BAB II
TINJAUAN PUSTAKA

II.1 Profil Instansi

Terminal kaos merupakan perusahaan yang bergerak di bidang konveksi baju, melayani berbagai macam pesanan konveksi dengan kualitas tinggi, harga yang terjangkau, dan sesuai dengan permintaan pelanggan. Mulai dari kaos, polo shirt, kemeja, jaket, sweater rajut, topi, dan produk konveksi lainnya. Pelanggan yang memesan ke terminal kaos berasal dari berbagai wilayah di seluruh Indonesia dan terdiri dari berbagai komunitas, lembaga pendidikan, partai politik, dan perusahaan. Terminal kaos berlokasi di jalan Gagak No.143B Bandung.

II.1.1 Sejarah Instansi

Terminal kaos berdiri pada Juni 2009. Saat itu terminal kaos baru bisa melakukan proses produksi potong dan sablon saja, sedangkan untuk proses jahit masih bekerjasama dengan pihak lain. Mulai November 2009, terminal kaos mendapatkan suntikan dana dari Program Mahasiswa Wirausaha (PMW) 2009 sebesar 37 juta rupiah. Dana tersebut digunakan untuk membeli mesin produksi jahit, sehingga mulai awal desember 2009, terminal kaos sudah mampu melakukan produksi jahit sendiri. Januari 2010 terminal kaos mengajukan pendirian CV dan akhirnya pada akhir Januari terminal kaos telah resmi berbadan hukum atas nama CV SMART CLOTHING INDONESIA. Sejak memiliki 100% mesin produksi sendiri, jumlah pesanan yang mampu dikerjakan semakin meningkat.

II.1.2 Visi dan Misi Instansi

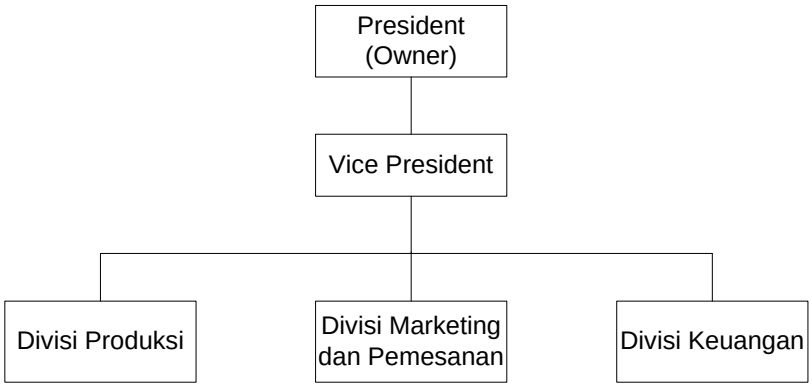
Visi dari terminal kaos sebagai pabrik konveksi adalah menyediakan jasa pembuatan produk konveksi sesuai permintaan dari konsumen.

Adapun misi dari terminal kaos yaitu:

1. Selalu menghasilkan produk yang berkualitas sesuai pesanan konsumen.
2. Mengerjakan produk konveksi dengan harga yang murah dan bersahabat.

II.1.3 Struktur Organisasi Instansi

Struktur organisasi adalah pola hubungan antara bagian-bagian dari instansi atau menggambarkan dengan jelas pemisahan kegiatan pekerjaan antara bagian yang satu dengan bagian yang lain dalam suatu instansi. Gambar II.1 Berikut merupakan struktur organisasi yang ada di terminal kaos.



Gambar II.1 Struktur Organisasi Terminal Kaos

II.1.4 Deskripsi Kerja Struktur Organisasi Instansi

PRESIDENT (Owner) : Arif Rochman Permana

Vice President : Rahmad Agus Dwianto

Team kerja CV. TERMINAL KAOS beranggotakan 47 Orang Karyawan dan 3 Manager yaitu:

1. **Divisi Produksi**

   Manager Produksi : 1 Orang (Tata)

   Bagian Desain dan Setting: 1 Orang (Dian)

   Bagian Jahit : 20 penjahit senior

   Bagian Sablon : 4 ahli sablon

   Bagian Potong : 5 ahli potong

   Bagian Finishing : 5 Orang

   Bagian Kurir dan belanja : 1 Orang

2. Divisi Marketing dan Pemesanan

Manager marketing : 1 orang (Rahmad Agus D)

Marketing : Baroto Ihsan Wicaksono

Muhammad Wildan

Topan Eko Raharjo

Asep Manan

Ibrahim Al Hanif

Median Fratama

Suryadi

Didiek Agus

Bagian administrasi : 1 orang (Kania)

3. Divisi keuangan

Manager Keuangan : 1 orang (Arif Rochman Permana)

II.2 Landasan Teori

Sub bab ini berisi teori-teori pendukung yang digunakan dalam proses analisis dan implementasi dalam tugas akhir ini.

II.2.1 Konsep Basis Data

1. Data

Data adalah rekaman mengenai fenomena/fakta yang ada atau yang terjadi. Data pada pokoknya adalah refleksi fakta yang ada.

Menurut Whitten, et al (2004 : 23), data adalah fakta mentah mengenai orang, tempat, kejadian, dan hal-hal yang penting dalam organisasi.

Menurut Prahasta (2005 : 30), data merupakan bahasa, simbol-simbol pengganti lain yang disepakati oleh umum dalam menggambarkan objek, manusia, peristiwa, aktivitas, konsep. Singkatnya data merupakan suatu kenyataan apa adanya.

Menurut Jogianto (1990), data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Data merupakan bentuk jamak dari datum, berasal dari bahasa Latin yang berarti “sesuatu yang diberikan”. Dalam penggunaan sehari-hari data berarti suatu pernyataan yang diterima secara apa adanya.

2. Database

Menurut Toni Fabbri: *Database* adalah sebuah sistem *file-file* yang terintegrasi yang mempunyai minimal *primary key* untuk pengulangan data.

Menurut C.J. Date: *Database* adalah koleksi "data operasional" yang tersimpan dan dipakai oleh sistem aplikasi dari suatu organisasi.

a. Data *input* adalah data yang masuk dari luar sistem.
b. Data *output* adalah data yang dihasilkan sistem.
c. Data operasional adalah data yang tersimpan pada sistem

Menurut Gordon C. Everest: *Database* adalah koleksi atau kumpulan data yang mekanis, terbagi/*shared,* terdefinisi secara formal dan dikontrol terpusat pada organisasi.

Menurut S. Attre: *Database* adalah koleksi data-data yang saling berhubungan mengenai suatu organisasi/*enterprise* dengan macam-macam pemakaiannya.

II.2.2 *Data Warehouse*

1. Pengertian *Data Warehouse*

Beberapa pengertian dari beberapa ahli tentang definisi *data warehouse*:

a. Menurut W.H. Inmon dan Richard D.H., *data warehousing* adalah koleksi data yang mempunyai sifat berorientasi subjek, terintegrasi, *time-variant*, dan bersifat tetap dari koleksi data dalam mendukung proses pengambilan keputusan *management*.
b. Menurut Vidette Poe, *data warehousing* merupakan basis data yang bersifat analisis dan *read only* yang digunakan sebagai fondasi dari sistem penunjang keputusan.
c. Menurut Paul Lane, *data warehousing* merupakan basis data relasional yang di desain lebih kepada *query* dan analisa dari pada proses transaksi, biasanya mengandung *history* data dari proses transaksi dan bisa juga data dari sumber lainnya. *Data warehousing* memisahkan beban kerja analisis dari beban kerja transaksi dan memungkinkan organisasi menggabung/konsolidasi data dari berbagai macam sumber.

2. Karakteristik Sistem

Karakteristik *data warehouse* menurut Inmon yaitu :

a. *Subject Oriented* (Berorientasi Subjek)

Data warehouse berorientasi subjek artinya *data warehouse* di desain untuk menganalisa data berdasarkan subjek-subjek tertentu dalam organisasi, bukan pada proses atau fungsi aplikasi tertentu. *Data warehouse* diorganisasikan disekitar subjek-subjek utama dari perusahaan (*customers*, *products* dan *sales*) dan tidak diorganisasikan pada area aplikasi utama (*customer invoicing*, *stock control* dan *product sales*). Hal ini dikarenakan kebutuhan dari *data warehouse* untuk menyimpan data-data yang bersifat sebagai penunjang suatu keputusan, dari pada aplikasi yang berorientasi terhadap data. Jadi dengan kata lain, data yang disimpan adalah berorientasi kepada subjek bukan terhadap proses.

b. *Integrated* (Terintegrasi)

Data Warehouse dapat menyimpan data-data yang berasal dari sumber-sumber yang terpisah kedalam suatu format yang konsisten dan saling terintegrasi satu dengan lainnya. Dengan demikian data tidak bisa dipecah-pecah karena data yang ada merupakan suatu kesatuan yang menunjang keseluruhan konsep *data warehouse* itu sendiri. Syarat integrasi sumber data dapat dipenuhi dengan berbagai cara sepeti konsisten dalam penamaan *variable*, konsisten dalam ukuran *variable*, konsisten dalam struktur pengkodean dan konsisten dalam atribut fisik dari data.

c. *Time Variant* (Rentang Waktu)

Seluruh data pada *data warehouse* dapat dikatakan akurat atau valid pada rentang waktu tertentu. Untuk melihat interval waktu yang digunakan dalam mengukur keakuratan suatu *data warehouse*, kita dapat menggunakan cara antara lain :

1. Cara yang paling sederhana adalah menyajikan *data warehouse* pada rentang waktu tertentu, misalnya antara 5 sampai 10 tahun ke depan.

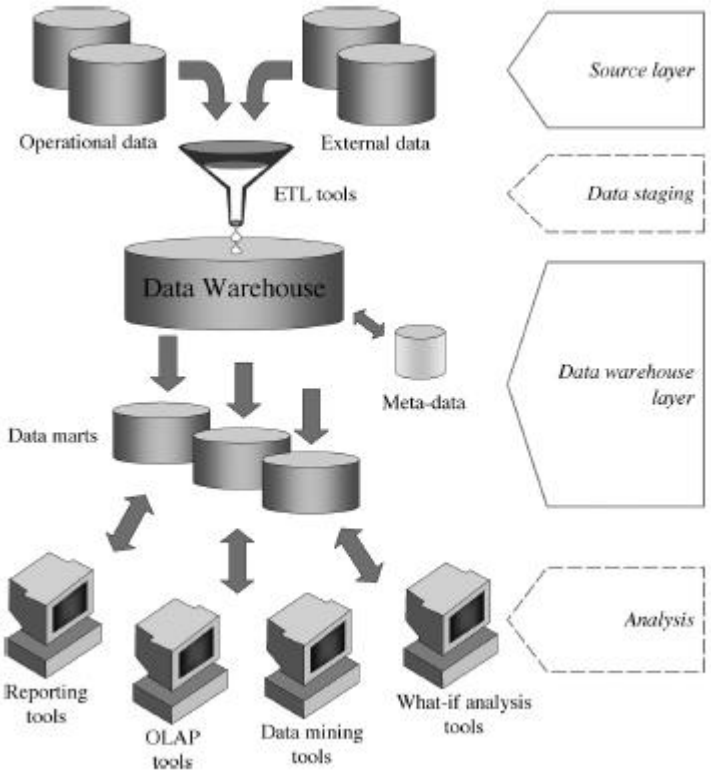
2. Cara yang kedua, dengan menggunakan variasi atau perbedaan waktu yang disajikan dalam *data warehouse* baik *implicit* maupun *explicit* secara *explicit* dengan unsur waktu dalam hari, minggu, dan bulan. Secara *implicit* misalnya pada saat data tersebut diduplikasi pada setiap akhir bulan, atau per tiga bulan. Unsur waktu akan tetap ada secara *implicit* didalam data tersebut.
3. Cara yang ketiga, variasi waktu yang disajikan *data warehouse* melalui serangkaian *snapshot* yang panjang. *Snapshot* merupakan tampilan dari sebagian data tertentu sesuai keinginan pemakai dari keseluruhan data yang ada bersifat *read-only*.

d. Non Volatile

Karakteristik keempat dari *data warehouse* adalah *non-volatile*, maksudnya data pada *data warehouse* tidak di-*update* secara *real time* tetapi di *refresh* dari sistem operasional secara reguler. Data yang baru selalu ditambahkan sebagai suplemen bagi *database* itu sendiri dari pada sebagai sebuah perubahan. *Database* tersebut secara *continue* menyerap data baru ini, kemudian secara *incremental* disatukan dengan data sebelumnya.

3. Arsitektur Sistem

Menurut Matteo Golfarelli dan Stefano Rizzi dalam bukunya yang berjudul *Data Warehouse Design: Modern Principles and Methodologies*, ada beberapa jenis arsitektur *data warehouse* yang dapat digunakan dalam pembangunan *data warehouse*. Salah satunya adalah *two layer architecture* seperti pada gambar II.2



Gambar II.2 *Two-Layer Architecture* [1]

Lapisan pertama adalah *source layer*. Pada lapisan ini, data masih berupa operasional data. Data operasional yang akan digunakan pada pembangunan *data warehouse* kali ini sudah berupa data *logic* yang ada di *database*.

Lapisan kedua adalah *data staging*. Pada lapisan ini, data operasional akan di-*ekstract* (proses *ETL*) ke dalam *data warehouse*.

Lapisan ketiga adalah *data warehouse layer*. Informasi akan disimpan pada sebuah penyimpanan *logic* yang tersentralisasi yaitu *data warehouse*. *Data warehouse* dapat diakses secara langsung.

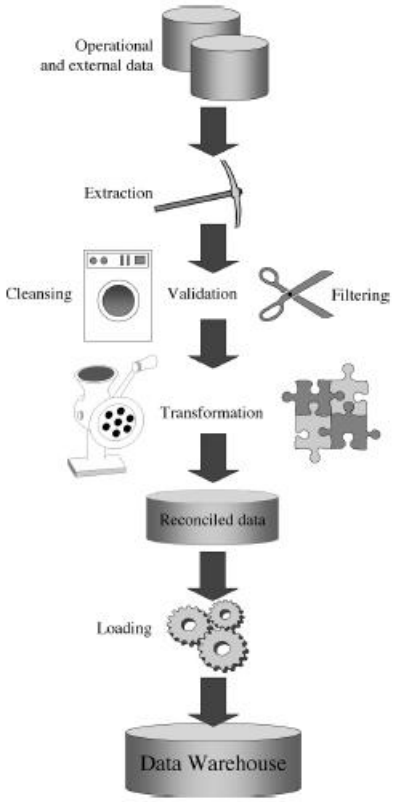
Lapisan keempat adalah *analysis*. Analisis disini nantinya akan menggunakan *OLAP* sebelum disajikan sebuah laporan khusus.

4. *Snowflake*

Snowflake adalah skema yang memiliki dimensi yang digunakan secara bersama-sama oleh banyak tabel fakta dan masih dipecah kembali. Skema ini dipilih karena pada dimensi barang dan dimensi toko, masih perlu didetailkan untuk mempermudah dalam proses *dice* pada *OLAP*.

5. *ETL* (*Extract, Transform, Loading)

Menurut Matteo Golfarelli dan Stefano Rizzi dalam bukunya yang berjudul *Data Warehouse Design: Modern Principles and Methodologies*, proses *ETL* adalah proses ekstraksi, integrasi, dan pembersihan data dari data operasional (*source layer*) sebelum kemudian dimasukkan ke dalam *data warehouse*. Seperti ditunjukkan pada gambar II.3 Proses *ETL*



Gambar II.3 Proses *ETL* [1]

Langkah pertama adalah *extraction*. Pada proses ini, data operasional yang diperlukan pada *data warehouse* akan dibaca untuk kemudian masuk dalam proses *cleaning*.

Langkah kedua adalah *cleaning*. Pada poses ini data operasional yang telah dibaca akan diperbaiki dari kesalahan-kesalahan pada proses *input* data. Misalnya terjadi redudansi dari data, *record* yang salah satu *field*-nya memiliki *null value* akan dibersihkan.

Langkah ketiga adalah *transform*. Pada proses ini data yang sudah dibersihkan diubah dari format data opersional menjadi format *data warehouse*.

Langkah ke empat adalah *loading*. Pada proses ini, data yang sudah dibaca, dibersihkan, dan dirubah formatnya akan disimpan pada *data warehouse*. Ada dua teknik dalam proses *loading* ini. Pertama adalah *refresh*, teknik ini akan mengganti data lama di *data warehouse* dengan data yang baru jika diperlukan. Teknik kedua adalah *update*, teknik ini tidak mengganti data yang sudah ada di *data warehouse*, tapi terus menambahkan data yang baru ke dalam *data warehouse*.

6. *Roll-up and Drill-down*

Drill-down and roll-up adalah operasi untuk melihat data global atau detail di sepanjang level hirarki dimensi. *Roll-up* untuk melihat data secara *global* atau rangkuman (*summary*). *Drill-down* memandu pengguna untuk memperoleh data yang lebih detail. *Drill-down* ini biasa digunakan untuk menjawab pertanyaan atas suatu kasus tertentu. Misalnya untuk menjawab pertanyaan ketika sebuah *summary* number (rata-rata atau jumlah) di bawah atau di atas harapan.

7. *Slice and Dice*

Slice and dice adalah operasi untuk melihat data sebagai visualisasi dari kubus. Dengan *slice* dan *dice* pengguna dapat melihat data dari beberapa perspektif. Pengguna dapat mengekstrak bagian dari *data agregrated* dan dapat memeriksa dengan detail berdasarkan dimensi-dimensi yang diinginkan. *Data Agregrated* merupakan data pra-perhitungan (*precalculated*) dalam bentuk rangkuman data (data *summarized*) sehingga *query* pada kubus (*cube*) lebih cepat. *Slice* memotong kubus sehingga dapat memfokuskan pada perspektif yang spesifik (pada suatu dimensi). Sedangkan *dice* memberikan kemampuan untuk melihat pemilihan data pada dua dimensi atau lebih, yaitu dengan merotasi *cube* pada perspektif yang lain sehingga pengguna dapat melihat lebih spesifik terhadap data yang di analisis.

8. *OLAP* (*On-Line Analytical Processing*)

OLAP (*On-Line Analytical Processing*) adalah salah satu cara untuk mengolah data yang ada pada sebuah *data warehouse*. *OLAP* memberikan jawaban terhadap *query analytic* untuk data yang bersifat multidimensional. *OLAP* menyediakan cara untuk menampilkan data multidimensional yang ada dalam *data mart* atau *data warehouse*, dengan *OLAP* dapat dibuat *cube* yang mengorganisasikan data dan membuat *summary* data untuk *query* yang efisien.

Teknik dalam *OLAP* ini ada berbagai macam. Dalam pengembangan kali ini, teknik yang akan digunakan adalah *roll-up and drill-down* dan *slice and dice*.

Karakteristik *OLAP*:

a. Menggunakan teknik analisis data multidimensional.
b. Menyediakan dukungan *database* tingkat lanjut.
c. Menyediakan cara pakai yang mudah dan *user interface* yang mudah dipahami.
d. Mendukung arsitektur *client-server*.

II.2.3 *Data Flow Diagram (DFD)*

DFD adalah suatu model logika data atau proses yang dibuat untuk menggambarkan darimana asal data dan kemana tujuan data yang keluar sistem, dimana data disimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang akan dikenakan pada data tersebut.

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan sacara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir (misalnya lewat telepon, surat dan sebagainya). Atau lingkungan fisik dimana data tersebut akan disimpan (misalnya file kartu, *hard disk*, *tape*, disket dan sebagainya).

DFD merupakan alat yang cukup populer saat ini, karena dapat menggambarkan arus data didalam sistem dengan terstruktur dan jelas. Lebih lanjut *DFD* merupakan dokumentasi dari sistem yang baik.

Beberapa simbol yang akan digunakan di dalam *DFD* antara lain menurut Jogianto adalah sebagai berikut :

1. Kesatuan luar (*External Entity*)

Setiap sistem pasti mempunyai batas sistem yang memisahkan suatu sistem dengan lingkungan luarnya. Sistem akan menerima *input* dan menghasilkan *output* kepada lingkungan luarnya. Kesatuan luar (*external entity*) merupakan kesatuan dilingkungan luar sistem dapat berupa orang, organisasi atau sistem lainnya yang berada dilingkungan luarnya yang akan memberikan *input* atau menerima *output* dari sistem. Kesatuan luar ini kebanyakan adalah salah satu dari berikut ini:

a. Suatu kantor, departemen atau divisi dalam perusahaan tetapi di luar sistem yang sedang dikembangkan.
b. Orang atau sekelompok orang di organisasi tetapi di luar sistem yang sedang dikembangkan.
c. Suatu organisasi atau orang di luar organisasi.
d. Sistem informasi yang lain di luar sistem yang sedang dikembangkan.
e. Sumber asli dari suatu transaksi.
f. Penerimaan akhir dari suatu laporan yang dihasilkan oleh sistem

2. Aliran Data (*Data Flow*)

Aliran data di *DFD* diberi simbol suatu panah. Aliran data ini mengalir diantara proses (*process*), simpan data (*data store*) dan kesatuan luar (*external entity*). Aliran data ini menunjukkan aliran dari data yang dapat berupa masukkan untuk sistem atau hasil dari proses sistem.

3. Proses

Proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses yang digambarkan secara umum. Suatu proses dapat ditunjukkan dengan simbol lingkaran atau dengan simbol empat persegi panjang tegak dengan sudut-sudutnya tumpul.

4. Berkas atau Simpanan Data (*Data Store*)

Berkas atau simpanan data merupakan simpanan dari data yang dapat berupa :

a. Suatu file atau *database* di sistem komputer.
b. Suatu arsip atau catatan manual.

II.2.4 *MySQL*

MySQL merupakan sebuah *software* yang berguna sebagai suatu *Database* Server yang cukup terkenal. *Database* server itu sendiri merupakan suatu *software* yang bertugas untuk melayani permintaan (*request*) *query* dari *client*.

MySQL sebagai suatu *database server* mempunyai beberapa kemampuan, salah satunya harus menyediakan suatu sistem manajemen *database* yang dapat mengatur bagaimana menyimpan, menambah, mengakses data dan transaksi-transaksi *database* lainnya. *MySQL* cepat sekali berkembang, karena *MySQL* merupakan suatu *software* yang bersifat *Open Source*.

II.2.5 *Borland Delphi*

Borland Delphi atau biasa yang disebut *Delphi* saja, merupakan sarana pemrograman aplikasi visual. Bahasa pemrograman yang digunakan adalah bahasa pemrograman *Pascal* atau kemudian juga yang disebut bahasa pemrograman *Delphi*. *Delphi* merupakan generasi penerus dari *Turbo Pascal*. *Turbo Pascal* yang diluncurkan pada tahun 1983 dirancang untuk dijalankan pada system operasi *DOS* (yang merupakan sistem operasi yang paling banyak digunakan pada saat itu). Sedangkan *Delphi* yang diluncurkan pertama kali tahun 1995 dirancang untuk beroperasi dibawah sistem operasi *windows*.

Delphi adalah *compiler* (penterjemah) bahasa *Delphi* (awalnya dari *pascal*) yang merupakan bahasa tingkat tinggi sekelas dengan *basic*, *C*. Bahasa pemrograman di *Delphi* disebut bahasa *procedural* yaitu bahasa atau *syntax*-nya mengikuti urutan tertentu. *Delphi* disebut juga *Visual Programming* artinya komponen-komponen yang ada tidak hanya berupa teks tetapi muncul berupa gambar-gambar.

Delphi memiliki sarana untuk pembuatan aplikasi, mulai dari sarana untuk pembuatan *form*, *menu*, *toolbar*, hingga kemampuan untuk menangani pengelolaan basis data yang besar. Kelebihan-kelebihan yang dimiliki *Delphi* antara lain karena pada *Delphi*, *form* dan komponen-komponennya dapat dipakai ulang dan dikembangkan, tersedia *template* aplikasi dan *template form*, memiliki lingkungan pengembangan visual yang dapat diatur sesuai kebutuhan,

menghasilkan file terkompilasi yang berjalan lebih cepat, serta kemampuan mengakses data dari bermacam-macam format.

Delphi menggunakan bahasa objek *pascal* didalam lingkungan pemrograman visual. Kombinasi ini menghasilkan sebuah lingkungan pengembangan aplikasi yang berorientasi objek (*Object Oriented Programming*). Dengan konsep seperti ini, maka pembuatan aplikasi menggunakan *Delphi* dapat dilakukan dengan cepat dan menghasilkan aplikasi yang tangguh. *Form* dan komponen yang ada didalamnya, misalnya, dapat disimpan dalam suatu paket komponen yang dapat digunakan kembali, atau dimodifikasi seperlunya saja.

Khususnya untuk pemrograman *database*, *Delphi* menyediakan *object* yang sangat kuat, canggih dan lengkap, sehingga memudahkan pemrograman dalam merancang, membuat dan menyelesaikan aplikasi *database* yang diinginkan. Selain itu, *Delphi* juga dapat menangani data dalam berbagai format *database*, misalnya format *MS.Access, Oracle, Foxro, Informix*. Format *database* yang dianggap asli dari *Delphi* adalah *Paradox* dan *dBase*.