BAB II

LANDASAN TEORI

2.1 Konsep Dasar Sistem

Definisi sistem menurut JOG[4] adalah sebagai berikut:

Terdapat dua kelompok pendekatan di dalam pendefinisian sistem, yaitu yang menekankan pada prosedur dan menekankan pada komponen atau elemen. Pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem sebagai berikut:

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

Pendekatan sistem yang menekankan pada komponen atau elemen-elemen mendefinisikan sebagai berikut:

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

Definisi sistem menurut ABD[1] adalah sebagai berikut:

"*Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu dan apabila terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama maka elemen tersebut dapat dipastikan bukan bagian dari sistem".*

2.1.1 Karakteristik Sistem

Suatu sistem dapat mempunyai karakteristik atau sifat-sifat tertentu, diantaranya adalah:

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi yang artinya saling bekerja sama untuk membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

2. Batas Sistem

   Batasan sistem *(boundary)* merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan. Batas suatu sistem merupakan ruang lingkup dari sistem tersebut.

3. Lingkungan Luar Sistem

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

4. Penghubung Sistem

   Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari suatu subsistem ke subsistem yang lainnya. Keluaran *(output)* dari suatu subsistem akan menjadi masukan *(input)* untuk subsistem yang lainnya dengan melalui penghubung.

5. Masukan *(Input)*

   Input merupakan elemen dari sistem yang bertugas untuk menerima seluruh masukan data, dimana masukan tersebut dapat berupa jenis data, frekuensi pemasukan data dan sebagainya.

6. Keluaran *(Output)*

   Output merupakan elemen dari sistem yang bertugas untuk mengolah atau memproses seluruh masukan data menjadi suatu informasi yang lebih berguna.

7. Pengolah Sistem

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

8. Sasaran Sistem

   Suatu sistem pasti mempunyai tujuan atau sasaran. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

2.1.2 Klasifikasi Sistem

Sistem dapat diklasifikasikan berdasarkan beberapa sudut pandang, diantaranya:

1. Sistem diklasifikasikan sebagai sistem abstrak dan sistem fisik.

   Sistem abstrak merupakan sistem yang tidak bisa dilihat secara mata biasa dan biasanya sistem ini berupa pemikiran atau ide-ide. Sistem fisik merupakan sistem yang dapat dilihat secara mata biasa/telanjang dan biasanya sering digunakan oleh manusia.

2. Sistem diklasifikasikan sebagai sistem alamiah dan sistem buatan manusia.

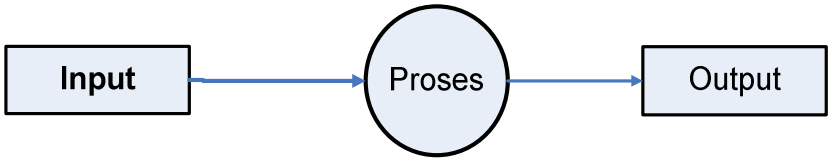
   Sistem alamiah merupakan sistem yang terjadi karena pengaruh alam. Sistem buatan merupakan sistem yang dirancang dan dibuat oleh manusia.

3. Sistem diklasifikasikan sebagai sistem tertutup dan sistem terbuka.

Sistem tertutup merupakan sistem yang tidak berhubungan dengan bagian luar sistem. Sedangkan sistem terbuka adalah sistem yang berhubungan langsung dengan bagian luar sistem.

2.1.3 Bentuk Umum Sistem

Bentuk umum Sistem dari suatu sistem terdiri atas Masukan (*Input*), Proses dan Keluaran (*Output*), dalam bentuk umum Sistem ini terdapat satu atau lebih masukan yang akan diproses dan akan menghasilkan suatu keluaran.



Gambar 2.1 Bentuk Umum Sistem
[Sumber : JOG[4]]

2.2 Konsep Dasar Informasi

Informasi sangat penting bagi perusahaan. Selain itu informasi juga penting bagi sistem, karena apabila suatu sistem tidak mempunyai informasi maka akan menjadi suatu sistem yang susah berkembang. Defenisi informasi menurut JOG[4] adalah sebagai berikut: "*Data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya*".

Menurut Joku Bruch dan Gary Grudnitski Sumber JOG[4] mengatakan bahwa kualitas sistem terdiri dari tiga hal, yaitu:

a. Akurat

Informasi harus bebas dari kesalahan dan tidak menyesatkan.

b. Tepat pada waktunya

Informasi yang datang pada penerima tidak boleh terlambat.

c. Relevan

Informasi tersebut mempunyai manfaat untuk pemakainya.

2.3 Konsep Dasar Sistem Informasi

Telah diketahui bahwa informasi merupakan hal yang sangat penting bagi manajemen di dalam pengambilan keputusan. Pertanyaannya adalah dari mana informasi tersebut bisa didapatkan?. Informasi dapat diperoleh dari sistem informasi (*information Sistem*) atau disebut juga dengan *processing sistems* atau *information processing Sistem* atau *information-enginering sistems*. Sistem informasi didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis Sumber JOG[4] sebagai berikut:

"*Sistem Informasi adalah 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-laporan yang diperlukan.*

Adapun pengertian sistem informasi menurut Joku Bruch dan Gary Grudnitski Sumber JOG[4] adalah sebagai berikut:

"Suatu sistem dalam suatu organisasi yang merupakan kombinasi dari orang-orang, fasilitas, teknologi, media, prosedur-prosedur dan pengendalian untuk mendapatkan jalur komunikasi penting, memproses tipe transaksi rutin tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian internal dan eksternal yang penting dan menyediakan suatu dasar informasi untuk pengambilan keputusan".

2.3.1 Komponen Sistem Informasi

Sistem informasi sama dengan sistem terotomatisasi, terdiri dari beberapa komponen, antara lain:

1. Hardware : CPU, Disk, terminal, printer.
2. Software : Sistem operasi, sistem database, progam pengontrol komunikasi, program aplikasi.
3. Personil : Orang yang mengoperasikan sistem, menyediakan masukan, mengkonsumsi keluaran dan melakukan aktivitas manual yang mendukung sistem.
4. Data : Data yang tersimpan dalam jangka waktu tertentu.
5. Prosedur : Instruksi dan kebijakan untuk mengoperasikan sistem.

2.3.2 Sistem Informasi Pembelian

Pada hakekatnya syarat pembelian tidak berbeda dengan syarat penjualan. Sebetulnya, syarat pembelian adalah syarat jual beli yang dipandang dari sudut pembeli. Buku pembelian adalah buku harian yang digunakan khusus untuk mencatat pembelian secara kredit.

Pengertian pembelian dalam suatu perusahaan menurut R, Soemarno S. Sumber MID[6] yaitu:

"Membeli aktiva produksi untuk digunakan dalam kegiatan perusahaan yang digunakan untuk mencatat pembelian barang dagang selama satu periode, hanya digunakan untuk mencatat pembelian barang dagang saja (pembelian aktiva produksi, misalnya: peralatan kantor)".

Kegiatan pembelian dalam suatu perusahaan dagang meliputi hal-hal sebagai berikut:

1. Membeli barang dagang secara tunai dan kredit.
2. Membeli aktiva produksi untuk digunakan dalam kegiatan perusahaan. Contoh: kegiatan pembelian kendaraan, peralatan kantor dan lain-lain.
3. Membeli barang dan jasa-jasa lain sehubungan dengan kegiatan perusahaan. Contoh: biaya pengiriman, biaya listrik serta biaya air dan telepon.

2.3.2.1 Prosedur Pembelian

Prosedur pembelian mengatur cara-cara dalam melakukan semua pembelian baik barang maupun jasa yang dibutuhkan oleh perusahaan. Prosedur ini dimulai dari adanya kebutuhan atas suatu barang atau jasa sampai barang atau jasa yang dibeli diterima.

Fungsi dari bagian-bagian yang terkait dalam prosedur pembelian menurut Baridwan, Z. Sumber MID[6] adalah sebagai berikut:

1. Bagian Pembelian

   Bagian pembelian berfungsi untuk melakukan pembelian barang-barang dan jasa yang dibutuhkan oleh perusahaan, seperti:

   a. Mesin, alat-alat, dan suku cadang
   b. Bahan baku, bahan penolong, supplies pabrik, dan supplies kantor
   c. Jasa serta barang-barang lainnya

   Untuk menjalankan fungsi ini, bagian pembelian sebelum mengeluarkan order pembelian harus melakukan langkah-langkah untuk menjamin bahwa:

Pembelian dilakukan dengan harga yang menguntungkan perusahaan dan kualitas yang sesuai. Barang-barang yang dibeli akan dapat diterima tepat pada waktu yang dibutuhkan.

2. Bagian Penerimaan Barang

   Bertugas untuk menerima semua barang yang dibeli perusahaan pada waktu menerima barang, bagian ini harus melakukan perhitungan fisik atas barang-barang yang diterima baik dengan cara menghitung, menimbang atau dengan cara yang lain.

   Disamping itu bagian penerimaan juga harus mengadakan pemeriksaan kualitas barang yang diterima surat pengangkutan barang dari pihak pengangkut ditandatangani oleh bagian penerimaan barang.

3. Bagian Gudang

   Bagian gudang bertugas untuk menyimpan barang-barang milik perusahaan. Penyimpanan barang dalam gudang harus disusun sedemikian rupa supaya memudahkan pada waktu dibutuhkan. Untuk mengawasi barang dalam gudang, bagian gudang menyelenggarakan pencatatan dalam kartu gudang dan kartu barang. Catatan ini hanya menunjukkan kuantitas tanpa jumlah rupiah.

4. Bagian Utang

   Bagian utang bertugas membandingkan faktur pembelian dengan laporan penerimaan barang. Apabila kedua dokouman tersebut menunjukkan data yang sama, bagian utang menentukan apakah ada potongan atau tidak dalam hal faktur pembelian ini informasi tentang potongan maka bagian utang bertugas menghitung jumlah potongan pembelian.

2.3.3 Sistem Informasi Penjualan

Penjualan berhubungan erat dengan pemasaran, tetapi penjualan hanyalah merupakan segi dari pemasaran artinya dalam suatu penjualan hanya memikirkan dan mengurusi soal pertukaran baik berupa barang maupun jasa.

Sehingga dari sini setiap perusahaan tentunya akan menyusun suatu akuntansi penjualan. Penyusunan akuntansi penjualan menurut sumber MID[6] ini bertujuan untuk mengetahui:

a. Penjualan merupakan sumber pendapatan perusahaan
b. Pendapatan dari penjualan merupakan sumber pendanaan perusahaan
c. Akibat penjualan akan menambah posisi harta pada perusahaan

Adapun klasifikasi transaksi penjualan dibagi menjadi:

1 Penjualan secara tunai
2 Penjualan secara kredit
3 Penjualan secara tandom
4 Penjualan secara ekspor
5 Penjualan secara konsinyasi
6 Penjualan secara grosir

Seperti dikemukakan sebelumnya bahwa tujuan penjualan adalah melakukan transaksi jual beli yang sebanyaknya, adapun jenis-jenis penjualan yang dilakukan diantaranya:

a. Penjualan secara percobaan, ialah penjualan baru terjadi apabila pihak pembeli telah menyatakan persetujuannya terhadap barang yang dijual.

b. Penjualan dengan contoh, ialah penjualan yang terjadi atas dasar contoh barang tersebut baik mengenai barang maupun kualitasnya dan sesuai harganya.
c. Penjualan dengan cicilan atau angsuran, ialah pembeli membayar harga barang dengan beberapa kali bayaran.
d. Penjualan dengan kontan, dalam hal ini harga dibayar tunai barang dibawa langsung oleh pembeli.

2.4 Metode Pengembangan Sistem dengan Model *Prototype*

Metode atau paradigma perangkat lunak yang digunakan untuk perancangan sistem informasi akuntansi pada PD Anugrah Grosir adalah metode *prototype*. Adapun tahapan-tahapan dari metode *prototype* menurut RAY[9] adalah sebagai berikut:

a. Mengidentifikasi kebutuhan pemakai. Pada tahap ini analisis sistem akan melakukan studi kelayakan dan studi terhadap kebutuhan pemakai, baik yang meliputi model *interface*, teknik prosedural, maupun dalam teknologi yang akan digunakan.

b. Mengembangkan *prototype*. Pada tahap kedua ini, analisa sistem bekerja sama dengan pemrogram dalam mengembangkan *prototype* sistem untuk memperlihatkan kepada pemesanan permodelan sistem yang akan dibangunnya.

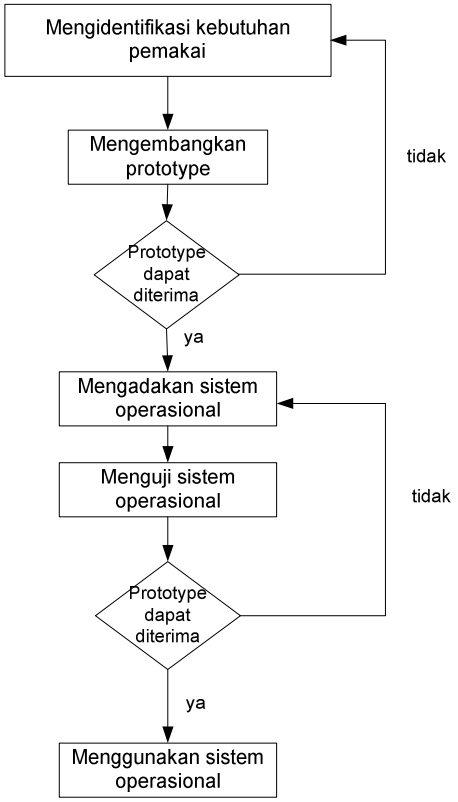
c. Menentukan apakah *prototype* dapat diterima oleh pemesan atau pemakai.

d. Mengadakan sistem operasional. Melalui pemrogram berdasarkan permodelan sistem yang telah disepakati oleh pemesan sistem.

e. Menguji sistem operasional. Pada tahap ini pemrogram akan melakukan uji coba baik menggunakan data sekunder maupun data primer untuk memastikan bahwa sistem dapat berlangsung dengan baik dan benar, sesuai dengan kebutuhan pemesan.

f. Menentukan sistem operasional. Apakah dapat diterima oleh pemesan atau harus dilakukan beberapa perbaikan atau bahkan harus dibongkar semuanya dimulai dari awal lagi.

g. **Jika sistem telah disetujui,** maka tahap terakhir adalah melakukan implementasi sistem.

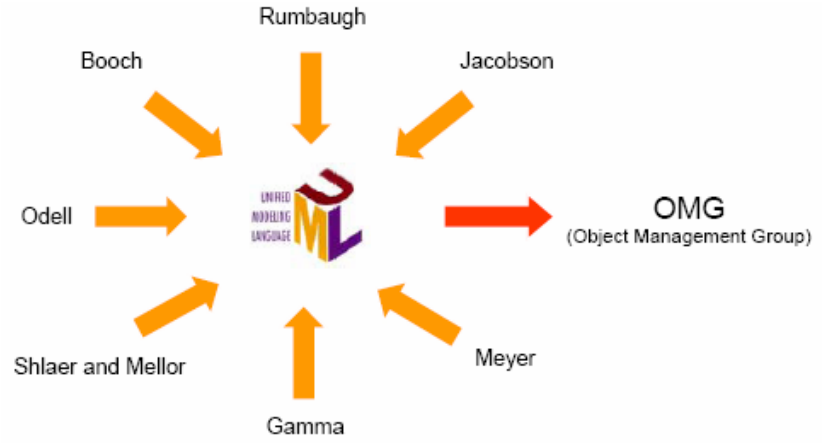


Gambar 2.2 Metode *Prototype*
[Sumber: RAY[9]]

2.5 Pengenalan UML

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch [1], metodologi coad [2], metodologi OOSE [3], metodologi OMT [4], metodologi shlaer-mellor [5], metodologi wirfs-brock [6], dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri yang mengakibatkan timbul masalah baru apabila kita bekerja sama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan.

Dimulai pada bulan Oktober 1994 Booch, Rumbaugh dan Jacobson, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 di *release* draft pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh ***Object Management Group*** (OMG – http://www.omg.org). Tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. Booch, Rumbaugh dan Jacobson menyusun tiga buku serial tentang UML pada tahun 1999 [7] [8] [9]. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek.



Gambar 2.3 Unsur-unsur Pembentuk UML
[Sumber: ILM[15]]

2.5.1 Pengertian UML

Pengertian *Unified Modelling Language* (UML) menurut sumber ILM[15] adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan perangkat lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk

menggambarkan berbagai diagram perangkat lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya, yaitu: Grady Booch OOD *(Object-Oriented Design)*, Jim Rumbaugh OMT *(Object Modeling Technique)*, dan Ivar Jacobson OOSE *(Object-Oriented Software Engineering)*.

2.5.2 Bagian-bagian UML

Bagian-bagian utama dari UML adalah *view, diagram, 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 adalah *use case view, logical view, component view, concurrency view,* dan *deployment view.*

a. Use Case View

Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan oleh *external actors*. *Actor* yang berinteraksi dengan sistem dapat berupa user atau sistem lainnya. *View* ini digambarkan dalam *use case diagrams* 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 *relationship*) dan kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu.

View ini digambarkan dalam *class diagrams* untuk struktur statis dan dalam *state, sequence, collaboration,* dan *activity diagram*s untuk model dinamisnya. View ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

c. Component View

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administratif lainnya.

View ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

d. Concurrency View

Membagi sistem kedalam proses dan prosesor. View ini digambarkan dalam diagram dinamis (*state, sequence, collaboration,* dan *activity diagrams*) 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 (*nodes*) dan bagaimana hubungannya dengan yang lain. View ini digambarkan

dalam *deployment diagrams* dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2. Diagram

Diagram berbentuk grafis yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu. Adapun jenis diagramnya antara lain:

a. Use Case Diagram

Menggambarkan sejumlah *external actors* dan hubungannya ke *use case* yang diberikan oleh sistem. *Use case* adalah deskripsi fungsi yang disediakan oleh sistem kedalam bentuk teks sebagai dokumentasi dari *use case symbol* namun dapat juga dilakukan dalam *activity diagrams*.

Use case digambarkan hanya yang dilihat dari luar oleh *actor* (keadaan lingkungan sistem yang dilihat user) dan bukan bagaimana fungsi yang ada di dalam sistem. Simbol yang digunakan yaitu:



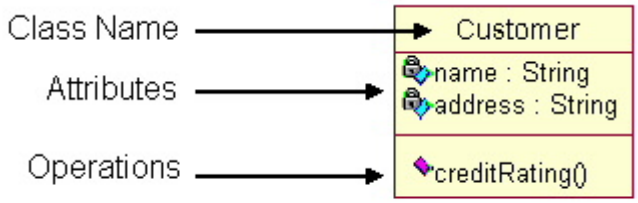
Gambar 2.4 *Use Case* Model
[Sumber: ATL[14]]

b. Class Diagram

Menggambarkan struktur statis *class* di dalam sistem. *Class* merepresentasikan sesuatu yang ditangani oleh sistem. *Class* dapat berhubungan dengan yang lain melalui berbagai cara: *associated*

(terhubung satu sama lain), *dependent* (satu *class* tergantung/menggunakan *class* yang lain), *specialed* (satu *class* merupakan spesialisasi dari *class* lainnya) atau *package* (grup bersama sebagai satu unit). Sebuah sistem biasanya mempunyai beberapa *class diagram.*

Suatu *class* biasanya terdiri dari 3 bagian, yaitu nama, atribut, dan operasi. Berikut adalah contoh dari suatu *class*:

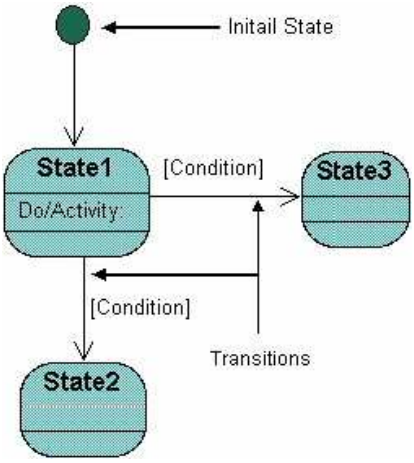


Gambar 2.5 Bagian-bagian dari C*lass*
[Sumber: ATL[14]]

c. State Diagram

Menggambarkan semua *state* (kondisi) yang dimiliki oleh suatu *object* dari suatu *class* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa *object* lain yang mengirim pesan.

State class tidak digambarkan untuk semua *class*, hanya yang mempunyai sejumlah *state* yang terdefinisi dengan baik dan kondisi *class* berubah oleh *state* yang berbeda.

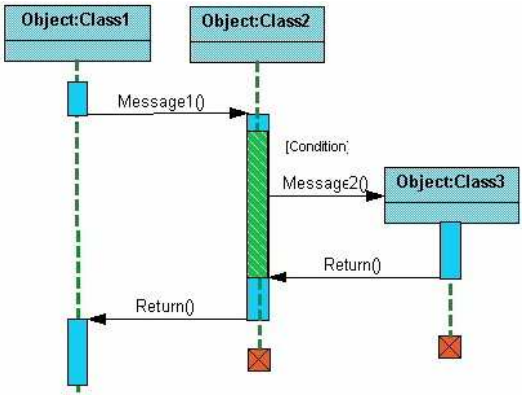


Gambar 2.6 *State Diagram*
[Sumber: ATL[14]]

d. Sequence Diagram

Menggambarkan kolaborasi dinamis antara sejumlah *object*. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object*, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

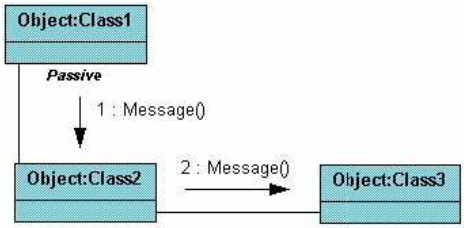
Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat bernama pesan diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan proses vertikal.



Gambar 2.7 *Sequence Diagram*
[Sumber: ATL[14]]

e. Collaboration Diagram

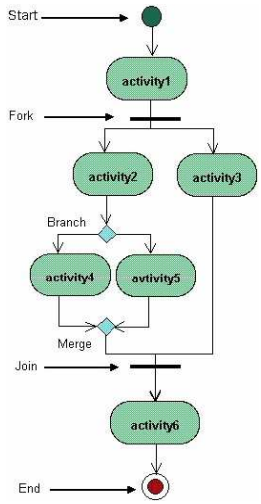
Menggambarkan kolaborasi dinamis seperti *sequence diagrams*. Dalam menunjukkan pertukaran pesan, *collaboration diagrams* menggambarkan object dan hubungannya (mengacu ke konteks). Jika penekanannya pada waktu atau urutan maka gunakan *sequence diagrams*, tapi jika penekanannya pada konteks gunakan *collaboration diagrams*.



Gambar 2.8 *Collaboration Diagram*
[Sumber: ATL[14]]

f. Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.



Gambar 2.9 *Activity Diagram*
[Sumber: ATL[14]]

g. Component Diagram

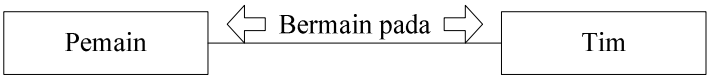
Menggambarkan struktur fisik kode dari komponen. Komponen dapat berupa *source code*, komponen biner atau *executeable component*. Sebuah komponen berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view.*

h. Deployment Diagram

Menggambarkan arsitektur fisik dari perangkat keras dan perangkat lunak sistem, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes, executeable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh *node* tertentu dan ketergantungan komponen.

2.5.3 *Indicator Multiplicity*

Multiplicity pada kasus asosiasi menunjukkan bahwa ada sejumlah objek pada sebuah *class* yang berhubungan dengan sebuah objek pada sebuah asosiasi *class*. Untuk menggambarkan hal tersebut bias dilihat pada fakta bahwa sebuah tim mungkin terdiri dari beberapa pemain. Untuk menunjukkan hal itu angka harus diletakkan di dekat *class* yang dimaksud.



Gambar 2.10 *Multiplicity* pada Asosiasi
[Sumber: Mun[9]]

Ada banyak multiplicity yang mungkin untuk dipakai. Sebuah class dapat berhubungan denga yang lain dalam satu ke satu *(one-to-one)*, satu ke banyak

(*one-to-many*), satu ke satu atau lebih (*one-to-one or more*), satu ke nol atau satu, satu ke sebuah interval tertentu (contoh satu ke lima sampai dengan sepuluh), satu ke sejumlah n atau satu ke serangkaian pilihan (contoh satu ke sembilan atau sepuluh).

Pada notasi UML, untuk menampilkan lebih atau banyak digunakan (*). Untuk menunjukkan 'atau' digunakan titik dua (..) seperti contohnya 1..* (berarti satu atau lebih). Notasi 'atau' bisa juga digunakan tanda koma (,). Contoh dari *multiplicity* diatas dapat dilihat pada gambar berikut:



Gambar 2.11 Macam-macam *multiplicity*
[Sumber: Mun[9]]

2.6 Basis Data

Basis data *(database)* menurut sumber FAT[3] merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di simpanan luar komputer dan digunakan perangkat lunak tertentu untuk memanipulasinya.

Database merupakan salah satu komponen yang penting dalam suatu sistem informasi, karena berfungsi sebagai basis penyedia informasi bagi para pemakainya. Penerapan *database* dalam sistem informasi disebut dengan *database system*.

Sistem basis data *(database system)* ini adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan satu dengan lainnya dan membuatnya tersedia untuk beberapa aplikasi yang bermacam-macam di dalam suatu organisasi.

Tujuan dari desain *database* adalah untuk menentukan data-data yang dibutuhkan dalam sistem, sehingga informasi yang dihasilkan dapat terpenuhi dengan baik. Terdapat beberapa alasan mengapa desain *database* perlu untuk dilakukan, salah satunya adalah untuk menghindari pengulangan data (*redudancy* data).

2.6.1 Sistem Pengelola Basis Data (DBMS)

Pengelolaan basis data secara fisik tidak dilakukan oleh pemakai secara langsung, tetapi ditangani oleh sebuah perangkat lunak (sistem) yang khusus/spesifik. Perangkat lunak inilah yang disebut dengan (*Database Manajemen Sistem* (DBMS) yang akan menetukan bagaimana data diorganisasi, disimpan, diubah dan diambil kembali. Perangkat lunak ini juga menerapakan mekanisme pengamanan data, pemakaian data secara bersamaan, keakuratan/ konsistensi data, dan sebagainya.

2.6.2 Perancangan Basis Data

Sebelum kita membuat basis data, terlebih dahulu harus dilakukan suatu perancangan. Proses perancangan ini bersifat konseptual. Kita belum menentukan DBMS apa yang akan kita gunakan untuk mengimplementasikan rancangan basis data yang dibuat. Tujuan perancangan basis data adalah untuk mendapatkan skema basis data yang meminimasi terjadinya redudansi dan duplikasi data serta menjaga integritas data. Kebanyakan metode perancangan yang ada berbasis pada model basis data relasional, yaitu struktur data diatur melalui pembuatan tabel-tabel dan keterkaitan antar tabel satu dengan yang lainnya *(*relasi*)*.

2.7 Arsitektur Aplikasi

Untuk mendukung kelancaran program aplikasi yang akan dibangun ini, maka diperlukan beberapa unit komputer, dimana semua komputer tersebut saling berhubungan dan melakukan komunikasi data. Agar proses komunikasi data tersebut dapat berjalan dengan baik maka dibutuhkan sebuah media atau perantara, yang disebut dengan jaringan komputer.

2.7.1 Pengertian Jaringan Komputer

Pengertian jaringan komputer menurut sumber ILM[15] adalah sebuah kumpulan komputer, printer dan peralatan lainnya yang saling terhubung. Informasi dan data bergerak melalui kabel-kabel sehingga memungkinkan pengguna jaringan komputer dapat saling bertukar dokumen dan data, mencetak pada printer yang sama dan bersama-sama menggunakan *hardware/software* yang terhubung dengan jaringan. Tiap komputer, printer atau *periferal* yang terhubung

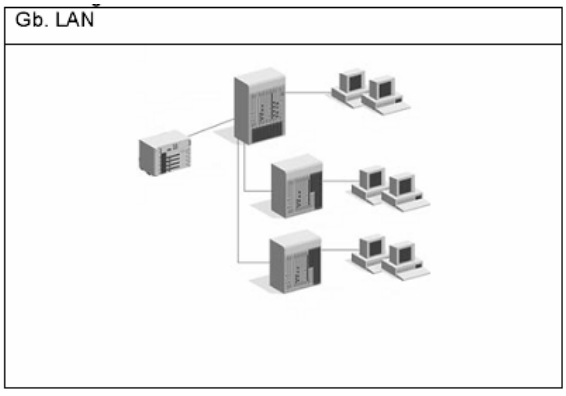
dengan jaringan disebut *node*. Sebuah jaringan komputer dapat memiliki dua, puluhan, ribuan atau bahkan jutaan node. Sebuah jaringan biasanya terdiri dari 2 atau lebih komputer yang saling berhubungan antara satu dengan yang lainnya, dan saling berbagi sumber daya misalnya CD-ROM, printer, pertukaran file, atau memungkinkan tiap komputer untuk dapat saling berkomunikasi secara elektronik. Komputer yang terhubung tersebut, dimungkinkan berhubungan dengan media kabel, saluran telepon, gelombang radio, satelit, atau sinar infra merah.

2.7.2 Jenis-jenis Jaringan Komputer

Ada beberapa jenis jaringan komputer, diantaranya yaitu:

1. *Local Area Network* (LAN)

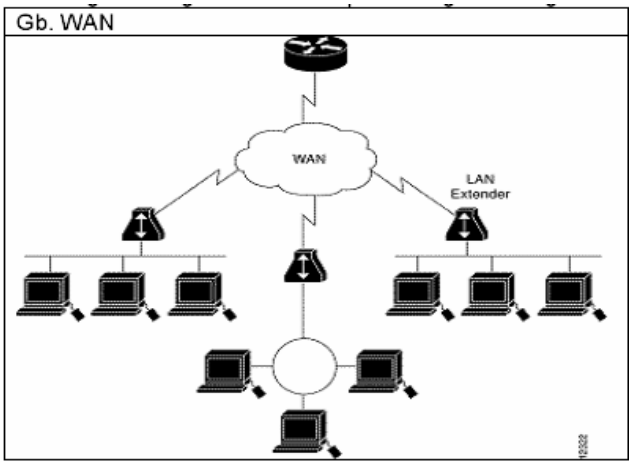
   Jaringan komputer yang secara geografis terletak berdekatan dan saling berhubungan.



Gambar 2.12 Contoh Gambar Jaringan LAN
[Sumber: ILM[15]]

2. *Wide Area Network* (WAN)

   Jaringan computer yang secara geografis terpisah jarak yang jauh dan dihubungkan dengan saluran telepon atau gelombang radio.



Gambar 2.13 Contoh Gambar Jaringan WAN
[Sumber: ILM[15]]

3. *Metropolitan Area Network* (MAN)

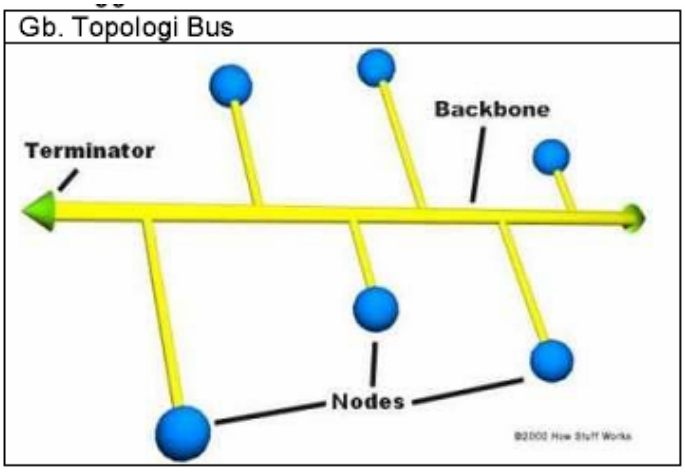
   Jaringan komputer yang menghubungkan komputer-komputer di seluruh dunia. Lebih dikenal sebagai internet.

2.7.3 Topologi Jaringan Komputer

Topologi jaringan adalah cara pengelompokan/konfigurasi elemen-elemen dari jaringan. Dua jaringan dikatakan mempunyai topologi yang sama bila konfigurasi koneksinya sama, meskipun secara fisik, jarak, dan transmisinya beda. Beikut adalah beberapa jenis topologi jaringan, yaitu:

1. Bus

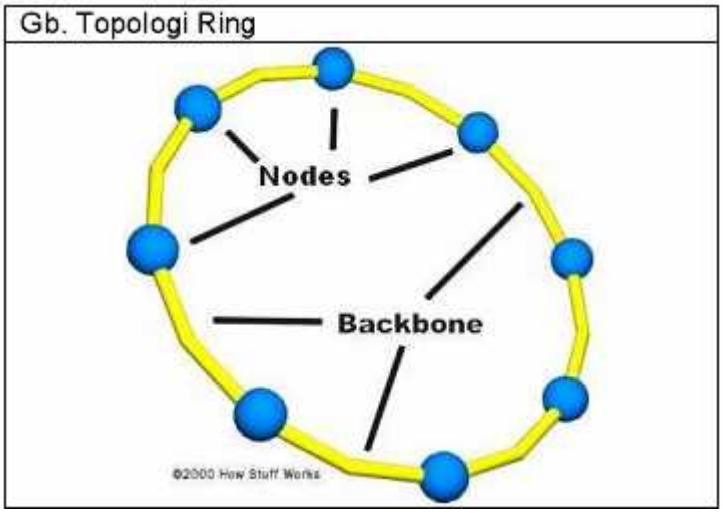
   Topologi jaringan yang konfigurasi koneksi semua nodenya terhubung menggunakan bus tunggal.



Gambar 2.14 Contoh Gambar Topologi Jaringan Bus
[Sumber: ILM[15]]

2. Ring

Topologi jaringan yang konfigurasi koneksi masing-masing nodenya ada dua. Bila diurutkan, koneksi ini akan berbentuk melingkar.

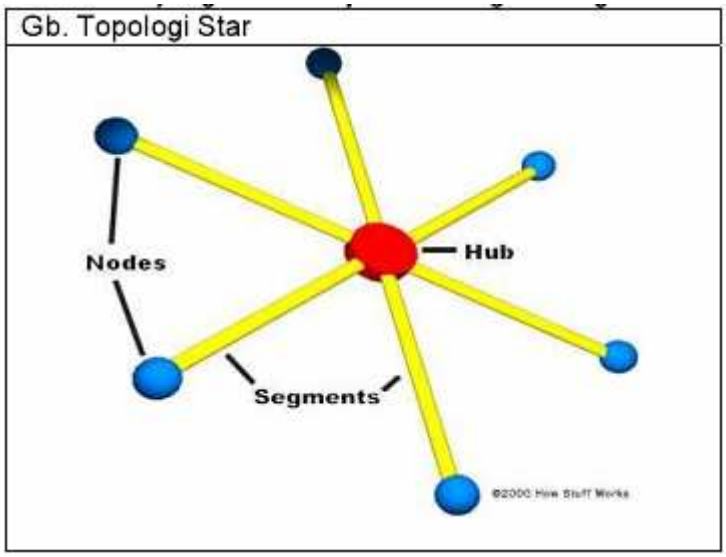


Gambar 2.15 Contoh Gambar Topologi Jaringan Ring
[Sumber: ILM[15]]

3. Star

Topologi jaringan yang konfigurasi koneksinya adalah semua node terhubung pada satu node pusat, yang biasanya menggunakan peralatan hub atau switch.

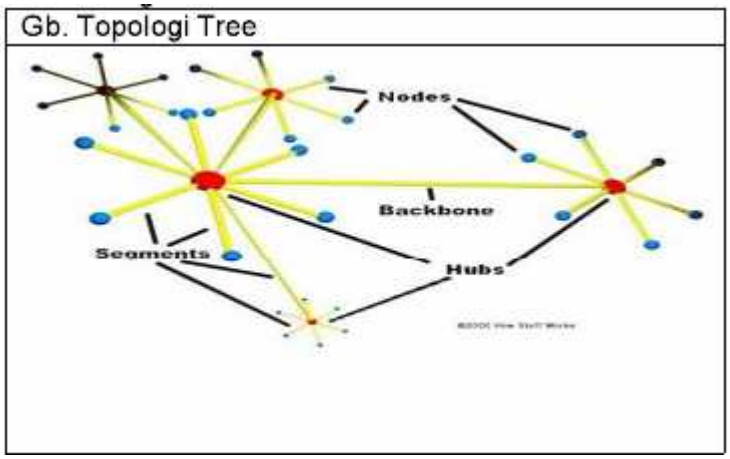
Node pusat ini yang melakukan rebroadcast semua transmisi yang diterima dari node-node yang ada ke tujuan masing-masing.



Gambar 2.16 Contoh Gambar Topologi Jaringan Star
[Sumber: ILM[15]]

4. Tree

Topologi jaringan yang konfigurasinya adalah gabungan dari beberapa topologi star sekaligus.



Gambar 2.17 Contoh Gambar Topologi Jaringan Tree
[Sumber: ILM[15]]

2.7.4 Manfaat Jaringan Komputer

Ada banyak keuntungan/manfaat yang didapat dari jaringan komputer menurut Andrews TAN[13], diantaranya adalah:

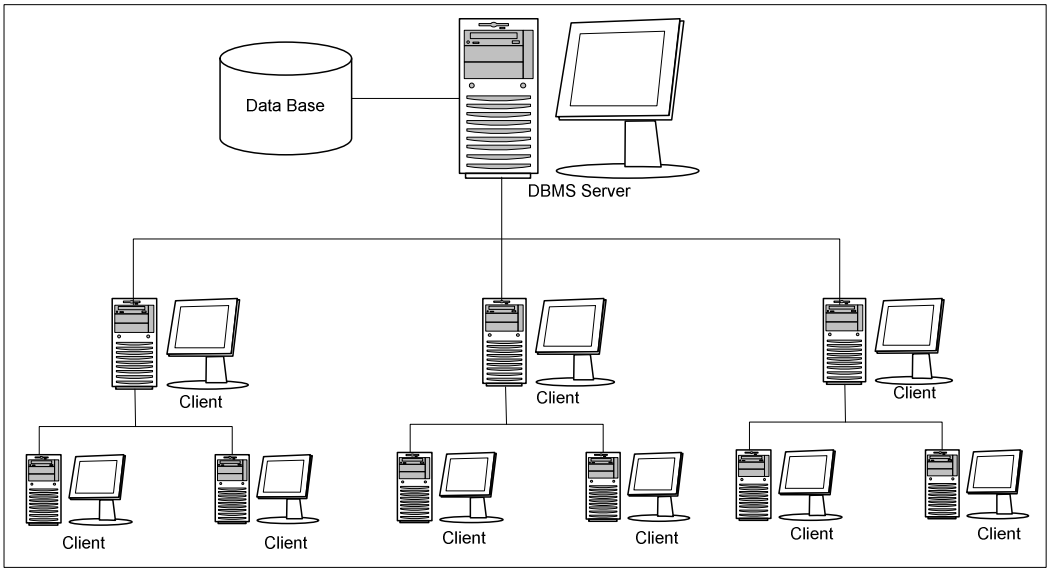
1. *Resource Sharing*, dapat menggunakan sumber daya yang ada secara bersama-sama. Misal, seorang pengguna yang berada 100 km jauhnya dari suatu data, tidak mendapatkan kesulitan dalam menggunakan data tersebut, seolah-olah data tersebut berada didekatnya. Hal ini sering diartikan bahwa jaringan komputer mangatasi masalah jarak.
2. Reliabilitas tinggi, dengan jaringan komputer kita akan mendapatkan reliabilitas yang tinggi dengan memiliki sumber-sumber alternatif persediaan. Misalnya, semua file dapat disimpan atau dicopy ke dua, tiga atu lebih komputer yang terkoneksi ke jaringan. Sehingga bila salah satu mesin rusak, maka salinan di mesin yang lain bisa digunakan.
3. Menghemat uang. Komputer berukuran kecil mempunyai rasio harga/kinerja yang lebih baik dibandingkan dengan komputer yang besar. Komputer besar seperti mainframe memiliki kecepatan kira-kira sepuluh kali lipat kecepatan komputer kecil/pribadi. Akan tetap, harga mainframe seribu kali lebih mahal dari komputer pribadi. Ketidakseimbangan rasio harga/kinerja dan kecepatan inilah membuat para perancang sistem untuk membangun sistem yang terdiri dari komputer-komputer pribadi.

2.8 Pengertian *Client / Server*

Client server menurut Spencerken Miller Ken, MIL[7] merupakan komputer yang memanfaatkan sumber daya dalam jaringan yang disediakan oleh

komputer lainnya, yang disebut dengan server. Client juga merupakan sebuah aplikasi atau proses yang meminta pelayanan dari komponen atau proses lainnya. Server merupakan perangkat khusus dalam jaringan komputer yang menjadi tempat bagi semua nodes di dalam jaringan untuk bisa melakukan *resource sharing*. *Client server* merupakan model jaringan yang memisahkan secara jelas mana yang dapat memberikan layanan jaringan (*server*) dan mana yang hanya menerima layanan (*client*).

Arsitektur adalah desain sebuah aplikasi. Arsitektur *client/server* merupakan arsitektur terdistribusi artinya pemrosesan dari suatu aplikasi terjadi pada lebih satu mesin. Dalam arsitektur *client/server* aplikasi dipecah menjadi koponen-komponen yang bekerja sama untuk mencapai tujuan.



Gambar 2.18 Sistem *Client-Server* Kompleks
[Sumber : FAT[3]]

2.9 Perangkat Lunak Pendukung

Perangkat Lunak (*Software)* pendukung sangatlah dibutuhkan dalam membangun sistem aplikasi ini, karena sistem aplikasi yang akan dibangun ini

membutuhkan beberapa program aplikasi yang digunakan untuk menghasilkan program aplikasi yang lengkap yang sesuai dengan yang dibutuhkan oleh pengguna. Adapun program apliksi yang digunakan oleh program aplikasi yang akan dibangun ini adalah *Microsoft Visual Basic 6.0, SQL Server 2000* dan *Crystal Report 8.5.*

2.9.1 Sekilas Tentang Visual Basic

Microsoft Visual Basic 6 atau yang sering disebut dengan VB adalah sebuah aplikasi pemrograman, selain itu VB sering disebut juga sebagai sarana *(Tool)* untuk menghasilkan program-program aplikasi berbasiskan windows. Bahasa VB sangat sederhana dan menggunakan bahasa inggris yang umum digunakan, sehingga bagi penggunanya tidak perlu lagi menghapalkan sintaks-sintaks ataupun format-format bahasa yang bermacam-macam. Di dalam VB objek-objek dalam pemrograman semuanya sudah disediakan dalam pilihan-pilihan komponen yang dapat diambil sesuai dengan kebutuhan penggunanya, selain itu sarana untuk pembangunannya yang bersifat visual memudahkan penggunanya untuk mengembangkan program aplikasi berbasis windows. VB juga bersifat mouse driven (digerakkan dengan mouse), dan berdaya guna tinggi.

Bahasa pemrograman VB merupakan bagian dari bahasa BASIC *(Beginers All Purpose Symbolic Instruction Code).* Sebuah bahasa pemrograman yang penulisan sintaksnya menggunakan bahasa yang umum digunakan (dalam bahasa inggris), sehingga bagi para programmer yang masih dalam tahap belajar dapat lebih mudah mempelajarinya. Kesimpulannya bahasa VB adalah sebuah sarana pembuat program yang lengkap namun mudah. Siapapun yang bisa menggunakan

windows, dipastikan tidak akan mengalami kesulitan di dalam mempelajari VB. Hanya dengan menggunakan mouse, memanipulasi jendela, serta logika pemrograman kita sudah dapat membuat sebuah program aplikasi dengan VB.

2.9.1.1 Lingkungan Visual Basic 6

VB 6 juga merupakan sebuah program aplikasi yang berbasis windows. didalam VB tedapat objek-objek yang digunakan untuk membantu penggunanya dalam membuat suatu program aplikasi. objek-objek tersebut diantaranya adalah:

1. Form

   Form merupakan sebuah objek di dalam aplikasi VB. form biasa disebut dengan objek ayah (*parent object*) karena form tempat untuk meletakkan objek-objek lainnya dalam aplikasi, form inilah yang nantinya akan menjadi latar belakang dari program aplikasi yang dibuat oleh pengguna (*user*)

2. Label

   Label berfungsi untuk menampilkan text pada program aplikasi yang akan dibuat. label biasa digunakan untuk menampilkan judul teks, penjelas di sebelah kotak teks *(textbox*), keterangan dan sebagainya.

3. *Command Button*

   Tombol perintah (*command button*) hampir selalu muncul pada setiap aplikasi. Tombol perintah dalam penampilannya tampak seperti sebuah segi empat dengan teks diatasnya. Fungsi dari tombol ini biasanya untuk mengeksekusi perintah tertentu seperti:

   a. Menyetujui atau menjalankan suatu perintah (tombol Ok)

   b. Keluar dari aplikasi (tombol *EXIT*)

c. Membatalkan perintah (tombol *CANCEL*)

d. Membatalkan file (tombol *OPEN*), dan lain sebagainya

4. Text Box

Kotak teks (*TextBox*) adalah sebuah objek kontrol yang menerima input dari pengguna (user) dalam bentuk teks yang di ketik di dalam kotak teks, biasanya objek ini digunakan untuk memasukkan informasi-informasi tertulis seperti misalnya nama, alamat, nilai numerikdan lain sebagainya.

5. *Combo Box*

Kotak combo (C*omboBox*) adalah kotak pilihan yang oleh pengguna (*user*) dapat diisikan nilai baru pada kotak combo, sebagian besar kotak combo merupakan daftar terbuka (drop-down list) yang baru akan menampilkan daftar nilainya apabila di klik, tombol panah bawah yang tedapat disebelah kanan daftar tersebut.

6. *Picture*

Picture berfungsi menampilkan file gambar pada pembuatan sebuah aplikasi seperti misalnya untuk tujuan hiasan, logo, perlengkapan data dan sebagainya. file-file gambar yang dapat ditampilkan picture adalah file-file yang mempunyai format BMP,CUR, EMF, GIF, ICO, JPG, WMF.

7. Data *Control*

Data control dapat digunakan untuk menampilkan record-record database pada sebuah program aplikasi yang dibuat. Data control menyediakan tombol penggulung yang dapat membantu pengguna melewati record demi record di dalam sebuah file database. Data control biasa menampilkan atau memunculkan database yang mempunyai format file selain format file VB.

Misalnya access (mdb), dBase (dbi), FoxPro(dbf), Excel (xls), lotus (wk*), paradox (db) dan database berbasis teks (txt)

8. *Frame*

   Frame berupa bingkai segi empat yang berfungsi mengakomodasi (menjadi tempat meletakkan) kontrol-kontrol yang lain. semua kontrol-kontrol yang diletakkan di dalam *frame* ini dianggap sebagai satu kelompok.

9. Menu Editor

   Menu editor adalah kontrol yang membuat serangkaian pilihan-pilihan yang dapat dipilih atau diklik untuk melakukan tugas tertentu. menu ini biasanya terdapat di bagian atas suatu program aplikasi yang dibuat.

10. Status Bar

    Status bar adalah baris satus yang biasanya terdapat dibawah program aplikasi yang dibuat yang berfungsi memberikan keterangan. pada status bar akan muncul teks yang menyatakan keterangan yang dapat berisikan tanggal, jam, tombol-tombol keyboard yang ditekan (tombol *Caps, Num, Insert*) dan lain sebagainya.

11. DBCombo

    DBCombo adalah kontrol data bound dengan sebuah daftar drop-down yang secara otomatis diisikan dengan suatu field yang diambil dari suatu database. DBCombo mendukung untuk pencarian otomatis untuk mendapatkan item dari daftar tanpa menambahkan sedikitpun kode pencarian.

12. DBGrid

    DBGrid adalah objek kontrol data bound, yang dapat menampilkan field-field sebuah database ke dalam bentuk tabel. Pada DBGrid pengguna dapat

membaca dan juga dapat menuliskan teks pada kolom atau baris tabel. DBGrid ini enghubungkan tampilan dari tempat penyimpanannya, sehingga isi dari tabel tersebut secara otomatis dapat di up-date.

2.9.1.2 Visual Data Manager

Visual data manager adalah sebuah program aplikasi pembuat database yang terdapat pada VB. Visual data manager dapat membuat data relasional (database yang datanya direpresentasikan dalam tabel yang terbentuk dari baris-baris dan kolom-kolom) dari aplikasi-aplikasi pembuat program aplikasi microsoft lainnya. diantaranya adalah melalui aplikasi Ms. Access, dBase, FoxPro dan Paradox. Sehingga apabila pengguna (user) sudah mempunyai sebuah file database berformat access atau dbase. Maka file tersebut dapat digunakan sebagai file database dalam VB.

2.9.2 SQL Server 2000

Menurut sumber Martina Inge MAR[5] bahwa *"Microsoft SQL server 2000 adalah perangkat lunak Relation Database Management Sistem (RDBMS) yang handal. Di design untuk mendukung proses transaksi yang besar seperti order entry online, inventori, akuntansi atau manufaktur. Microsoft SQL Server 2000 dapat dijalankan pada NT 4.0 Server atau Microsoft Windows 2000 Server, selain itu dapat pula di install pada personal desktop di Windows 2000 Profesional, Windows 98 Milenium"*.

Sistem akan dipergunakan oleh beberapa komputer (*client-server*) maka database yang dibangun adalah merupakan database yang berfungsi untuk

menunjang hal tersebut untuk itu dalam penulisan ini dipakai database Microsoft SQL Server 2000, selain itu database ini juga compatible dengan bahasa pemrograman yang digunakan. Secara teoritis, program SQL yang sudah terinstall pada komputer dapat menampung 32.767 database dan terdapat lebih dari 2 billion objek.

Kelebihan Microsoft SQL Server 2000 dalam pembuatan database adalah sebagai berikut:

1. Mempunyai transaction log tersendiri dan mengatur transaksi dalam database.
2. Data dapat berkisar antara 1 MB sampai 1.048.516 TB.
3. Dapat menambah ukuran data secara manual atau otomatis.
4. Dapat diset sesuai dengan keinginan, misal sebuah database hanya dapat di baca tetapi tidak bisa diedit.

2.10 Kasus yang Dianalisis/Dirancang

Untuk kasus yang dianalisis Penulis tertarik untuk mengambil masalah penjualan dan pembelian di PD Anugrah Grosir, karena proses pencatatan transaksi penjualan dan pembelian tersebut banyak yang masih dilakukan secara manual dimana tingkat kesalahan manusia masih besar. Untuk itu Penulis mengambil kasus penjualan dan pembelian di PD Anugrah Grosir Karawang agar semua masalah yang menyangkut penjualan dan pembelian di PD Anugrah Grosir Karawang dapat diatasi.