanya digunakan untuk komputer *single processor* (Pentium 4) atau lebih tepatnya processor 32 bit atau Windows XP
    2. Versi 64 Bit (x64), yang biasanya digunakan oleh computer yang lebih sari satu processor (Misalnya *Core 2 duo)* dan system operasi 64 bit, Vista dan Windows 7.
b. Sedangkan secara keseluruhan terdapat versi-versi seperti berikut :
    1. Versi *Compact* ini adalah versi "tipis" dari semu versi yang ada
    2. Versi *Express* ini adalah versi "ringan"

II.8.3 Instalasi SQL Server 2008 *Express Edition*

Proses instalasi SQL Server 2008 *Express Edition* tidak sama dengan instalasi versi-versi sebelumnya. Proses SQL Server 2008 *Express Edition* agak panjang melalui beberapa tahapan. Tahapan yang dilakukan akan membawa beberapa pilihan yang akan diisi dalam *setting* sebuah *server database*. Berikut ini adalah pilihan-pilihan yang akan dijumpai dalam proses instalasi SQL Server 2008 *Express Edition.*

1. Tempat direktori utama dan penyimpanan file database

   Direktori utama adalah adalah direktori dimana semua file program akan ditempatkan dan file-file tersebut tidak akan berubah selama anda menjalankan SQL server. Direktori utama secara standard akan berada dalam direktori "C:\Program Files\Microsoft SQL Server".
2. Penggunaan *Multiple instance*

Instance adalah sebuah turunan dari server database SQL Server. Karena sebuah tiruan maka sebuah *Instance* memiliki fungsi yang sama dengan database server aslinya. Arti sebenarnya *Instance SQL Server* adalah sebuah server database yang tidak men-*sharing* sistemnya dan database user dengan database server lainnya yang ada dalam komputer yang sama.

3. Jasa *Autentification User* (Menggunakan Windows atau mixed)

   Autentification User diperlukan supaya server tidak dapat dipergunakan oleh orang yang tidak bertanggungjawab dan tidak berhak. Dalam SQL server ada dua *Autentification User* yang dapat digunakan yaitu :

   a. Mode Windows, Pada mode ini SQL Server akan melakukan autentifikasi dengan menggunakan level login pada system operasi.
   b. Mode Mixel atau campuran, mode ini menginjinkan *user* untuk masuk kedalam sistem SQL server dengan menggunakan *Account* yang dibuat di sistem operasi windows atau juga menggunakan *account* yang di *set up* pada SQL Server (Wahana Komputer, 2010:2)