BAB II

LANDASAN TEORI

2.1 Konsep Dasar Sistem

Menurut JOG[7], terdapat dua kelompok pendekatan di dalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan sistem yang lebih menekankan pada prosedurnya mendefinisikan sistem sebagai berikut :

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

Pendekatan sistem yang lebih menekankan pada elemen atau komponennya mendefinisikan sistem sebagai berikut :

> *"Sistem adalah kumpulan dari elemen-elemen yang saling berinteraksi untuk mencapai suatu tujuan tertentu".*

Sistem itu sendiri memiliki karakteristik atau beberapa sifat tertentu, yaitu mempunyai komponen (*components*), batas sistem (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*), dan sasaran (*objectives*) atau suatu tujuan (*goal*).

Adapun penjelasan dari karateristik suatu sistem adalah sebagai berikut:

a. Komponen sistem (*Components*) adalah bagian sistem yang saling ber-interaksi dan membentuk satu kesatuan. Komponen atau elemen sistem dapat berupa subsistem atau beberapa bagian sistem.

b. Batas sistem (*Boundary*) adalah daerah yang membatasi antara suatu sistem dengan lingkungannya atau dengan sistem lainnya. Batasan sistem inilah yang membuat sistem dipandang sebagai satu kesatuan.

c. Lingkungan luar sistem (*Environment*) merupakan segala sesuatu yang berada di luar sistem yang mempengaruhi sistem. Lingkungan luar sistem dapat bersifat menguntungkan sistem atau merugikan sistem.

d. Penghubung sistem (*Interface*) merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Penghubung inilah yang menyebabkan beberapa subsistem berintegrasi dan membentuk satu kesatuan.

e. Masukan sistem (*Input*) adalah sesuatu yang dimasukkan ke dalam sistem yang berasal dari lingkungan.

f. Keluaran sistem (*Output*) adalah suatu hasil dari proses pengolahan sistem yang dikeluarkan ke lingkungan.

g. Pengolah sistem (*Process*) merupakan bagian dari sistem yang mengubah masukan (*input*) menjadi keluaran (*output*).

h. Sasaran sistem (*Objectives*) atau tujuan (*Goal*) adalah sesuatu yang menyebabkan mengapa sistem itu dibuat atau ada.

2.2 Konsep Dasar informasi

Informasi sangat penting bagi suatu organisasi. Suatu organisasi yang kurang mendapatkan informasi akan menjadi lemah dan akhirnya berakhir. Robert N. Anthony dan John Dearden (JOG[7]) menyebutkan keadaan dari sistem dalam hubungannya dengan keberakhirannya dengan istilah *entropy*. Informasi yang berguna bagi sistem akan menghindari proses *entropy* yang disebut dengan *negative entropy* atau *negentropy*.

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian pada saat tertentu dan kesatuan nyata.

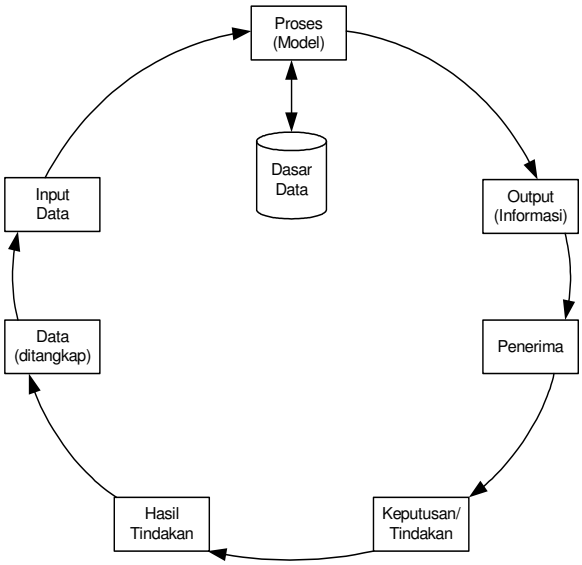
2.2.1 Siklus Informasi

Data merupakan sumber informasi, bentuk yang masih mentah yang belum dapat bercerita banyak, sehingga perlu diolah lebih lanjut. Data diolah melalui suatu model untuk menghasilkan informasi.

Pengolahan data merupakan proses permulaan dari penyusunan perencanaan yang kemudian diidentifikasi. Pengolahan data adalah proses memanipulasi atau transformasi dari data untuk menghasilkan informasi yang berguna.

Menurut Drs. Moekijat MOE[9] dalam bukunya “Pengantar sistem Informasi manajemen, 1998”, menjelaskan pengolahan data adalah kegiatan pikiran dengan bantuan tangan atau suatu peralatan dan mengikuti serangkaian langkah-langkah, perumusan atau pola-pola tertentu untuk mengubah data, sehingga data tersebut baik bentuk, susunan, sifat atau isinya menjadi lebih berguna.

Data yang telah diolah menjadi informasi diberikan kepada yang membutuhkan informasi, kemudian penerima informasi tersebut akan membuat suatu keputusan dan melakukan tindakan, yang berarti melakukan tindakan yang lain yang akan menghasilkan kembali suatu data. Data tersebut akan ditangkap sebagai input, diproses kembali melalui serangkaian langkah-langkah dan seterusnya membentuk suatu siklus. Siklus ini disebut siklus informasi (*information cycle*) dan disebut juga dengan siklus pengolahan data (*data processing cycle*).



Gambar 2.1 Siklus Informasi

[Sumber : Analisis dan Desain Sistem Informasi, Jogiyanto Hartono MBA., Ph.D., Penerbit Andi Yogyakarta]

2.2.2 Kualitas Informasi

Kualitas dari suatu informasi (*quality of information*) tergantung dari tiga hal, yaitu informasi harus akurat (*accurate*), tepat waktu (*timeliness*) dan relevan (*relevance*).

Akurat berarti informasi harus bebas dari kesalahan-kesalahan dan tidak bias atau menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya.

Tepat waktu berarti informasi harus ada pada saat diperlukan, tidak boleh terlambat. Karena informasi merupakan landasan di dalam pengambilan keputusan, bila pengambilan keputusan terlambat, maka dapat berakibat fatal bagi suatu organisasi. Informasi yang sudah usang tidak mempunyai nilai lagi.

Relevan berarti informasi harus mempunyai manfaat bagi pemakainya. Relevansi informasi untuk tiap-tiap orang atau bagian yang satu dengan bagian lainnya di dalam suatu organisasai berbeda-beda.

2.2.3 Nilai Informasi

Nilai dari informasi (*value of information*) ditentukan oleh dua hal, yaitu manfaat dan biaya mendapatkannya. Suatu informasi dikatakan bernilai bila manfaatnya lebih efektip dibanding dengan biaya mendapatkannya. Akan tetapi perlu diperhatikan bahwa informasi yang digunakan di dalam suatu sistem informasi umumnya digunakan untuk beberapa kegunaan, sehingga tidak memungkinkan dan sulit untuk menghubungkan suatu bagian informasi pada suatu masalah tertentu dengan perbandingan biaya untuk memperolehnya, karena

sebagian besar informasi digunakan tidak hanya oleh satu pihak di dalam perusahaan.

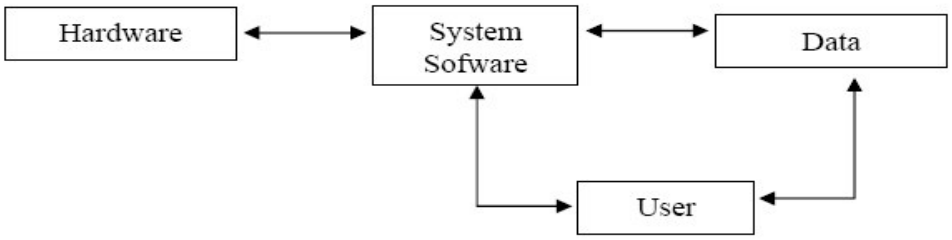
Informasi tidak dapat percis ditaksir dengan satuan uang, tetapi dapat ditaksir dengan efektivitasnya. Pengukuran nilai informasi biasanya dihubungkan dengan analisis *cost effectiveness* atau *cost benefit*.

2.3 Konsep Dasar Sistem Informasi

Telah diketahui bahwa informasi merupakan hal yang sangat penting di dalam mendukung pengambilan keputusan. Informasi dapat diperoleh dari sistem informasi (*Information system*) atau disebut juga dengan *processing system* atau *information processing system* atau *information generating system*. Sistem informasi didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis (JOG[7]) sebagai berikut:

> *"Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan yang diperlukan".*

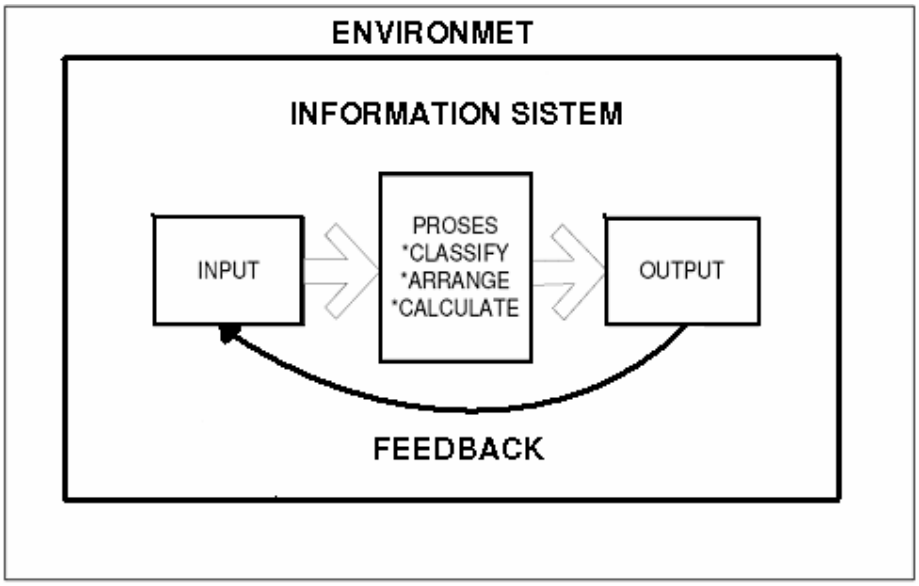
Sistem informasi sendiri memiliki sejumlah komponen tertentu. Seperti yang dikemukakan oleh Robert dan Donald Symanzky (JOG[7]), bahwa sistem informasi terdiri dari beberapa komponen yang berbeda yaitu, manusia, data, *hardware*, dan *software*. Sebagai suatu sistem, setiap komponen tersebut berinteraksi satu dengan lainnya membentuk satu kesatuan untuk mencapai sasarannya.



Gambar 2.2 Komponen Sistem Informasi

[Sumber : http://library.gunadarma.ac.id/files/disk1/2/jbptgunadarma-gdl-s1-2004-basukirach-93-bab2.pdf]

Ada tiga aktifitas pada sistem informasi untuk menghasilkan informasi pada suatu organisasi yang diperlukan untuk membuat suatu keputusan, pengkontrolan, analisis masalah-masalah dan membuat produk baru atau layanan. Aktifitas itu adalah input, proses dan output (BLA[15]).



Gambar 2.3 Aktifitas Sistem Informasi

[Sumber : http://www.bl.ac.id/dosen/hari/kuliahg/buku_pengol.pdf]

INPUT : Menangkap atau pengumpulkan data mentah (*raw data*) di dalam organisasi atau dari luar organisasi (*environment*/lingkungan).

PROSES : Menterjemahkan data input ke dalam bentuk yang lebih berarti.

OUTPUT: Mengirim informasi yang telah diproses kepada manusia dan aktivitasnya

Sistem informasi juga memerlukan *feedback* yang dihasilkan dari output, yang dikembalikan dan digunakan oleh elemen organisasi untuk membantu mereka, mengevaluasi atau menyempurnakan inputan yang akan dimasukkan.

2.4 Pengembangan Sistem

Informasi merupakan sumber daya yang sangat penting bagi organisasi, dengan sistem informasi berbasis komputer dapat mempermudah dalam pengolahan data sehingga informasi yang dihasilkan lebih cepat dan lebih baik daripada informasi yang dihasilkan sistem informasi yang dikelola secara manual.

Pengembangan sistem (*system development*) dapat berarti menyusun sistem yang baru untuk menggantikan sistem yang lama secara keseluruhan atau memperbaiki sistem yang telah ada. Sistem yang lama perlu diganti atau diperbaiki disebabkan karena beberapa hal, yaitu :

1. Adanya permasalahan-permasalahan (*Problems)* yang timbul di dalam sistem lama. Permasalahan-permasalahan yang timbul dapat berupa :
    a. Ketidakberesan

        Ketidakberesan dalam sistem yang lama menyebabkan sistem yang lama tidak dapat beroperasi sesuai dengan yang diharapkan. Ketidakberesan ini dapat berupa :

1. Kecurangan-kecurangan yang disengaja yang menyebabkan tidak amannya harta kekayaan perusahaan dan kebenaran dari data menjadi tidak terjamin.
2. Kesalahan-kesalahan yang tidak disengaja yang juga dapat menyebabkan kebenaran dari data tidak terjamin.
3. Tidak efisiennya operasi.
4. Tidak ditaatinya kebijaksanaan manajemen yang telah ditetapkan.

b. Pertumbuhan Organisasi

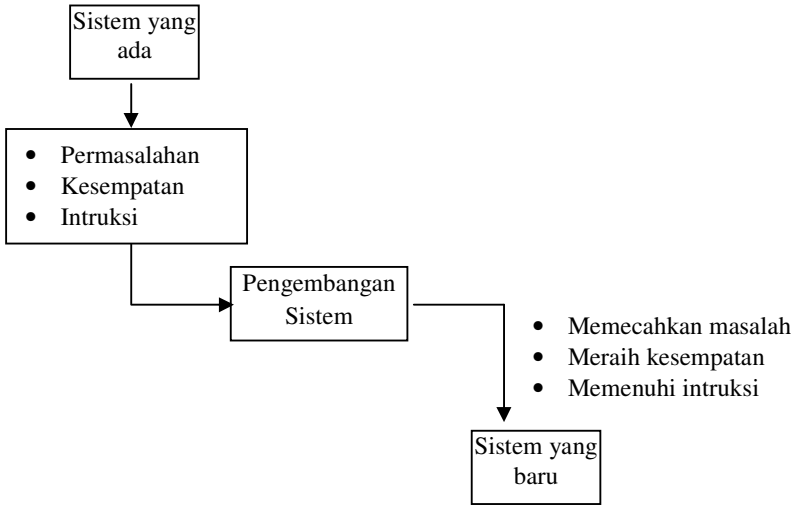
Pertumbuhan organisasi menyebabkan harus disusunnya sistem yang baru. Pertumbuhan organisasi diantaranya adalah kebutuhan informasi yang semakin luas, *volume* pengolahan data semakin meningkat, perubahan prinsip akuntansi yang baru. Karena adanya perubahan ini menyebabkan sistem yang lama tidak efektif lagi, sehingga sistem yang lama sudah tidak dapat memenuhi lagi kebutuhan informasi yang dibutuhkan manajemen.

2. Untuk meraih kesempatan-kesempatan (*Opportunities*)

Teknologi informasi telah berkembang dengan cepatnya. Organisasi mulai merasakan bahwa teknologi informasi ini perlu digunakan untuk meningkatkan penyediaan informasi sehingga dapat mendukung dalam pengambilan keputusan. Kecepatan informasi atau efisiensi waktu sangat menentukan berhasil atau tidaknya strategi dan rencana yang telah disusun untuk meraih kesempatan-kesempatan yang ada.

3. Adanya intruksi-intruksi (*Directives*)

Pengembangan sistem yang baru dapat juga terjadi karena adanya intruksi-intruksi dari atas pimpinan ataupun dari luar organisasi, seperti misalnya peraturan pemerintah.



Gambar 2.4 Pengembangan Sistem

[Sumber : Analisis dan Desain Sistem Informasi, Jogiyanto Hartono MBA., Ph.D., Penerbit Andi Yogyakarta]

2.5 Siklus Hidup Pengembangan Sistem

Proses pengembangan sistem melewati beberapa tahapan-tahapan, mulai dari sistem itu direncanakan sampai dengan sistem itu diterapkan, dioperasikan dan dipelihara. Bila operasi sistem yang sudah dikembangkan masih timbul kembali permasalahan-permasalahan kritis serta tidak dapat diatasi dalam tahap pemeliharaan sistem, maka perlu dikembangkan kembali suatu sistem untuk mengatasinya dan proses ini kembali ke tahap yang pertama. Siklus ini disebut dengan siklus hidup sistem (*system life cycle*).

Ide dari *system life cycle* adalah sederhana dan masuk akal. Di *system life cycle*, tiap-tiap bagian dari perkembangan sistem dibagi menjadi beberapa tahapan kerja. Tiap-tiap tahapan ini mempunyai karakteristik tersendiri.

Menurut KUL[16], SDLC berfungsi untuk menggambarkan tahapan-tahapan utama dalam mengembangkan suatu sistem informasi. Kegiatan-kegiatan utama dalam SDLC dapat digolongkan menjadi tiga bagian, yaitu penelitian, pengembangan sistem, dan manajemen sistem. Ketiga bagian tersebut secara lebih rinci dapat dijelaskan seperti berikut :

1. Penelitian

   Kegiatan ini merupakan tahapan paling awal dalam siklus hidup sistem, tahapan perencanaan sistem termsuk dalam kegiatan ini. Sebelum suatu sistem dikembangkan, maka perlu direncanakan terlebih dahulu secara cermat. Perencanaan sistem (*system planning*) ini menyangkut estimasi dari kebutuhan-kebutuhan fisik, tenaga kerja, dan dana yang dibutuhkan untuk mendukung pengembangan sistem serta untuk mendukung operasinya setelah diterapkan.

2. Pengembangan Sistem

   Secara garis besar kegiatan ini dapat dibagi menjadi tiga bagian, setiap kegiatan dalam pengembangan sistem ini dapat dijelaskan melalui tujuan (*purpose*) dan hasil kegiatannya (*deliverable*).

   a. Analisis

      Dalam tahapan ini, sistem yang ada dianalisis untuk :

1. Membuat keputusan apabila sistem yang ada mempunyai masalah atau tidak berfungsi secara baik dan hasil analisisnya digunakan sebagai dasar untuk memperbaiki sistem.
2. Mengetahui ruang lingkup pekerjaan yang akan ditangani.
3. Memahami sistem yang ada/berjalan.
4. Mengidentifikasi masalah dan mencari solusinya.

Kegiatan yang dilakukan dalam tahap analisis ini adalah :

1. *Problem detection,* mendeteksi masalah-masalah pada sistem apabila sistem makin berkurang manfaatnya.
2. *Initial investigation,* memeriksa sistem yang sedang berjalan dengan memfokuskan pada bagian-bagian yang mengalami masalah.
3. *Requirement analysis,* mengetahui kebutuhan sistem sesuai dengan harapan *user*, penerapan sistem yang baru akan menimbulkan perbedaan antara sistem yang lama dengan sistem yang ideal (berbasis komputer).
4. *Generation of system alternatives,* mencari perbedaan antara alternatif sistem yang baru dalam upaya penggantian sistem yang lama.
5. *Selection of proper system,* membandingkan alternatif-alternatif sistem dengan mengunakan suatu metode, memilih alternatif sistem yang paling baik, dan merancangnya untuk *user*.

b. Desain

Tahap ini memiliki tujuan untuk mendesain sistem yang baru yang dapat mengatasi permasalahan-permasalahan yang muncul dipilih dari alternatif pemilihan sistem yang terbaik. Kegiatan yang dilakukan pada tahapan ini adalah :

1. *Input design*, merancang bentuk-bentuk inputan baik yang berupa dokumen atau dilayar/antarmuka pengguna (*user interface*).
2. *Output design*, merancang tampilan-tampilan untuk output dari sistem, termasuk dokumen serta antarmuka pengguna (*user interface*).
3. *File design*, merancang bentuk *file/basisdata* yang dibutuhkan dalam suatu sistem informasi.

c. Implementasi

Tahapan implementasi memiliki beberapa tujuan, yaitu :

1. Melakukan spesifikasi dari konsep yang ada untuk penerapan sistem informasi yang dibangun.
2. Mengimplementasikan sistem yang baru.
3. Menjamin bahwa sistem yang baru dapat berjalan secara optimal.

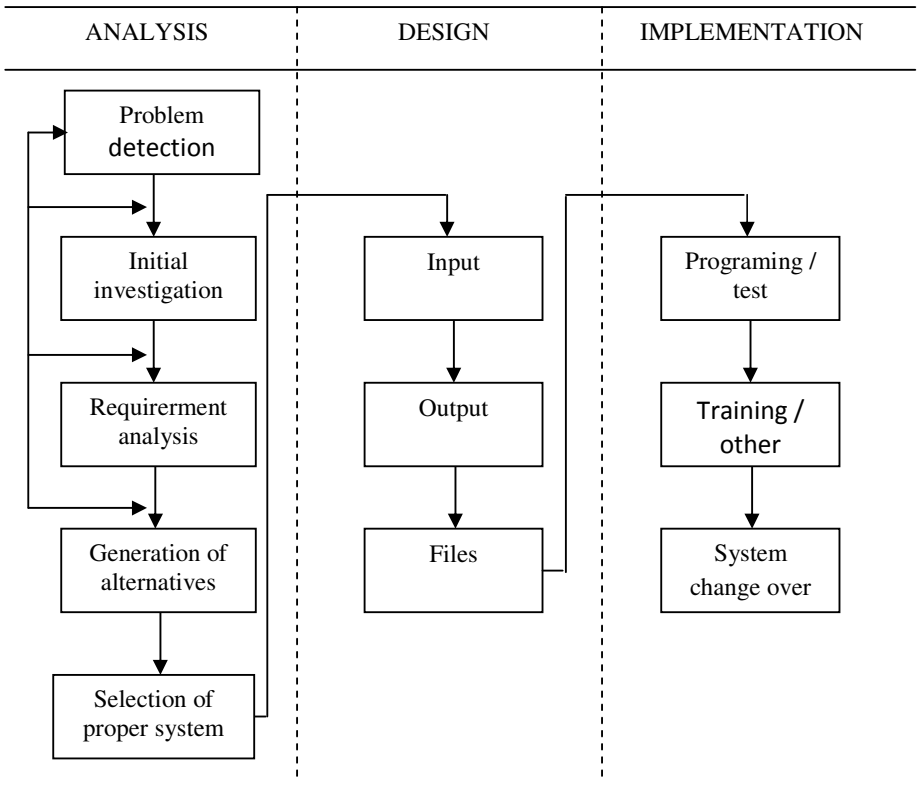
Kegiatan-kegiatan yang dilakukan dalam tahap implementasi ini adalah :

1. *Programming & Testing*, membuat program berdasarkan acuan dari hasil analisis, dan mengetes keseluruhan program

untuk memastikan semua fungsi/modul-modul program dapat berfungsi secara benar.

2. *Training*, melakukan pelatihan-pelatihan terhadap *user* atau operator-operator yang akan menggunakan sistem.
3. *Change over*, mengganti sistem yang lama dengan sistem yang baru.

Apabila ketiga kegiatan utama dalam pengembanagan sistem tersebut dijabarkan ke dalam langkah-langkah yang lebih rinci dapat digambarkan sebagai berikut :



Gambar 2.5 Tahapan Pengembangan Sistem Informasi

[Sumber : http://kuliah.dinus.ac.id/ika/asi4.html]

3. Manajemen Sistem

   Tahap ini adalah tahap akhir dalam daur hidup sistem informasi. Pada tahapan ini, sistem dioperasikan dan dikelola. Namun apabila terjadi permasalahan-permasalahan kritis yang sekiranya tidak dapat diatasi pada tahap pemeliharaan sistem, maka perlu dibangun kembali suatu sistem informasi yang baru untuk mengatasinya dan proses ini kembali lagi ke tahapan yang paling awal.

2.6 Metoda Pendekatan Berorientasi Objek

Dalam suatu proses pengembangan sistem informasi, analisis dan perancangan merupakan terminologi yang sudah sangat tua. Sebenarnya pada saat masalah ditelusuri dan spesifikasi dinegoisasikan, dapat dikatakan kita berada pada tahap perancangan. Merancang adalah menemukan suatu cara untuk menyelesaikan masalah, salah satu *tool/*model untuk merancang *software* yang berorientasi objek (*object oriented*) adalah UML (FOW[4]).

2.6.1 Konsep Objek

Objek dalam "*software analysis & design"* adalah sesuatu yang berupa konsep (*concept*), benda (*thing*), dan sesuatu yang membedakannya dengan lingkungannya. Objek biasanya adalah kata benda, namum dalam konteks OOP objek bukan saja yang bisa dilihat atau diraba, tapi objek dapat pula merupakan sesuatu yang abstrak yang hidup di dalam sistem seperti *file*, tabel, *database, event, system messages* (NUG[10]).

Objek dapat dikenali dari keadaannya dan juga operasinya, sebagai contoh sebuah mobil dikenali dari warnanya, bentuknya, sedangkan manusia dari suaranya, namanya, serta ciri-ciri yang akan membedakan objek tersebut dari objek lainnya.

Alasan mengapa pada saat ini memilih pendekatan berorientasi objek (*object oriented*) dalam pengembangan sistem informasi, pertama adalah *scalability* dimana objek lebih mudah dipakai untuk menggambarkan sistem yang besar dan komplek. Kedua, *dynamic modeling*, dimana objek dapat dipakai untuk pemodelan sistem dinamis dan *real time* (NUG[10]).

2.6.2 Pengembangan Berorinetasi Objek

Object oriented development adalah suatu cara untuk pengembangan perangkat lunak dan sistem informasi berdasarkan abstraksi objek-objek yang ada di dunia nyata. (Brooks 1987, dikutip dari buku Object Oriented Modeling And Design, tulisan James Rumbaugh, dkk) (NUG[10]) mengemukakan bagian tersulit dari pengembangan perangkat lunak dan/atau sistem informasi adalah tahapan analisis dimana kita harus menganalisis masalah yang sangat rumit, yang kita jumpai di dunia nyata serta melakukan abstraksi terhadap masalah itu, kemudian melakukan perancangan agar kelak dapat diimplementasikan dengan cepat serta akurat pada komputer (baik secara perangkat keras maupun perangkat lunak) menjadi suatu sistem/perangkat lunak yang sesuai kebutuhan serta harapan pengguna.

2.6.3 Teknik Dasar OOA/D *(Object-Oriented Analysis/Design)*

Dalam dunia pemodelan, metodologi berorientasi objek walaupun terikat kaidah-kaidah standar, namun teknik pemilihan objek tidak terlepas pada subyektifitas *software analyst* dan *designer*. Beberapa objek akan diabaikan dan beberapa objek menjadi perhatian untuk diimplementasikan di dalam sistem, hal ini sah-sah saja karena kenyataan bahwa suatu permasalahan sudah tentu memiliki lebih dari satu solusi.

Ada 3 (tiga) teknik/konsep dasar dalam OOA/D, yaitu pemodulan (*encapsulation*), penurunan (*inheritance)* dan *polymorphism.*

1. *Encapsulation*

   Encapsulation (pengkapsulan) merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses.

2. *Inheritance*

   Inheritance (*pewarisan*) adalah teknik yang menyatakan bahwa anak (turunan) dari objek akan mewarisi data/atribut dan metoda dari induknya langsung.

3. *Polymorfhism*

   Polimorfisme adalah konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku yang berbeda.

2.6.4 Pengenalan UML

UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasi, membangun, dan mendokumentasikan dari sebuah pengembangan sistem berorientasi objek

(*Object-Oriented*). UML sendiri juga memberikan standar penulisan sebuah *system blue print*, yang meliputi konsep proses bisnis, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem *software*.

UML merupakan bahasa pemodelan yang memiliki pembendaharaan kata dan cara untuk mempresentasikan secara fokus pada konseptual dan fisik dari suatu sistem. Contoh untuk sistem *software* yang *intensive* membutuhkan bahasa yang menunjukan pandangan yang berbeda dari arsitektur sistem, ini sama seperti menyusun/mengembangkan *software develpoment life cycle*. Dengan UML akan memberitahukan kita bagaimana untuk membuat dan membaca bentuk model yang baik, tetapi UML tidak dapat memberitahukan model apa yang akan dibangun dan kapan akan membangun model tersebut. Ini merupakan aturan dalam *software development process* (NUG[10]).

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*.

Dalam kerangka spesifikasi, UML menyediakan model-model yang tidak mendua-arti (ambigu), serta lengkap. Secara khusus, UML menspeifikasi langkah-langkah penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (*software intensive system*).

2.6.5 Bagian-Bagian dari UML

Bagian–bagian utama dari UML adalah *view, diagarams, model element, dan general mechanism.*

1. *View*

   View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. *View* bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram. Beberapa jenis *view* dalam UML antara lain *use case view, logical view, component view, concurrences view,* dan *deployment view*.

   a. *Use case view*

      Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai dengan yang diinginkan *external actors. Actor* yang berinterakasi dengan sistem dapat berupa *user* atau sistem lainnya. *View* ini digambarkan dalam *use case diagram* dan kadang–kadang dengan *activity diagrams. View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang *(developer*), dan penguji sistem (*tester*).

   b. *Logical view*

      Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class, object*, dan r*elationship*) dan kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu. *View* ini digambarkan dalam *class diagram* untuk struktur statis dan dalam *state sequence*, *collaboration,* dan *activity diagrams*

untuk model dinamisnya. *View* ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

c. *Component view*

Mendiskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi *administrative* lainnya. *View* ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

d. *Concurrency view*

Membagi sistem ke dalam proses dan *processor*. *View* ini digambarkan dalam diagram dinamis (*state, sequence, collaboration*, dan *activity diagram*) dan diagram implementasi (*component* dan *deployment diagrams*) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

e. *Deployment view*

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat lain serta bagaimana hubungan dengan yang lainnya. *View* ini digambarkan dalam *deplovment diagram* dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2. *Diagrams*

Setiap sistem yang kompleks seharusnya bisa dipandang dari sudut yang berbeda-beda sehingga kita bisa mendapatkan pemahaman secara menyeluruh. Untuk upaya tersebut UML menyediakan setidaknya

sembilan jenis diagram. Diagram-diagram ini tidak mutlak harus digunakan semuanya, hanya dibuat sesuai dengan kebutuhan. Pada pemodelan dengan UML juga dimungkinkan menggunakan diagram-diagram lain sejauh itu memang diperlukan untuk mendapatkan pemahaman mendalam tentang suatu sistem atau suatu perangkat lunak. Diagram-diagram yang disediakan UML antara lain :

a. *Class diagram*

   Diagram ini memperlihatkan himpunan kelas, antarmuka, kolaborasi serta relasi. Diagram ini umumnya dijumpai pada pemodelan berorientasi objek. Meskipun bersifat statis sering pula diagram kelas memuat kelas-kelas aktif.

b. *Object diagram*

   Diagram ini memperlihatkan objek-objek serta relasi antar objek. Diagram objek memperlihatkan istansiasi statis dari segala sesuatu yang dijumpai pada diagram kelas.

c. *Use Case diagram*

   Diagram ini memperlihatkan himpunan *Use Case* dan aktor-aktor. Diagram ini sangat penting terutama untuk mengorganisasi dan memodelkan perilaku-perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.

d. *Sequence diagram*

   Diagram ini berfungsi sebagai interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu.

e. *Collaboration diagram*

Diagram interaksi yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan (*message*).

f. *State diagram*

Menggambarkan semua *state* pada sistem, memuat *state*, transisi, *event* serta aktifitas. Diagram ini penting untuk memperlihatkan sifat dinamis dari antarmuka, kelas serta kolaborasi terutama pada pemodelan sistem-sistem yang reaktif.

g. *Aktivity diagram*

Diagram ini adalah tipe khusus dari diagram *state* yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini penting bagi pemodelan fungsi-fungsi dalam suatu sistem dan memberikan tekanan pada aliran kendali antarobjek.

h. *Component Diagram*

Diagram ini memperlihatkan organisasi serta ketergantungannya pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas, antarmuka serta kolaborasi.

i. *Deployment Diagram*

Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Diagram ini memuat simpul-simpul (*node*) beserta komponen-komponen yang ada di dalamnya. *Deployment diagram* berhubungan erat dengan *component diagram* dimana *deployment diagram* memuat

satu atau lebih komponen. Diagram ini berguna saat aplikasi akan diterapkan pada banyak komputer (*distributed computing*).

2.7 Jaringan Komputer

Jaringan komputer adalah sekumpulan komputer yang berhubungan antara yang satu dengan yang lainnya menggunakan *protocol* komunikasi sehingga dapat saling bertukar data dan informasi serta dapat menggunakan perangkat keras secara bersama. Selain itu jaringan komputer dapat diartikan sebagai sekumpulan terminal komunikasi yang berada di berbagai lokasi berbeda yang terdiri dari dua komputer atau lebih yang saling berhubungan (IRA[6]).

2.7.1 Local Area Network

Local Area Network (LAN) adalah jaringan yang dibatasi oleh area yang relatif kecil, umumnya dibatasi oleh area lingkungan seperti perkantoran di sebuah gedung atau sebuah sekolah dan biasanya jaraknya tidak lebih dari 1 km.

Beberapa model konfigurasi LAN biasanya berupa satu komputer yang dijadikan *file server* yang digunakan untuk menyimpan perangkat lunak (*Software* yang mengatur aktifitas jaringan) serta beberapa komputer yang terhubung ke *file server* yang disebut dengan *workstation*. LAN terbagi menjadi dua model hubungan, yaitu :

1. *Peer-To-Peer*

   Model hubungan *peer-to-peer* memungkinkan *user* membagi sumber daya baik itu berupa *file*, layanan printer atau lainnya. Namun model ini tidak mempunyai *file server* atau sumber daya yang terpusat. Di dalam model

peer-to- peer ini, seluruh komputer adalah sama, berarti mempunyai hak yang sama untuk memakai sumber daya yang tersedia di dalam jaringan. Model ini didesain untuk jaringan bersekala kecil dan menengah.

2. *Client-Server*

Model hubungan ini memungkinkan jaringan untuk mensentralisasi fungsi atau aplikasi kepada satu atau lebih *dedicated file server.* Sebuah *file server* menjadi jantung bagi keseluruhan sistem, memungkinkan untuk mengakses sumber daya, dan menyediakan keamanan. Model hubungan ini menyediakan mekanisme untuk mengintegrasikan seluruh komponen yang ada pada jaringan dan memungkinkan banyak pengguna secara bersama-sama memakai sumberdaya pada *file server.*

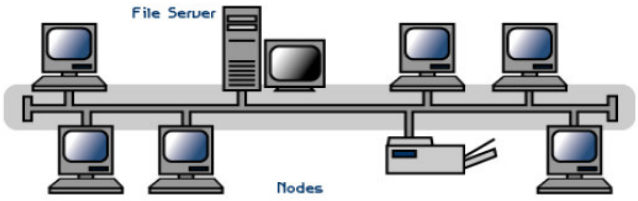
2.7.2 Topologi Jaringan

Topologi secara fisik dari suatu jaringan lokal adalah merujuk kepada konfigurasi kabel, komputer serta perangkat lainnya.

Tipe-tipe utama topologi jaringan :

1. *Linear bus*

   Topologi *linear bus* terdiri dari satu jalur kabel utama, dimana pada masing-masing ujungnya diberikan sebuah *terminator*. Semua *nodes* pada jaringan (*file server*, *workstation*, dan perangkat lainnya) terkoneksi pada kabel utama (*backbone*). Jaringan *Ethernet* dan *local talk* mengunakan topologi ini.

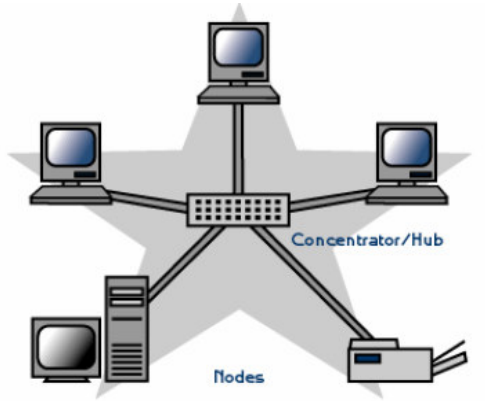


Gambar 2.6 Topologi Linear Bus

[Sumber : http://fcit.coedu.usf.edu/network/chap5/chap5.htm]

2. *Star*

Topologi model ini dirancang dimana setiap *nodes* terkoneksi ke jaringan melewati sebuah *concentrator*. Data yang dikirim ke jaringan lokal akan melewati *concentrator* sebelum melanjutkan ke tempat tujuannya. *Concentrator* akan mengatur dan mengendalikan keseluruhan fungsi jaringan, dan juga bertindak sebagai *repeater* (penguat aliran data). Konfigurasi pada model ini menggunakan kabel *twisted pair*, dan dapat pula digunakan kabel *coaxial* atau kabel *fiber optic*.

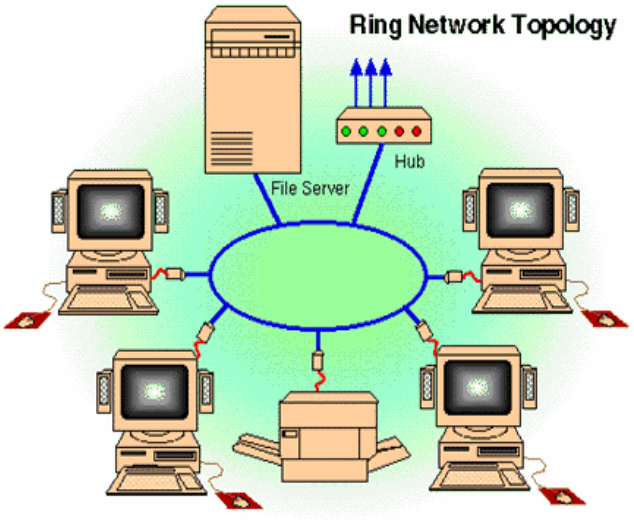


Gambar 2.7 Topologi Star (Bintang)

[Sumber : http://fcit.coedu.usf.edu/network/chap5/chap5.htm]

3. *Ring*

Pada dasarnya topologi ini menggunakan teknik konfigurasi yang sama dengan topologi *star*, tetapi pada topologi ini terlihat bahwa jalur media transmisi menyerupai suatu lingkaran tertutup seperti cincin (*ring*).

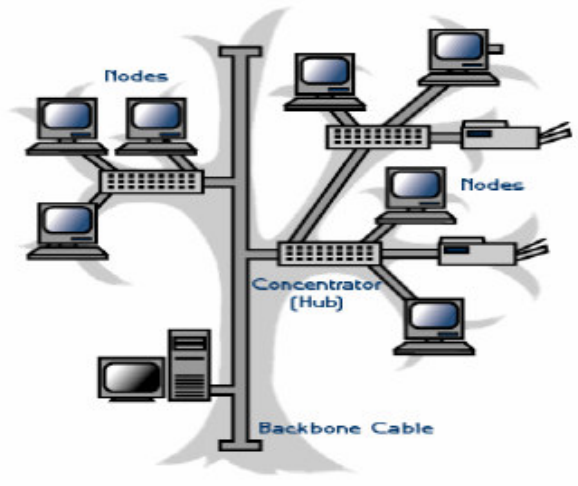


Gambar 2.8 Topologi Ring

[Sumber : http://myschoolnet.ppk.kpm.my/tutor_it/rangkai/topologi/cincin.htm]

4. *Tree*

Topologi model ini merupakan perpaduan antara topologi *linear bus*, dan *star* yang terdiri dari kelompok-kelompok *workstation* dengan konfigurasi *star* yang terkoneksi ke kabel utama yang menggunakan topologi *linear bus*. Topologi ini memungkinkan untuk pengembangan jaringan yang telah ada.



Gambar 2.9 Topologi Tree

[Sumber : http://fcit.coedu.usf.edu/network/chap5/chap5.htm]

2.8 Perangkat Lunak Pendukung

Dalam pengembangan suatu sistem informasi, diperlukan perangkat lunak lain yang bisa digunakan untuk membuat suatu aplikasi sesuai dengan keinginan *user.*

2.8.1 Rational Rose Enterprise

Rational rose adalah *software* yang memiliki perangkat-perangkat pemodelan secara *visual*, dasar-dasar pemodelan dengan *rational rose* yaitu :

Visual modeling adalah proses menggambarkan cetak biru suatu sistem secara grafis terdiri dari komponen-komponen, grafis-grafis dan koneksi-koneksi yang ada di dalam sistem tersebut agar mudah dipahami dan dikomunikasikan.

Visual modeling dapat membantu kita untuk menampilkan elemen-elemen yang penting secara detail dari suatu masalah yang kompleks dan menyaring untuk kemudian membuang elemen-elemen tidak penting.

2.8.2 Visual Basic .NET

Sekarang ini Microsoft telah dapat merealisasikan visi "Sistem operasi *Windows* dalam setiap PC dan PC dalam setiap *desktop*". Salah satu *tool* untuk mengembangkan aplikasi .NET adalah Microsoft Visual Basic .NET (disingkat dengan VB net). VB .NET bersama dengan Visual C++ .NET, Visual C# .NET, Visual J++ .NET dan Visual J# .NET merupakan bagian dari Microsoft Visual Studio .NET.

VB .NET adalah bahasa pemrograman untuk membuat aplikasi berbasis *Windows*, aplikasi *form* Web ASP .NET, layanan Web XML dan aplikasi *mobile* seperti komputer Palm dan *Pocket PC*. VB .NET dibangun di atas fondasi *.NET Framework* (lingkungan kerja .NET).

Setiap generasi baru dari perangkat lunak bahasa pemrograman datang karena adanya keterbatasan dari generasi sebelumnya. Teknologi *device*, *hardware*, *network* dan internet baru yang muncul menyebabkan bahasa pemrograman yang ada tidak lagi menjadi alat yang ideal untuk mengembangkan perangkat lunak yang dapat bekerja dengan teknologi baru tersebut (WAH[12]).

Sekarang untuk pertama kalinya, *platform* pengembang perangkat lunak yang lengkap, Microsoft .NET telah didesain dari dasar dengan internet sebagai fokus utamanya (walaupun tidak secara eksklusif hanya untuk pengembang internet saja). Banyak inovasi baru yang berada dalam *platform* ini akan mengatasi keterbatasan dari *tool-tool* dan teknologi lama.

Visual Basic .NET adalah pengembangan dari Visual basic sebelumnya. Kelebihan VB .NET 2005 terletak pada tampilannya yang lebih canggih dibandingkan dengan edisi Visual Basic sebelumnya.

Selain memiliki kelebihan, VB .NET 2005 memiliki kekurangan. Kekurangan VB .NET 2005 yang terlihat jelas adalah beratnya aplikasi ini apabila dijalankan pada komputer yang memiliki spesifikasi sederhana.

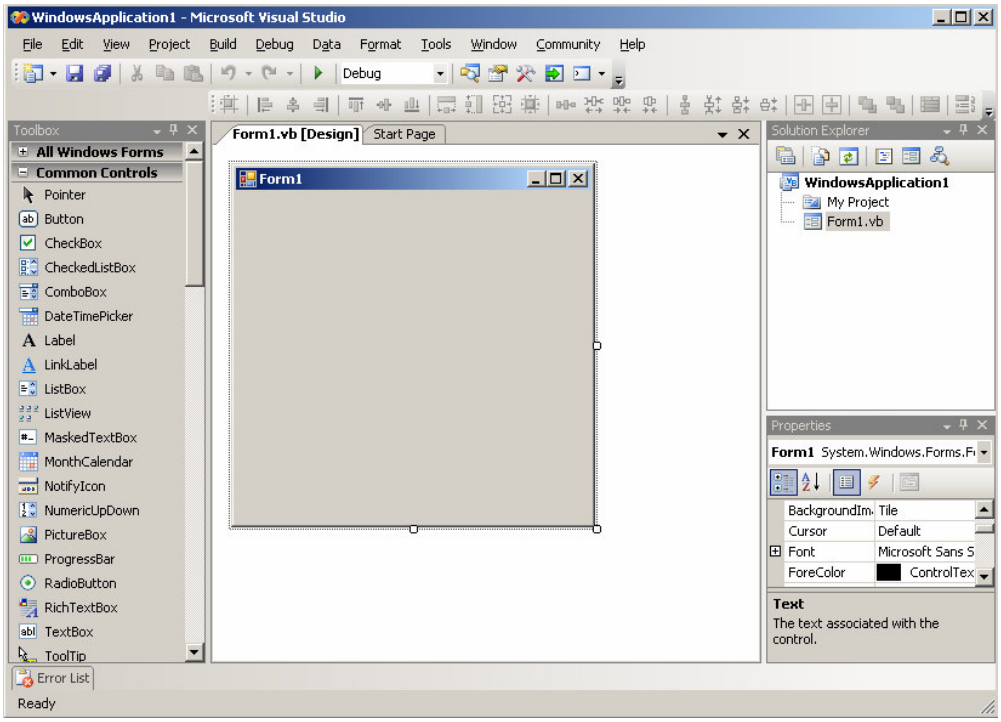
Syarat minimal yang harus dipenuhi oleh komputer agar VB .NET 2005 dapat diinstal adalah :

Tabel 2.1 Spesifikasi Minimal Visual Basic.NET 2005

[Sumber : Microsoft Visual Studio 2005 Documentation]

Processor	**Minimum :** 600 Mhz Pentium Processor **Recommended :** 1 Ghz Pentium Processor
Operating System	Microsoft Windows® 2003 Server ™ SP 1, Windows XP SP2, Windows 2000 SP4
RAM	**Minimum :** 128 MB **Recommended :** 256 MB
Had Disk	**Without MSDN :** 2 GB of available space on installation drive. 1 GB of available space required on system drive **With MSDN :** 3.8 GB of available space on installation drive with a full MSDN install or 2.8 GB of available space on installation drive with a default MSDN install. 1 GB of available space on system drive
CD or DVD Drive	Not Required
Display	**Minimum :** 800 X 600 256 colors **Recommended :** 1024 X 768 High Color-16 Bit

Visual Basic .NET mempunyai lingkungan pengembangan yang terintegrasi atau sering disebut dengan IDE (*Integrated Developmnet Environment*) dengan lingkungan kerja yang luas. Area kerja adalah jendela yang berguna untuk melakukan kegiatan pengisian kode (*coding*) ketika berupa tampilan kode (*code view*) dan untuk mengatur tampilan desain ketika berupa tampilan desainer (*desaign view*).



Gambar 2.10 IDE Visual Basic.NET 2005

2.8.3 SQL Server 2005

SQL Server adalah salah satu produk *Relational Database Management System* (RDBMS) populer saat ini. Fungsi utamanya adalah sebagai *database server* yang mengatur semua proses penyimpanan data dan transaksi suatu aplikasi.

SQL Server 2005 memiliki fitur unik dengan mengintegrasikan *.NET framework* menjadi bagian dari *database engine*. Hal tersebut memungkinkan manipulasi data dapat dilakukan di dalam SQL Server dengan memanfaatkan fitur-fitur yang dimiliki SQL Server, seperti kode yang terkelola, pendekatan keamanan .NET, hingga integrasi pengembangan dengan Visual Studio 2005.

Assembly pada SQL Server 2005 memiliki kesamaan dengan *Assembly* yang terdapat pada .NET *runtime*. *Assembly* adalah unit distribusi aplikasi berbasis *.NET framework* yang dikembangkan dengan bahasa pemrograman yang mendukung *.NET framework*. Hal yang menarik bahwa hasil kompilasi *assembly* yang dipasang pada SQL Server 2005 akan memiliki sifat dan prilaku yang sama dengan CLR pada umumnya seperti kode yang terkelola (*Managed Code*). Sama halnya dengan aplikasi berbasis .NET, SQL Server *assembly* juga dikompilasi oleh *language compiler* menjadi *assembly* yang didalamnya terdapat metadata dan MSIL. SQL Server *Assembly* dikompilasi dalam bentuk *class library* (*dll*). Pada saat SQL Server *Assembly* dieksekusi maka SQL Server akan memanggil pustaka *Microsoft common object runtime (mscoree.dll)* yang akan menjalankan *assembly* yang bersangkutan di dalam ruang lingkup SQL Server setelah *assembly object* yang bersangkutan telah dieksekusi maka *assembly* akan di-*cache* dalam *memory* sementara waktu, dan hanya akan dihapus bila SQL Server mengalami tekanan atau kebutuhan *memory* yang cukup besar.

1. *Code Access Security*. *Assembly pada* SQL server memiliki tiga tingkat pengamanan dari sisi kode dan sumber daya, yakni :
    a. *SAFE, assembly* yang menggunakan ijin ini hanya berjalan diruang lingkup aplikasi .NET *runtime* saja.

b. *EXTERNAL_ACCESS, assembly* yang menggunakan ijin ini memiliki kemampuan untuk mengakses sumber daya eksternal, seperti I/O, internet atau *e-mail*.

c. *UNSAFE, assembly* yang memiliki izin ini melakukan berbagai aksi, termasuk didalamnya mengeksekusi *assembly* lain yang tidak terverifikasi, atau memanggil kode tak terkelola (*unmanaged code*).

2. Manajemen Memori, SQL Server *assembly* juga menggunakan konsep *automatic garbage collection*. Hal yang menarik pada SQL server *assembly* adalah besarnya memori yang diutilasikan oleh CLR akan disesuaikan dengan alokasi yang digunakan SQL Server, sehingga SQL server dapat mengontrol proses CLR yang terjadi dalam ruang lingkup memori.

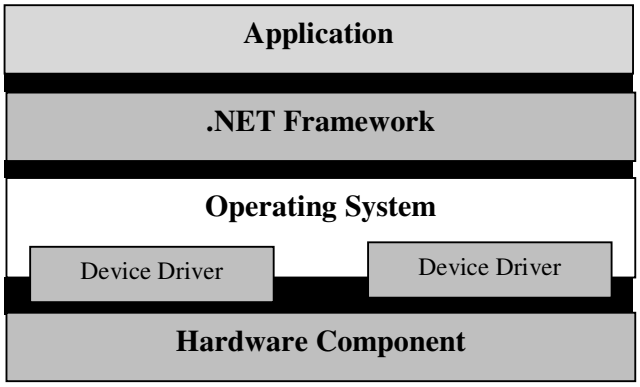
3. *Application Domains,* CLR mengijinkan setiap *assembly* berjalan dalam sebuah *Application Domains (AppDomains)*. *AppDomain* memungkinkan suatu *assembly* yang dieksekusi terisolasi dengan *assembly* yang lain. CLR juga memiliki kemampuan menampung banyak *AppDomains*, sehingga CLR mampu menjalankan banyak aplikasi dalam sebuah proses tunggal. *Assembly* SQL Server akan berjalan dalam *database* yang *notabene* dimiliki oleh satu atau lebih pengguna. Sebuah *assembly* yang dimiliki oleh pengguna yang sama dalam sebuah *database* akan berjalan dalam sebuah *AppDomains*. Begitu pula sebaliknya *assembly* yang dimiliki oleh penggguna yang berbeda walau menggunakan *database* yang sama bekerja dalam *AppDomain* yang berbeda. Hal ini memberi suatu fleksibilitas proses, sehingga bila dalam suatu waktu SQL Server *process* harus me-

reset Appdomain, maka pengguna lain yang mungkin menggunakan *database* yang sama tidak terpengaruh.

2.8.4 Mengenal .NET Framework

Dalam pengembangan Visual Basic .NET ada banyak perubahan radikal, salah satunya adalah pengembangan dasar bagi semua alat-alat pengembangan .NET. dasar ini disbut *.NET Framework* yang menyediakan dua hal yaitu *Common Language Runtime* (CLR) dan *class library* (YUS[13]).

CLR mirip dengan sistem operasi, menyediakan lapisan antara program dengan kompleksitas dari sistem, menampilkan pelayanan dari aplikasi dan penyederhanaan akses pada fungsi lapisan paling bawah.



Gambar 2.11 Lapisan .NET Framework

[Sumber : Belajar Sendidri dalam 21 Hari Visual Basic .NET, Duncan Mackenzie & Kent Sharkey, Penerbit Andi Yogyakarta]

Class library menyediakan satu set besar fungsi *wrapping* dan abstraksi seperti teknologi *Internet Protocol*, akses *file system*, manipulasi XML dan masih banyak lagi.

Dengan demikian *.NET Framework* didesain untuk memenuhi tujuan berikut :

1. Menyediakan lingkungan pemrograman berorientasi objek yang konsisten meskipun kode objek disimpan dan dijalankan secara lokal, tetapi disebarkan melalui internet atau dijalankan secara *remote*.
2. Menyediakan lingkungan yang menjalankan kode dengan meminimalkan konflik saat *software* disebarkan (*deployment*).
3. Menyediakan lingkungan yang menjamin keamanan saat kode dijalankan, termasuk kode yang dijalankan oleh pihak ketiga yang tidak diketahui atau kurang dipercaya.
4. Menyediakan lingkungan untuk menjalankan kode yang dapat menghilangkan masalah performa dari lingkungan *scripted* dan *interpreted.*
5. Membangun komunikasi standar industri guna memastikan kode berbasis *.NET Framework* dapat disatukan dengan kode-kode lain.

2.9 Pengujian Sistem

Langkah terakhir sebelum sistem/perangkat lunak diserahkan kepada user adalah melakukan pengujian(testing) terhadap sistem/perangkat lunak tersebut.

Menurut NUG[10] pengujian pada dasarnya adalah menemukan dan menghilangkan kesalahan pada sistem/perangkat lunak yang akan diterapkan. Kesalahan tersebut dapat diakibatkan oleh beberapa hal utama, antara lain kesalahan saat spesifikasi sistem/perangkat lunak, kesalahan saat melakukan

analisis permasalahan, kesalahan saat perancangan serta kesalahan saat implementasi.

Konsep kualitas sangat penting demi kepuasan pengguna dan pengembang. Untuk mencapai kualitas yang diharapkan dari sistem/perangkat lunak yang dikembangkan pada umumnya ada beberapa strategi pengujian yang dapat dilakukan.

1. *Black-Box Testing*

   Pada pengujian ini kita tidak perlu tahu apa yang sesungguhnya terjadi dalam sistem/perangkat lunak yang kita kembangkan. Kita menguji masukan dan keluarannya, artinya dengan berbagai masukan yang kita berikan, sistem/perangkat lunak memberikan keluaran yang kita harapkan.

2. *White-Box Testing*

   Pengujian jenis ini mengasumsikan spesifikasi logika sangat penting dan perlu dilakukan pengujian untuk menjamin sistem/perangkat lunak berfungsi dengan baik. Cara ynag dilakukan dalam pengujian ini yaitu :

   a. memeriksa semua fungsi dalam setiap objek dengan mengeksekusinya satu-persatu.
   b. memeriksa setiap alternatif yang mungkin dalam setiap objek.

3. *Top-DownTesting*

   Strategi pengujian ini berasumsi bahwa logika utama atau interaksi antarobjek perlu diuji lebih lanjut. Strategi ini sering kali dapat mendeteksi cacat, kesalahan atau kekurangan yang serius pada sistem/perangkat lunak.

4. *Bottom-Up Testing*

   Strategi pengujian ini dimulai dengan menguji rincian sistem kemudian beanjak ke tingkat yang lebih tinggi, misalkan dimulai dari menguji metoda-metoda dalam kelas, menguji setiap kelas serta interaksi antarkelas dan seterusnya hingga ke tingkat yang paling tinggi.

2.10 Pengertian Administrasi Kesiswaan

Kata administrasi berasal bahasa yunani yang terdiri dari kata "*AD*" dan "*MINISTRATE*" yang berarti mengabdi, melayani dan memenuhi. Administrasi merupakan pelaksanaan kegiatan-kegiatan untuk mencapai tujuan yang telah ditetapkan atau secara umum dapat disebut sebagai penyelenggara kerja. Di masa modern sekarang, administrasi merupakan kunci dalam pelaksanaan kegiatan atau alat yang paling dan perlu dilaksanakan.

Pengertian administrasi yang bersifat sempit hampir menyerupai pengertian "tata usaha" pada zaman masa penjajahan Belanda, dimana kata administrasi masih disamakan dengan kata "*administratie*"(bahasa Belanda) yaitu tata usaha yang artinya adalah kegiatan penyusunan keterangan-keterangan dan pencataan secara tertulis dan sistematis yang diperlukan untuk memperoleh suatu ikhtisar dari keseluruhan kegiatan dan lainnya yang terbatas pada kegiatan kantor (*office work*) (ATM[2]).

Menurut Prof. Dr. Prayudi Atmosoedirdjo SH., ATM[2], pengertian administrasi dalam arti luas dapat dilihat dari tiga sudut, yaitu :

1. Sudut Proses : administrasi merupakan keseluruhan proses yang terdiri dari kegiatan-kegiatan, pemikiran-pemikiran, pengaturan-pengaturan mulai dari

penentuan tujuan sampai dengan pelaksanaannya, sehingga tujuan tersebut dapat tercapai.

2. Sudut Fungsional : administrasi merupakan segala kegiatan yang secara sadar dilakukan kearah tercapainya tujuan yang bersifat melihat ke depan, merencanakan, memikirkan tujuan dan memeperkirakannya serta menentukan kebijaksanaannya (*policy*), memberikan kepemimpinan (*leadership*), mengambil keputusan-keputusan (*decision making*), memberikan bimbingan (*guidance*) dan mengatur (*to manage*) orang-orang, alat, waktu, pekerjaan dan sebagainya.
3. Sudut Institusional : administrasi adalah kumpulan orang-orang, baik secara perseorangan ataupun kelompok yang menjalankan proses serta kegiatan-kegiatan kearah tercapainya tujuan yang telah ditentukan sebelumnya.

Dari uraian diatas, dapat diketahui pengertian administrasi kesiswaan serta maksud dari administrasi kesiswaan tersebut. Administrasi kesiswaan merupakan suatu kegiatan penyelanggaraan kerja yang mengatur, memikirkan kebijakan, melaksanakan keputusan-keputusan bagi siswa mengenai tujuan atau sasaran yang ingin dicapai.