BAB II

LANDASAN TEORI

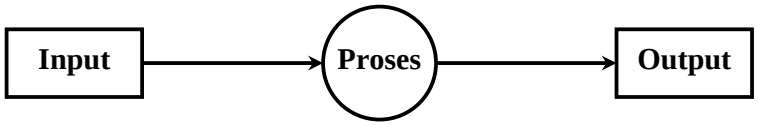
2.1 Pengertian Sistem

Sistem dapat diartikan dengan adanya suatu bagian-bagian yang saling terkait dan melakukan kerjasama untuk mencapai suatu tujuan. Keterkaitan antar komponen tersebut sangat diperlukan, karena tanpa adanya kerjasama yang baik antar komponen pendukung sistem, maka tidak akan dapat mencapai hasil atau tujuan yang maksimal. Pengertian sistem akan dijelaskan lebih lanjut pada pembahasan selanjutnya.

Menurut Jogiyanto bahwa "*Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu"* [JOG : 6]. Sedangkan menurut Abdul Kadir bahwa "*Sistem adalah sekumpulan elemen yang saling berhubungan yang dimaksudkan untuk mencapai suatu tujuan*" [ABD : 2]. Sehingga sistem dapat diartikan sebagai kumpulan komponen-komponen yang saling bekerjasama untuk mencapai suatu tujuan tertentu.

2.1.1 Bentuk Umum Sistem

Bentuk umum dari suatu sistem adalah terdiri dari Masukan (*Input*), Proses dan keluaran (*Output*). Dalam bentuk umum sistem ini dapat memasukan satu atau lebih masukan yang selanjutnya akan diproses dan menghasilkan suatu keluaran. Bentuk umum sistem dapat dilihat pada gambar 2.1 dibawah ini.



Gambar 2.1 Bentuk Umum Sistem

[Sumber : Hartono, JOG : 6]

2.1.2 Karakteristik Sistem

Untuk mencapai tujuannya, suatu sistem harus memiliki sifat-sifat tertentu atau suatu karakteristik seperti berikut :

1. Komponen (*Components*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi dan bekerja sama membentuk satu kesatuan.

2. Batas Sistem (*Boundary*)

   Merupakan daerah yang membatasi antara satu sistem dengan sistem lainnya atau dengan lingkungan luarnya.

3. Lingkungan Luar Sistem (*Environments*)

   Adalah segala sesuatu yang berada diluar batas sistem yang mempengaruhi operasi sistem baik itu yang bersifat merugikan ataupun menguntungkan.

4. Penghubung (*Interface*)

   Merupakan media penghubung antar subsitem yang memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lain.

5. Masukan (*Input*)

   Adalah energi yang dimasukan kedalam sistem, yang dapat berupa masukan perawatan (*Maintenance Input*) dan masukan sinyal (*Signal Input*).

6. Keluaran (*Output*)

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

7. Pengolah (*Process*)

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

8. Sasaran (*Objectives*) atau Tujuan (*Goal*)

   Suatu sistem pasti mempunyai tujuan (*goal*), jika suatu sistem tidak mempunyai tujuan yang jelas, maka semua operasi sistem tidak ada gunanya.

2.1.3 Model Sistem

Model sistem merupakan penyederhanaan dari sesuatu. Suatu model yang dibentuk akan mewakili sebuah objek atau entitas. Terdapat beberapa jenis model sistem, yaitu :

a. Model Fisik, yaitu penggambaran entitas atau objek dalam bentuk tiga dimensi
b. Model Naratif, yaitu penggambaran entitas atau objek dalam bentuk lisan atau tulisan
c. Model Grafik, yaitu penggambaran entitas atau objek dalam bentuk simbol dan garis
d. Model Matematika, yaitu penggambaran entitas atau objek dalam bentuk persamaan atau formula matematika.

2.2 Pengertian Informasi

Informasi sangat erat kaitannya dengan data, karena informasi dihasilkan dari data-data. Oleh karena itu, sebelum menjelaskan pengertian informasi, terlebih dahulu akan dijelaskan pengertian dari data.

Menurut Gordon B. Davis *"data adalah bahan mentah bagi informasi, dirumuskan sebagai kelompok lambang tidak acak yang menunjukan jumlah, tindakan, hal-hal dan sebagainya. Data-data dibentuk dari lambang grafis yang bisa berbentuk alfabetis, numeric atau berupa lambang. Data disusun untuk mengelola tujuan-tujuan menjadi susunan data, susunan kearsipan dan pusat data atau landasan data"* [ABD : 2]. Jadi dapat diambil kesimpulan, bahwa data adalah merupakan suatu bahan mentah yang bisa berbentuk lambang garfis, angka, atau simbol yang belum memiliki arti bagi pengguna.

Masih menurut Gordon B. Davis bahwa "*Informasi adalah data yang telah diolah menjadi suatu bentuk yang penting bagi si penerima dan mempunyai nilai yang nyata atau dapat dirasakan dalam keputusan-keputusan yang sekarang atau yang akan datang*" [ABD : 2]. Sedangkan menurut Jogiyanto *"Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya"* [JOG : 6]. Jadi informasi dengan kata lain adalah hasil dari suatu proses pengolahan atau manipulasi data yang belum mempunyai arti menjadi lebih mempunyai arti bagi pengguna.

2.2.1 Kualitas Informasi

Kualitas dari suatu informasi (*quality of information*) tergantung dari tiga hal, yaitu :

1. Akurat, yang berarti informasi tersebut harus terbebas dari kesalahan-kesalahan. Bisa juga berarti bahwa informasi harus jelas dalam mencerminkan maksudnya.
2. Tepat pada waktunya, suatu informasi tidak boleh terlambat, juka informasi tersebut sampai terlambat, maka nilai informasi tersebut tidak akan berarti lagi karena informasi merupakan landasan yang dipakai dalam mengambil keputusan.
3. Relevan, berarti suatu informasi mempunyai manfaat apabila diberikan kepada orang yang membutuhkan informasi tersebut.

2.3 Pengertian Sistem Informasi

Sistem informasi pada suatu sistem meliputi pemasukan data, yang selanjutnya diproses atau diolah untuk mendapatkan suatu hasil atau keluaran. Hasil informasi tersebut akan diambil kembali sebagai input untuk proses selanjutnya, sehingga akan membentuk suatu siklus informasi.

Sistem informasi didefinisikan oleh Robet A. Leitch dan K. Roscoe Davis "*Sistem Informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan"* [JOG : 6]. Sistem informasi berguna untuk mempercepat sampainya informasi ke tangan pengguna.

2.3.1 Komponen Sistem Informasi

Komponen sistem informasi terdiri dari [JOG : 6] :

1. Blok Masukan

   Meliputi metode-metode dan media untuk menangkap data yang akan dimasukan yang dapat berbentuk dokumen.

2. Blok Teknologi

   Digunakan untuk menerima input, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran serta membantu pengendalian sistem secara keseluruhan.

3. Blok Keluaran

   Berupa data keluaran yang berisi informasi yang berkualitas

4. Blok Basis Data

   Merupakan kumpulan data yang berhubungan satu sama lain, tersimpan di perangkat keras komputer dan perangkat lunak untuk memanipulasinya

5. Blok Kendali

   Meliputi masalah pengendalian terhadap masalah operasional yang berfungsi untuk mencegah dan menangani kesalahan atau kegagalan sistem.

2.4 Konsep Sistem Informasi Pengelolaan BOS

Menurut Buku Panduan BOS, dapat dijelaskan bahwa "*Pengelolaan adalah suatu cara atau metode untuk mengatur dan menggunakan suatu sumber daya sesuai dengan ketentuan atau peraturan yang berlaku*"[DIR : 9]. Tujuan dari kegiatan pengelolaan tesebut adalah untuk mengatur segala kegiatan yang ada agar lebih teratur dan terstruktur guna mencapai suatu tujuan.

Sedangkan pengertian dari BOS *"BOS adalah dana bantuan dari pemerintah untuk membantu membebaskan biaya bagi siswa miskin dan meringankan untuk siswa yang mampu, juga sebagai bantuan untuk membantu membiayai kegiatan operasional yang dikeluarkan oleh pihak sekolah"*[DIR : 9]. Bos diberikan oleh pemerintah sebagai pengganti biaya SPP perbulan untuk anak-anak kurang mampu, yang tujuannya agar pendidikan dapat dirasakan oleh semua kalangan.

Sistem Informasi Pengelolaan Dana Bantuan Operasional Sekolah (BOS) ini bertujuan agar informasi yang dibutuhkan oleh Kepala Sekolah ataupun yang lainnya bisa diperoleh dengan cepat, tepat dan akurat. Selain itu sistem ini bertujuan agar pengolahan data bisa berjalan lebih efektif dan efisien.

2.5 Analisis Sistem

Tahap analisis dilakukan sebelum tahap perancangan sistem, untuk mengetahui kebutuhan-kebutuhan yang ada. Tahap analisis merupakan tahap yang kritis, sangat penting dan sangat menentukan. Apabila dalam tahap ini terdapat kesalahan, maka akan menyebabkan kesalahan-kesalahan lain pada tahap berikutnya.

Langkah-langkah dasar dalam menganalisis sistem adalah sebagai berikut :

a. *Identify,* yaitu mengidentifikasi masalah
b. *Understand,* yaitu mengerti atau memahami cara kerja sistem yang sudah ada
c. *Analyze,* yaitu menganalisis sistem yang sudah ada
d. *Report,* yaitu membuat laporan hasil analisis

2.6 Pengenalan *Object Oriented*

Object orinted adalah suatu metode pendekatan sistem yang dalam pelaksanaannya lebih menekankan pada objek yang ada. Dibawah ini akan dijelaskan tentang metode dalam *object oriented*.

2.6.1 *Object Oriented Analiysis* (OOA)

Analisis mempunyai beberapa definisi, yaitu :

a. Pemecahan, pembagian (*dekomposisi*) dari sesuatu yang bersifat menyeluruh/global (*whole*) menjadi bentuk bagian (*parts*).
b. Memisahkan atau memecahkan sejumlah *whole* ke dalam *parts* sehingga dapat menemukan habitatnya, proporsi, fungsi dan relasi.
c. Kegiatan yang mempelajari suatu domain permasalahan, menemukan spesifikasi dan tingkah laku eksternal yang diamati secara lengkap, konsisten terhadap pernyataan yang diperlukan.

Dalam *Object oriented Analiysis* (OOA) terdapat lima lapisan kegiatan (tidak bersifat sequensial), yaitu :

1. Menentukan lapisan subjek.
2. Menentukan lapisan kelas dan objek
3. Menentukan lapisan struktur.
4. Menentukan lapisan *attribute.*
5. Menentukan lapisan *service.*

2.6.2 *Object Oriented Desigen* (OOD)

Desain mempunyai beberapa definisi, yaitu :

a. Menandai, membuat rencana besar. Sketsa, pola.
b. Kegiatan untuk mengambil dan menentukan spesifikasi dari prilaku luar yang dapat diamati dan menambah detail yang diperlukan untuk implementasi sistem komputer (secara nyata), termasuk interaksi manusia, pengolahan tugas/aksi, pengolahan data secara detail.

Dalam *object oriented design* (OOD), terdapat lima lapisan kegiatan (tidak bersifat sequensial), yaitu :

1. Menentukan lapisan subjek
2. Menentukan lapisan kelas dan objek.
3. Menentukan lapisan struktur.
4. Menentukan lapisan *attribute*.
5. Menentukan lapisan *service.*

2.7 Pengertian *Unified Modeling Language* (UML)

UML adalah suatu bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan *artifacs* (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, *artifacs* tersebut dapat berupa model deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.

UML juga merupakan bahasa pemodelan yang menggunakan konsep orientasi objek. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya, dibuat oleh Grady Booch OOD (*Object Oriented Design*), James

Rambaugh OMT (*Object Modeling Technique*) dan Ivar Jacobson OOSE (*Object Oriented Software Engineering*) dibawah perusahaan *software Rational Software Corp*. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai persepektif.

2.7.1 Bagian-Bagian UML

UML dapat dibagi menjadi dua kategori yaitu *view* dan *diagram.* Berikut penjelasan dari kedua kategori tesebut.

2.7.1.1 *View*

View digunakan untuk melihat sistem yang akan dimodelkan dari beberapa aspek berbeda. *View* bukan merupakan grafik, namun merupakan sebuah abstraksi yang berisi sejumlah diagram.

Beberapa jenis *View* dalam UML diantaranya : *use case view, logical view, concurrency view, component view* dan *deployment view.*

1. Use case View

View ini mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai dengan yang diinginkan *Actor*. Digambarkan dalam *use case diagram* dan kadang-kadang dengan *activity diagram.*

2. Logical View

View ini digambarkan dalam *class diagram* untuk struktur statis dan dalam *state, sequence, collaboration* dan *activity diagram* untuk model dinamisnya.

3. Component View

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lain dari *code module* diperlihatkan dengan struktur dan ketergantungan juga alokasi sumberdaya komponen dan informasi lainnya.

4. Concurrency View

Membagi sistem kedalam proses dan prosesor. *View* ini digambarkan dalam diagram dinamis (*state, sequence, collaboration* dan *activity diagram*) dan diagram implementasi (*component depelopment diagram*).

5. Deployment View

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat keras serta hubungannya dengan komponen lain.

2.7.1.2 Diagram

Diagram dalam bentuk grafik berfungsi untuk menunjukan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Diagram biasanya merupakan bagian dari sebuah v*iew* tertentu dan dialokasikan untuk view tertentu pula. Adapun jenis diagram diantaranya :

1. 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 dalam bentuk teks sebagai bentuk dokumentasi dari *use case symbol* namun dapat juga dilakukan dalam *activity diagram*.

2. Class Diagram

Menggambarkan struktur statis *class* didalam sistem. *Class* mempresentasikan sesuatu yang ditangani oleh sistem.

2.1 Class

Class adalah deskripsi sekelompok *object* dari property (atribut), sifat (operasi), relasi antar *object* dan sematik yang umum[RIY : 3]. *Class* merupakan *template* untuk membentuk *object*. Setiap *object* merupakan contoh dari beberapa *class* dan *object* tidak dapat menjadi contoh lebih dari satu *class*.

3. State Diagram

Menggambarkan semua kondisi yang dimiliki oleh suatu objek dari suatu c*lass* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa objek lain yang mengirim pesan.

4. Sequence Diagram

Menggambarkan kolaborasi yang dinamis antara sejumlah objek. Kegunaannya adalah untuk menunjukan rangkaian pesan yang dikirim antara objek juga interaksi antara objek, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

5. Collaboration Diagram

Collaboration Diagram menggambarkan objek dan hubungannya (mengacu pada konteks). Jika penekanannya mengacu pada waktu atau urutan, maka gunakan *sequence diagram,* tapi jika penekanannya mengacu pada konteks, maka gunakan *collaboration diagrams.*

6. Activity Diagram

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

7. Component Diagram

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

8. Deployment Diagram

Menggambarkan arsitektur fisik dari perangkat keras dan perangkat lunak sistem, menunjukan hubungan komputer dengan perangkat satu sama lain dan jenis hubungannya.

2.7.2 Gambaran UML

1. UML sebagai bahasa pemodelan

UML merupakan bahasa pemodelan yang memiliki perbendaharaan kata dan cara untuk mempresentasikan secara fokus pada konseptual dan fisik dari suatu sistem. Dengan UML kita akan menjadi tahu bagaimana cara untuk membuat dan membaca bentuk model yang baik.

2. UML sebagai bahasa untuk melakukan spesifikasi sistem

UML menunjukan semua spesifikasi keputusan analisis, desain dan implementasi yang penting yang harus dibuat pada saat pengembangan dan penyebaran dari sistem.

3. UML sebagai bahasa untuk melakukan penggambaran sistem

UML, menggambarkan model yang dapat dimengerti dan dapat dengan mudah dipresentasikan kedalam modul tekstual bahasa pemrograman. UML merupakan suatu model eksplisit yang menggambarkan komunikasi informasi pada sistem. Sehingga kita tidak akan kehilangan informasi kode implementasi yang hilang dikarenakan *developer* memotong *coding* dari implementasi.

4. UML sebagai bahasa untuk membangun sistem

Bahasa UML dapat dihubungkan secara langsung kedalam bahasa pemrograman visual. Atau dengan kata lain UML dapat membangun model yang dapat dimapping ke bahasa pemrograman seperti VB, Delphi, Java, C++ dan pada table *database relational* serta penyimpanan tetap pada *database* berorientasi objek.

5. UML sebagai bahasa untuk pendokumentasian sistem

UML dapat menunjukan dokumentasi dari arsitektur sistem dan detail dari keseluruhannya. UML memberikan bahasa untuk memperlihatkan permintaan dan untuk pengetesan. UML juga menyediakan bahasa untuk memodelkan aktifitas dari perencanaan *project* dan *release management.*

2.8 Pengertian Database

Menurut Jogiyanto, *"Database adalah kumpulan file-file yang saling berelasi, relasi tersebut biasa ditunjukan dengan adanya kunci pada tiap file yang ada"* [JOG : 6]. *Database* juga dapat diartikan sebagai kumpulan dari beberapa file yang sejenis. Sedangkan dalam *Database Management system*, *Database*

berarti “sekumpulan data yang saling berhubungan dan berkaitan satu dengan yang lain dan digunakan oleh suatu organisasi”.

Berikut ini terdapat beberapa istilah-istilah yang merupakan bagian dari *Database* adalah sebagai berikut :

1. *Entity*, adalah orang, tempat, kejadian atau konsep yang informasinya direkam
2. *Atribute*, sebagai elemen, data field, data item untuk mewakili suatu *entity*
3. *Data Value*, adalah data actual atau informasi yang disimpan pada tiap data elemen atau *attribute*
4. *Record*, ialah kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu *entity* secara lengkap.

2.8.1 Jenis Database

Adapun jenis database terdiri dari tiga jenis, yaitu :

1. Database Hirarki

Sistem manajemen database hirarki menyediakan data yang dapat ditampilkan dengan struktur pohon, misalnya aplikasi dimana orang tua boleh mempunyai banyak anak tetapi seorang anak hanya mempunyai satu orang tua.

2. Database Jaringan

Sistem manajemen database jaringan dapat mempersiapkan struktur jaringan, misalnya dimana seorang mahasiswa boleh memilih banyak mata kuliah.

3. Database Relasional

Sistem manajemen database relasional lebih fleksibel dibandingkan dengan kedua DBMS diatas, karena database relasional mudah didesain dan digunakan. Kebanyakan suatu DBMS relasional tidak mensyaratkan pendisain

untuk menspesifikasikan bentuk hubungan-hubungan ketika database mulai dibuat. Hubungan-hubungan tersebut baru perlu dispesifikasikan bila database digunakan untuk mendapatkan informasi atau untuk keperluan membuat laporan.

2.9 Sistem Cleint-Server

Sistem yang ada pada *Client* berfungsi untuk menjalankan aplikasi basis data, dimana *client* akan menangani setiap aktivitas yang dikehendaki para pemakai dan mengusahakan agar semua proses dapat ditangani sendiri, jika ada proses yang harus melibatkan data yang tersimpan pada basis data, barulah *client* melakukan kontak dengan *server*. *Server* merupakan terminal yang menjalankan DBMS dan berisi basis data sebagi tempat *user* berinteraksi dengan sistem.

Dalam sebuah sistem terdapat hubungan antara *node-node* yang saling dihubungkan biasanya disebut sebagai arsiktektur jaringan atau topologi jaringan, ada tiga jenis topologi yang biasa digunakan, yaitu:

1. Topologi *Bus*

Ethernet menggunakan satu kabel yang berfungsi sebagai medium untuk mentranmisikan data. *Node* yang merupakan bagian dari jaringan dihubungkan seluruhnya kabel tersebut. *Node* yang terhubung mengirim dan menerima data jaringan melalui kabel sebagai pembawa sinyal dan melihat apakah data tersebut ditujukan buat dirinya.

2. Topologi *Ring*

Salah satu *node* yang ada dihubungkan dengan *node* yang ada didepan dan dibelakangnya sehingga membentuk *ring*. Setiap mendapatkan giliran menggunakan jaringan dengan mengirim *token*. *Node* yang mendapat giliran

mengirim data dan *node* lain mendapat giliran untuk menerima data, serta melihat apakah data ditujukan kepadanya atau bukan.

3. Topologi *Star* atau *Hub*

Susunan dari topologi ini mirip dengan sebuah bintang. Topologi ini memiliki satu *hub* pusat dimana data ditransmisikan keseluruh *node* dalam jaringan. Skema ini memiliki kelebihan dibandingkan dengan topologi lainnya, yaitu bila terjadi kerusakan pada kabel tidak membawa dampak bagi *node* yang lain, hanya *node* yang bersangkutan saja yang rusak sehingga aktifitas jaringan yang lain tidak terganggu.

4. Topologi *Point to Point*

Topologi ini memiliki bentuk yaitu *node* yang satu terhubung secara langsung dengan *node* lain oleh alat transmisi yaitu kabel. topologi ini biasa digunakan apabila jumlah *node* hanya dua.

2.10 Gambaran *Software* Yang Digunakan

Software merupakan suatu istilah umum yang menyatakan segala jenis perangkat lunak seperti program pada komputer. *Software* dapat membantu pengguna untuk mengolah data sehingga menjadi sebuah informasi yang dibutuhkan oleh pengguna.

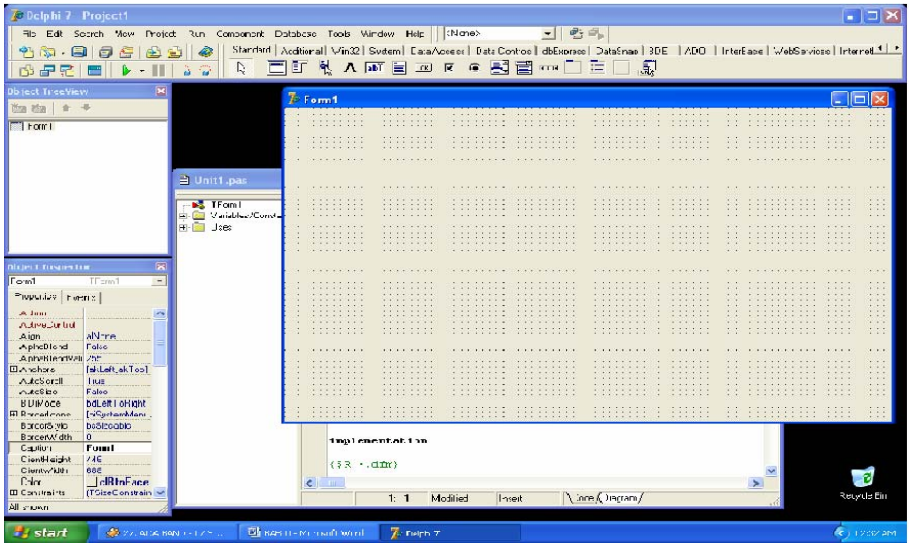
2.10.1 Sejarah Borland Delphi

Borland Delphi dirilis pertama kali pada bulan Februari 1995 dan telah mencapai perkembangannya sampai dengan Delphi veri 7.0. Delphi adalah bahasa pemrograman visual yang merupakan kelanjutan dari Turbo Pascal. Delphi adalah

produk dari *Borland International* yang berkedudukan di *Scotts Valley, California.* Setelah bergabung dengan *Visigenic*, *Borland International* berganti nama menjadi *Inprise Corporation* pada pertengahan tahun 1998.

2.10.2 Bagian-bagian dari Aplikasi Borland Delphi

Pada dasarnya IDE (*Integrated Development Environment*) milik Delphi dibagi menjadi lima bagian utama, yaitu Jendela Utama, *Object Treeview, Object Inspector, Form Designer* dan *Code Editor*. Tampilan awal Delphi dapat dilihat pada gambar 2.2 dibawah ini.



Gambar 2.2 Tampilan awal Delphi 7

Gambar diatas adalah merupakan tampilan awal dari Borland Delphi 7.0 yang juga merupakan IDE atau lingkungan pengembangan Delphi.

1. Jendela Utama

Pada jendela utama terdapat menu-menu seperti pada aplikasi windows pada umumnya, *toolbar* merupakan langkah cepat dari beberapa menu,

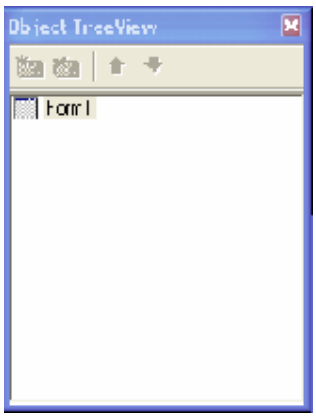
Component Pallete yaitu tempat komponen-komponen yang akan digunakan membuat aplikasi. Berikut ini adalah gambar 2.3 jendela utama Delphi :



Gambar 2.3 Jendela Utama

2. *Object Treeview*

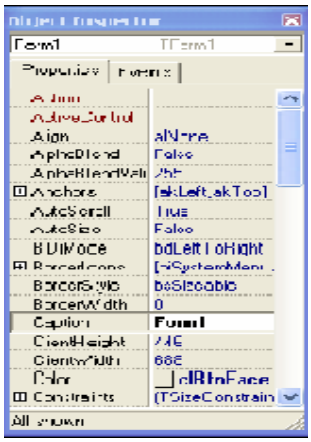
Object ini baru dimiliki Delphi mulai dari Delphi versi 6 dan 7, sehingga Delphi 5 ke bawah belum memiliki objek ini. *Object Treeview* ini berfungsi untuk menampilkan daftar komponen yang digunakan dalam pengembangan aplikasi. Berikut ini adalah gambar 2.4 *Object Treeview* :



Gambar 2.4 *Object Treeview*

3. *Object Inspector*

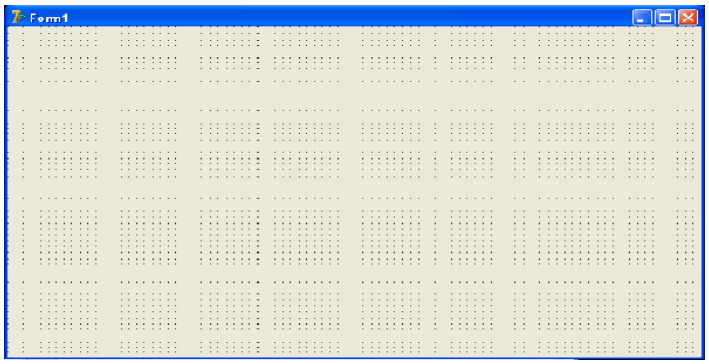
Object Inspector digunakan untuk mengatur *property* atau *even* suatu komponen. Berikut adalah gambar 2.5 *Object Inspector* :



Gambar 2.5 *Object Inspecto*

4. *Form Designer*

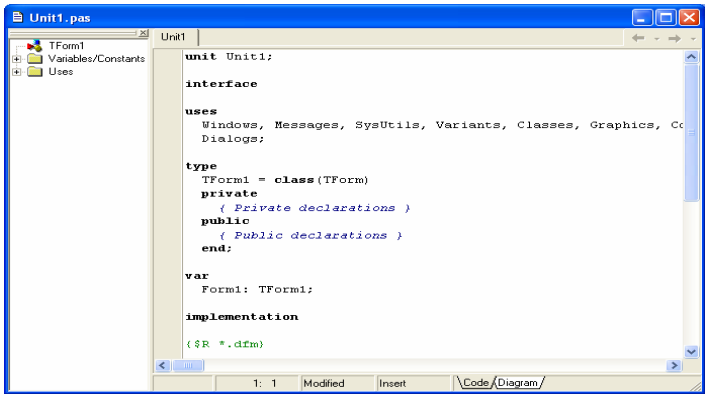
Form designer adalah tempat untuk meletakan komponen-komponen lain. Berikut adalah gambar 2.6 *Form designer* :



Gambar 2.6 *Form Designer*

5. *Code Editor*

Code editor adalah tempat untuk menuliskan kode program yang diperlukan untuk mengatur aplikasi yang sedang dibuat. Berikut ini merupakan gambar 2.7 *code editor* :



Gambar 2.7 *Code Editor*

2.11 Sekilas Tentang Database Desktop (Paradox)

Delphi menyediakan fasilitas yang lengkap untuk mengolah database. Diantaranya dengan menggunakan suatu software yang bernama Database Desktop (DBD). Database Desktop adalah suatu program yang dikeluarkan oleh Borland sebelum delphi. Versi terakhir dari database desktop adalah versi 7.0 yang hadir sejak Delphi versi 2. format database yang biasa atau lebih disukai oleh *programmer* adalah paradox.

Database desktop adalah suatu program *Add-Ins*, yaitu program terpisah yang ditempelkan dalam delphi, sehingga bisa dipakai sebagai bagian dari program delphi. Dalam membuat file database dengan menggunakan paradox, kita hanya perlu membuatnya satu kali dengan tidak perlu mengisi record-recordnya (hanya struktur database nya saja). Untuk selanjutnya file database tersebut diproses lewat program.