BAB II

LANDASAN TEORI

II.1. Teori Sistem

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

a. 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.
b. 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

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

a. 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.

b. Batasan sistem (*Boundary*)

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

c. 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

d. 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.

e. Masukan sistem (*Input*)

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

f. Keluaran sistem (*Output*)

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

g. Pengolah sistem (*Process*)

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

h. 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

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

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

Menurut Dicky Nofriansyah (2014:1), Sistem pendukung keputusan (SPK) dibangun untuk mendukung solusi atas suatu masalah atau untuk suatu peluang. Aplikasi Sistem pendukung keputusan (SPK) digunakan untuk pengambilan keputusan. Aplikasi Sistem pendukung keputusan (SPK) menggunakan *CBIS (Computer Based Information System)* yang *fleksibel*, *interaktif* dan dapat diadaptasi yang dikembangkan untuk mendukung solusi masalah manajemen spesifik yang tidak terstruktur.

Menurut Bonczek dkk, (1980) dalam buku "*Decission Support System and intelligent system* (Turban 2005:137) mendefinisikan Sistem pendukung keputusan (SPK) sebagai sistem berbasis komputer yang terdiri dari tiga komponen yang saling berinteraksi, sistem bahasa (mekanisme untuk memberikan komunikasi antara pengguna dan komponen sistem pendukung keputusan lain), sistem pengetahuan (repository pengetahuan domain masalah yang ada pada sistem pendukung keputusan (SPK) atau sebagai data atau sebagai prosedur), dan sistem pemrosesan masalah (hubungan antara dua komponen lainnya, terdiri dari satu atau lebih kapabilitas manipulasi masalah umum yang diperlukan untuk pengambilan keputusan).

II.2.2. Karakteristik Sistem Pendukung Keputusan

Menurut Dicky Nofriansyah (2014:2), Karakteristik Sistem pendukung keputusan (SPK) yaitu :

a. Mendukung proses pengambilan keputusan suatu organisasi atau perusahaan
b. Adanya *interface* manusia/mesin dimana manusia *(user)* tetap memegang kontrol proses pengambilan keputusan.
c. Mendukung pengambilan keputusan untuk membahas masalah terstruktur, semi terstruktur serta mendukung beberapa keputusan yang saling berinteraksi.
d. Memiliki kapasitas dialog untuk memperoleh informasi sesuai dengan kebutuhan.
e. Memiliki subsistem yang terintegrasi sedemikan rupa sehingga dapat berfungsi sebagai kesatuan sistem.

II.2.2. Ciri-Ciri Sistem Pendukung Keputusan

Menurut Dicky Nofriansyah (2014:2), Kriteria atau ciri-ciri sistem pendukung keputusan adalah sebagai berikut :

a. Banyak pilihan/alternatif
b. Ada kendala atau surat
c. Mengikuti suatu pola atau model tingkah laku, baik yang terstruktur maupun tidak terstruktur.
d. Banyak *input/Variabel*
e. Ada faktor resiko, dibutuhkan kecepatan, ketepatan dan keakuratan

II.2.3. *Fase* Dalam Sistem Pendukung Keputusan

Menurut Dicky Nofriansyah (2014:2), Tiga *fase* dalam pengambilan keputusan yaitu :

a. Inteligence

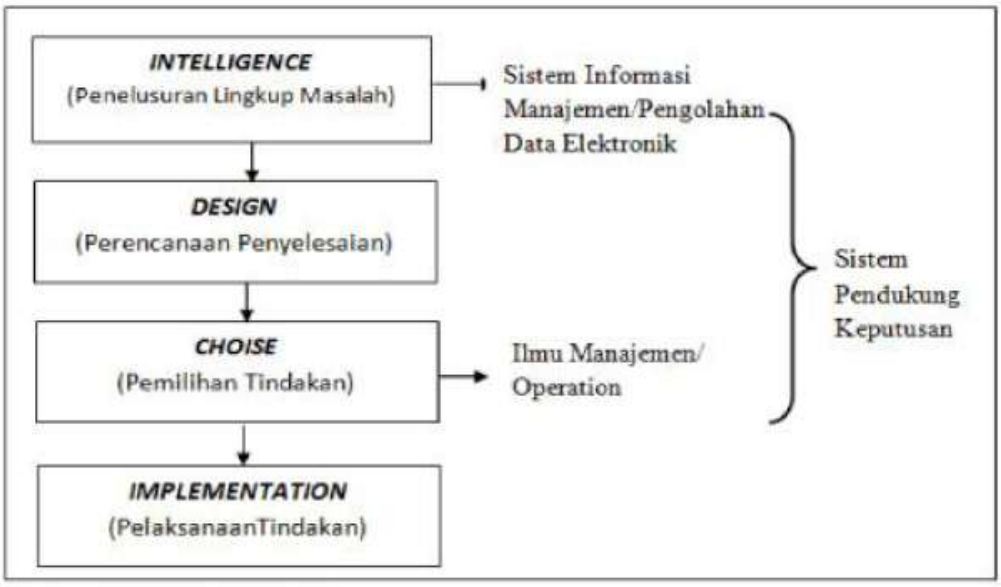
Tahap ini merupakan proses penelusuran dan pendeteksian dari ruang lingkup problematika secara proses pengenalan masalah. Data masukan diperoleh dan diuji dalam rangka mengidentifikasi masalah.

b. Design

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

c. Choice

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



Gambar 2.1 Fase Proses Pengambilan Keputusan

(Sumber : Dicky Nofriansyah ; 2014)

II.2.4 Komponen Sistem Pendukung Keputusan

Menurut Dicky Nofriansyah (2014:3),Secara garis besar sistem pendukung keputusan dibangun oleh tiga komponen utama yaitu :

a. 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)*

b. 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 harsu 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.

c. 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.2.5. Tujuan Sistem Pendukung Keputusan

Menurut Dicky Nofriansyah (2014:4), Tujuan dari system pendukung keputusan adalah sebaagi berikut :

a. Membantu dalam pengambilan keputusan atas masalah yang terstruktur
b. Memberikan dukungan atas pertimbangan manager dan bukannya dimaksudkan untuk menggantikan fungsi manager.
c. Meningkatkan efektifitas keputusan yang diambil lebih dari perbaikan efesiensinya.
d. Kecepatan komputasi komputer memungkinkan para pengambil keputusan untuk banyak melakukan komputasi secara cepat dengan biaya yang sangat rendah.
e. 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 hukum) bisa ditingkatkan. Produktivitas juga ditingkatkan menggunakan peralatan optimalisasi yang menjalankan sebuah bisnis.

III.3. Metode *Algoritma Iteraktive Dichotomizer Three (ID3)*

Menurut Yohannes Maruli Sitanggang (2014:41), *Iterative Dichotomizer 3* adalah algoritma *decision tree learning* (algoritma pembelajaran pohon keputusan) yang paling dasar. Algoritma ini melakukan pencarian secara rakus/menyeluruh (*greedy*) pada semua kemungkinan pohon keputusan. Salah satu algoritma induksi pohon keputusan yaitu (*Iterative Dichotomizer* 3).

Metode *Iterative Dichotomizer* 3 dapat diimplementasikan menggunakan fungsi rekursif (fungsi yang memanggil dirinya sendiri). Algoritma berusaha membangun *decision tree* (pohon keputusan) secara *top-down* (dari atas ke bawah), mulai dengan pertanyaan : "atribut mana yang pertama kali harus dicek dan diletakkan pada *root*?" pertanyaan ini dijawab dengan mengevaluasi semua atribut yang ada dengan menggunakan suatu ukuran statistik (yang banyak digunakan adalah *information gain*) untuk mengukur efektivitas suatu atribut dalam mengklasifikasikan kumpulan *sampel* data.

Karakteristik *Iterative Dichotomizer 3* dalam membangun pohon keputusan adalah secara *topdown* dan *divide and conquer*. *Top-down* artinya pohon keputusan dibangun dari simpul akar ke daun, sementara *divide and conquer* artinya *training data* secara rekursif dipartisi ke dalam bagian-bagian yang lebih kecil saat pembangunan pohon.

Decision Tree adalah sebuah struktur pohon, dimana setiap *node* pohon merepresentasikan atribut yang telah diuji, setiap cabang merupakan suatu pembagian hasil uji, dan *node* daun (*leaf*) merepresentasikan kelompok kelas tertentu. *Level node* teratas dari sebuah *Decision Tree* adalah node akar (*root*) yang biasanya berupa atribut yang paling memiliki pengaruh terbesar pada suatu kelas tertentu. Pada umumnya *Decision Tree* melakukan strategi pencarian secara *top-down* untuk solusinya.

Pada proses mengklasifikasi data yang tidak diketahui, nilai atribut akan diuji dengan cara melacak jalur dari *node* akar (*root*) sampai *node* akhir (daun) dan kemudian akan diprediksi kelas yang dimiliki oleh suatu data baru tertentu.

III.3.1. Pohon (Tree)

Menurut Yohannes Maruli Sitanggang (2014:41), Pohon merupakan sebuah *graf* terhubung yang tidak mengandung sirkuit. Konsep pohon *(tree)* dalam teori *graf* merupakan konsep yang sangat penting, karena terapannya diberbagai bidang ilmu. Oleh karenanya antara pohon *(tree)* sangat erat hubungannya dengan teori *graf.*

Definisi pohon adalah *graf* tak berarah terhubung yang tidak mengandung sirkuit, menurut definisi tersebut, ada dua sifat penting pada pohon yaitu terhubung dan tidak mengandung sirkuit. Pohon *(tree)* merupakan *graf* dimana dua simpul memiliki paling banyak satu lintasan yang menghubungkannya. Pohon seringkali memiliki akar karena setiap simpul pada pohon hanya memiliki satu lintasan akses dari setiap simpul lainnya, maka tidak mungkin bagi sebuah lintasan untuk membentuk simpul (*loop*) atau siklus (*cycle*) yang secara berkesinambungan melalui serangkaian simpul.

Pohon (*tree*) banyak digunakan dalam bidang pendidikan yang selalu memecahkan masalah dengan cepat. Metode ini merupakan metode yang berusaha menemukan fungsi-fungsi pendekatan yang bernilai diskrit dan tahan terhadap data-data yang terdapat kesalahan (*noisy data*) serta mampu mempelajari ekspresi-ekspresi *disjunctive* (ekspresi pohon keputusan (*decision tree*) dan aturan-aturan keputusan (*rule*).

Perhitungan *Algoritma Iteraktive Dichotomizer Three (ID3)* dapat dihitung dengan persamaan dibawah ini.

Entropy(S) = - p+ log2p + -p -log2p

Dimana :

S adalah ruang (data) sampel yang digunakan untuk training.

P adalah jumlah data sampel untuk criteria

Information Gain

$$InformationGain = Entropy(S) - \sum_{v\in} * Entropy(Si)$$

dimana :

A : atribut

V : menyatakan suatu nilai yang mungkin untuk atribut A

Values(A) : himpunan yang mungkin untuk atribut A

|Sv| : jumlah sampel untuk nilai v

|S| : jumlah seluruh sampel data

Entropy(Sv) : *entropy* untuk *sampel-sampel* yang memilki nilai v

II.4. *Unified Modeling Language* (*UML)*

Adi Nugroho (2010:6), *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.

II.4.1. Pengenalan *UML*

Menurut Julius Hermawan (2010:7), UML (*Unified Modeling Language)* adalah bahasa *standard* yang digunakan untuk menjelaskan dan memvisualisasikan artifak dan proses analisis dan desain berorientasi objek. *UML* Menyediakan *standart* pada notasi dan diagram yang bias digunakan untuk memodelkan suatu sistem. *UML* dikembangkan oleh tiga pendekar "berorientasi objek" yaitu Gradi Booch, Jim Rumbaugh dan Ivar Jacobson. *UML* menjadi bahasa yang bias digunakan untuk berkomunikasi dalam *prespektif* objek antara user dengan *developer*, antara *developer* analisis dengan *developer* desain dan antara *developer* desain dengan *developer* pemrograman.

UML memungkin *developer* melakukan pemodelan secara *visual*, yaitu penekanan pada penggambaran, bukan didominasi oleh narasi. Pemodelan *visual* membantu untuk menangkap struktur dan kelakuan dari si objek, mempermudah

penggambaran interaksi antara elemen dalam sistem dan mempertahankan konsistensi antara desain dan implementasi dalam bahasa pemrograman.

Namun karena *UML* hanya merupakan bahasa pemodelan maka UML bukanlan rujukan bagaimana melakukan analisis dan desain berorientasi objek. Untuk mengetahui bagaimana melakukan analisis dan desain berorientasi objek secara baik, sudah terdapat beberapa metodologi yang bias diikuti.

II.4.2. Notasi dan Artifak dalam *UML*

Menurut Julius Hermawan (2010:13), *UML* menyediakan beberapa notasi dan artifak *standard* yang bias digunakan sebagai alat komunikasi bagi para proses analisis dan desain. Artifak didalam *UML* didefenisikan sebagai informasi dalam berbagai bentuk yang digunakan atau dihasilkan dalam proses pengembangan perangkat lunak.

1. Aktor

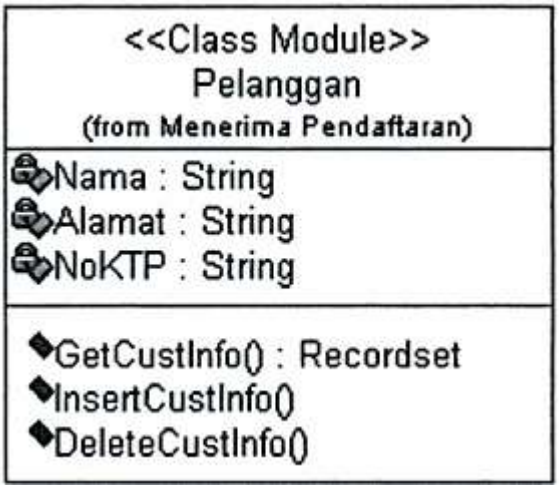
   Aktor adalah segala sesuatu yang berinteraksi dengan sistem aplikasi computer. Jadi aktor ini bisa berupa orang, perangkat keras atau juga objek lain dalam sistem yang sama. Biasanya yang dilakukan oleh aktor adalah memberikan informasi pada sistem dan\atau memerintahkan sistem untuk melakukan sesuatu.

Gambar II.2. Notasi Aktor

(Sumber : Julius Hermawan ; 2010)

2. *Class*

   Class merupakan pembentuk utama dari sistem berorientasi objek karena *class* menunjukan kumpulan objek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface*



Gambar II.3. Notasi Class

(Sumber : Julius Hermawan ; 2010)

   Class digunakan untuk mengabstraksikan elemen-elemen dari sistem yang dibangun. *Class* bisa untuk direpresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata. Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas direpresentasikan informasi yang disimpan didalamnya. Operasi menunjukkan sesuatu yang bisa dilakukan oleh objek, dan menggunakan kata kerja.

3. *Interface*

   Interface merupakan kumpulan informasi tanpa implementasi dari suatu *class*. Implementasi operasi dari suatu *interface* dijabarkan oleh operasi didalam *class.* Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang

mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam objek.

Gambar II.4. Notasi Interface

(Sumber : Julius Hermawan ; 2010)

4. *Use Case*

Use case menjelaskan urutan kegiatan yang dilakukan aktor dan sistem untuk mencapai tujuan tertentu. Walaupun menjelaskan kegiatan namun *use case* hanya menjelaskan apa yang dilakukan oleh aktor dan sistem, bukan bagaimana sistem melakukan kegiatan tersebut.

Gambar II.5 Notasi *Use Case*

(Sumber : Julius Hermawan ; 2010)

Didalam *use case* terdapat teks untuk menjelaskan urutan kegiatan yang disebut *use case specification. use case specification* terdiri dari :

a. Nama *Use Case*

Mencantumkan nama dari *use case* yang bersangkutan. Sebaiknya diawali dengan kata kerja untuk menunjukkan suatu aktivitas

b. Deskripsi singkat

Menjelaskan secara singkat dalam 1 atau 2 kalimat tentang tujuan dari *use case* ini.

c. Aliran Normal (*Basic Flow)*

Ini adalah jantung dari *use case*. Menjelaskan interaksi antara *actor* dan sistem dalam kondisi normal, yaitu segala sesuatu berjalan dengan lancar tiada halangan atau hambatan dalam mencapai tujuan dalam *use case*.

d. Aliran Alternatif *(Alternative Flow)*

Merupakan pelengkap dari *basic flow* tidak ada yang sempurna dalam setiap kali *use case* berlangsung. Didalam *Alternative Flow* ini dijelaskan dalam apa yang terjadi bila suatu halangan atau hambatan terjadi sewaktu *use case* berlangsung. Ini terutama berhubungan dengan *error* yang mungkin terjadi terutama karena sistem kekurangan data untuk diolah.

e. Special Requirement

Berisi kebutuhan lain yang belum tercakup dalam kebutuhan normal dan alternatif. Biasanya secara tegas dibedakan bahwa *basic flow* dan *alternate flow* menangani kebutuhan fungsional dari *use case* sementara *Special Requirement* yang tidak berhubungan dengan kebutuhan fungsional, misalnya kecepatan transaksi maksimum artinya berapa cepat dan berapa lama, kapasitas akses yaitu jumlah *user* yang akan mengakses dalam waktu bersamaan.

f. Pre-Condition

Menjelaskan persyaratan yang harus dipenuhi sebelum *use case* bisa dimulai.

g. Post-Condition

Menjelaskan kondisi yang berubah atau terjadi saat *use case* selesai dieksekusi.

5. *Interaction*

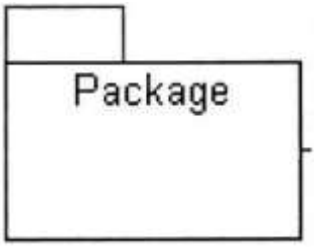
   Digunakan untuk menunjukan baik aliran pesan maupun informasi antara objek maupun antara hubungan objek. Biasanya *Interaction* dilengkapi juga dengan *teks* bernama *operation signature* yang tersusun dari mana operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.

Gambar II.6. Notasi *Interaction*

(Sumber : Julius Hermawan ; 2010)

6. *Package*

   Package adalah kontainer atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang dibangun, sehingga bisa dibuat model menjadi lebih sederhana. Tujuannya adalah untuk mempermudah pengelihatan dari model yang sedang dibangun.



Gambar II.7. Notasi *Packag*

(Sumber : Julius Hermawan ; 2010)

7. *Note*

   Note digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. *Note* ini bisa ditempelkan ke semua elemen notasi yang lain.

8. *Dependency*

   Dependency merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen member pengaruh pada elemen lain.

Gambar II.8. Notasi *Dependency*

(Sumber : Julius Hermawan ; 2010)

II.5. Pengertian Basis Data (*Database*)

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

Menurut Wahana Komputer (2010:140), 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.

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

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.5.1. Tujuan Basis Data

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

a. 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.

b. 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.

c. *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.5.2. Manafaat/Kelebihan Basis Data

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

a. 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.

b. 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.

c. Pemusatan kontrol data

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

d. 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.

e. 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.

f. 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.

g. 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.

h. 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.

i. Pemakaian secara langsung

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

j. Kebebasan data

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

k. User View

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

II.5.3. Operasi Dasar Database

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

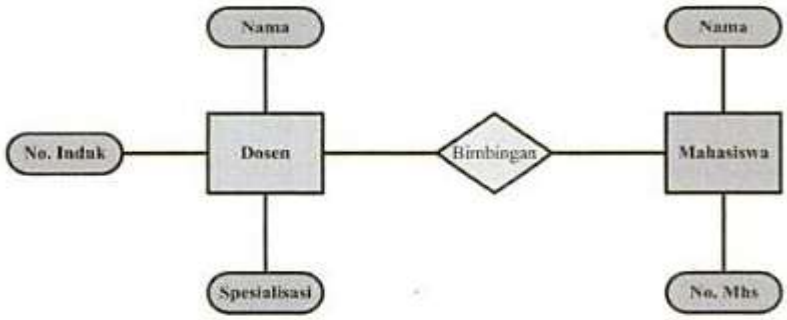
a. Pembuatan basis data
b. Penghapusan basis data
c. Pembuatan *file/*tabel
d. Penghapusan *file/*tabel
e. Pengubahan tabel
f. Penambahan/pengisian
g. Pengambilan data
h. Penghapusan data

II.5.4. 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 sistem 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*. *Entity Relationship Model* adalah abstraksi konseptual yang mewakili struktur dari suatu basis data.



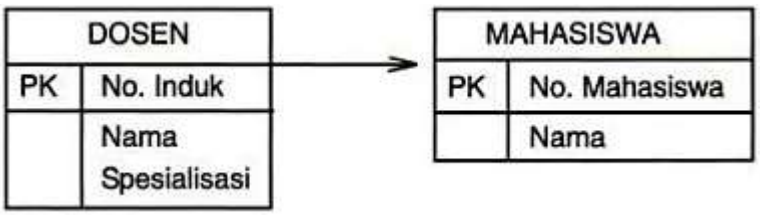
Gambar II.9. Diagram Dengan Notasi *Chen ERD*

(Sumber : Samiaji Sarosa ; 2010)

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

Gambar II.10. Diagram Dengan Notasi *Crows Foot*

(Sumber : Samiaji Sarosa ; 2010)



Gambar II.11. Diagram Dengan Notasi *Relational*

(Sumber : Samiaji Sarosa ; 2010)

II.5.5. Normalisasi

Menurut Samiaji Sarosa (2010:5), 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 :

a. *Insertion Anomaly*

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

b. *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 :

a. First Nornal Form (1NF)

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

b. Second Nornal Form (2NF)

Untuk menjadi *2NF* suatu tabel 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*.

c. 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.6. *Visual Basic* 2010

Menurut Edi Winarno (2010:1), *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.

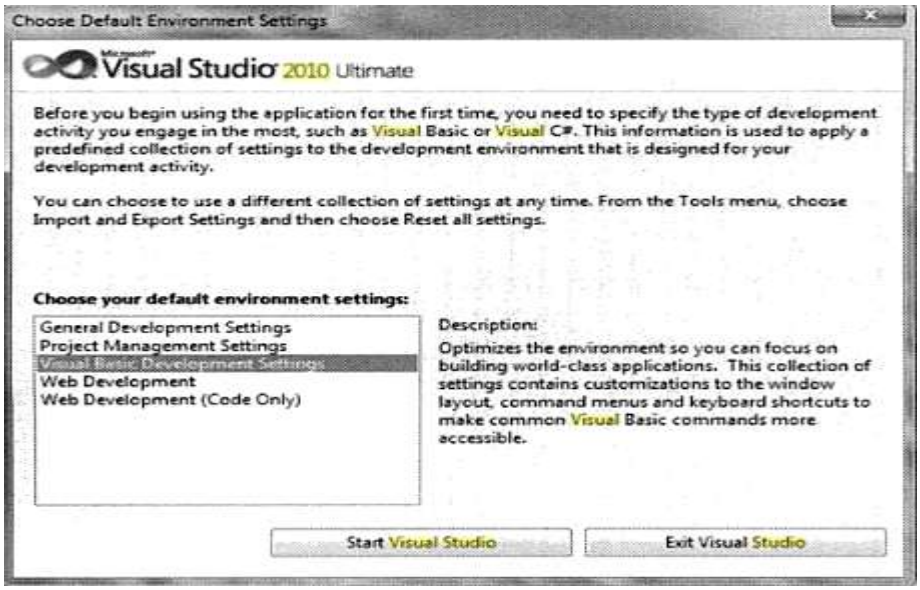
Visual Studio 2010 pada dasarnya adalah sebuah bahasa pemrograman komputer. Dimana pengertian dari bahasa pemrograman itu adalah perintah-perintah atau instruksi yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. *visual studio* 2010 selain disebut dengan bahasa pemrograman, juga sering 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.6.1. Antar Muka *Visual Basic* 2010

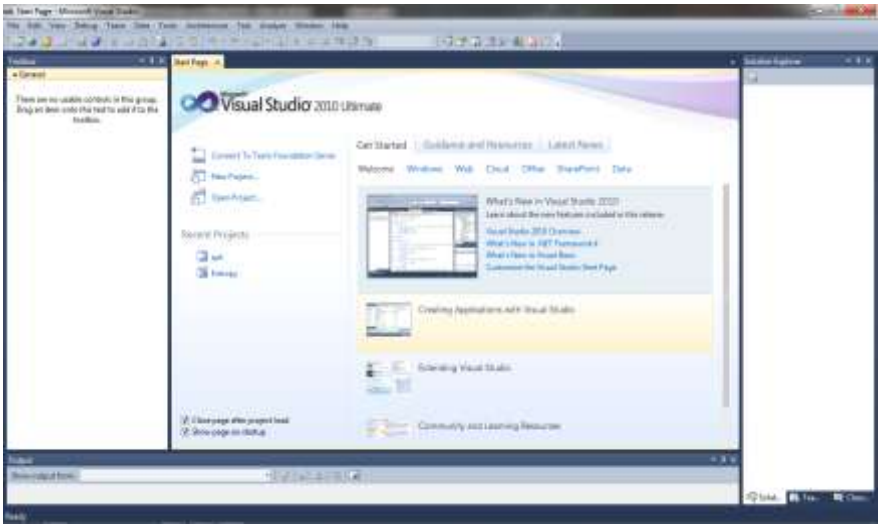
Saat menjalankan *Visual Basic* 2010 pertama kali muncul jendela *chose default environtment settings.* Disini bisa memilih apakah ingin memilih antar muka di *Visual Studio*. Untuk *programmer visual basic* lebih baik memilih *Visual Basic Development Centre.*



Gambar II.12 Form Chose Default Environtment Settings

(Sumber : Edi Winarno ; 2010)

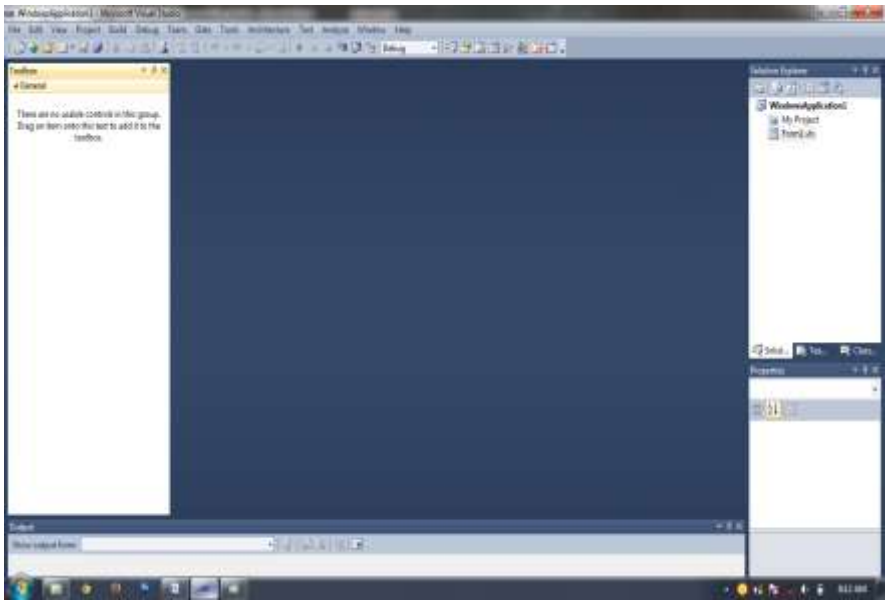
Dibagian awal visual basic, bisa memilih *Start Page*. *Start Page* adalah halaman yang mencantumkan informasi-informasi seputar program dan juga informasi. Jika tidak ingin menampilkan hal ini hilangkan tanda centang pada *Show Page On Startup.*



Gambar II.13. *Start Page Visual Basic* 2010

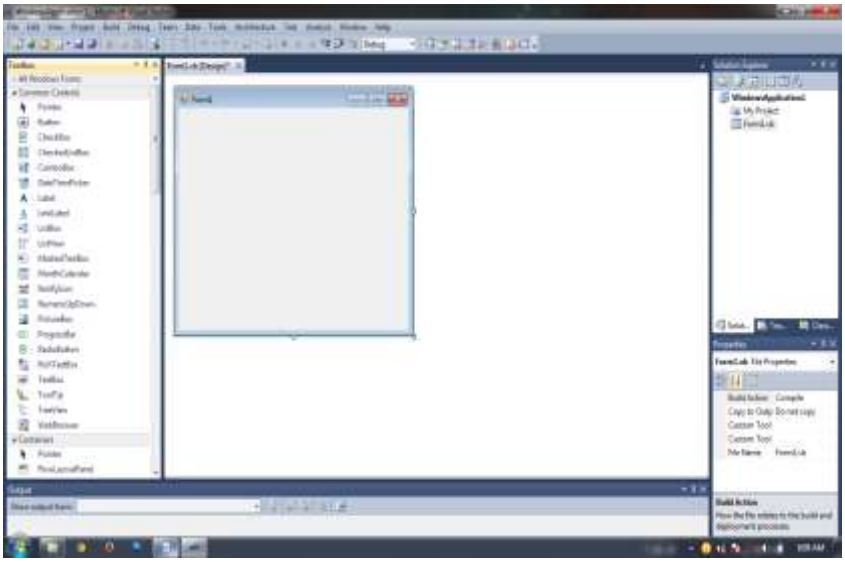
(Sumber : Edi Winarno ; 2010)

Jika *start page* ditutup terlihat tampilan sebagai berikut :



Gambar II.14. Tampilan *IDE* (*Integrated Development Environtment)* setelah *Start Page* ditutup
(Sumber : Edi Winarno ; 2010)

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



Gambar II.15. Tampilan lengkap IDE

(Sumber : Edi Winarno ; 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, dkk, 2010:1)

II.7. *SQL Server* 2008 *Express Edition*

Menurut Wahana Komputer (2010:2), *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.

II.7.1 Kebutuhan *Hardware*

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

a. *Processco*r minimal 1 *GHz*
b. Memori minimal 512 *MB*
c. Sistem Operasi *Windows*

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

a. *Sharer Memory*
b. TCP/IP
c. *Named Pipes*
d. *Virtual Interface Adapter (VIA)*(Wahana Komputer, 2010:2).

II.7.2. Versi SQL Server 2008 *Express Edition*

Microsoft merilis *SQL Server* 2008 *Express Edition* dalam beberapa versi yang disesuaikan dengan segmen-segmen pasar yang ditujuh. 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 komputer yang lebih sari satu *processor* (Misalnya *Core 2 duo)* dan sistem 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.7.3. 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 bertanggung jawab 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 sistem 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)

II.8. *Client Server*

Prasetyo (2004), Aplikasi database client-server merupakan suatu aplikasi yang melibatkan beberapa entitas, yaitu aplikasi *client* dan aplikasi *server*. Dalam aplikasi *client-server*, terjadi pembagian tugas antara komputer *client* dan komputer *server*. Komputer client digunakan untuk melakukan permintaan, sedangkan komputer *server* berfungsi untuk mengolah permintaan dari *client* dan mengembalikan hasilnya pada *client* yang meminta. Adanya pembagian tugas ini akan dapat mengurangi lalu lintas data didalam jaringan.

Sistem *client-server* merupakan sistem yang paling baik untuk digunakan , sistem ini mampu menghasilkan aplikasi database yang tangguh dalam hal sekuritas, serta mampu mengurangi kepadatan lalu-lintas jaringan.