BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Perusahaan

Dari cikal bakal Laboratorium Penelitian & Pengembangan Industri Bidang Pos dan Telekomunikasi (LPPI-POSTEL), pada 30 Desember 1974 berdirilah PT. Industri Telekomunikasi Indonesia (INTI) sebagai Badan Usaha Milik Negara (BUMN) dengan misi untuk menjadi basis dan tulang punggung pembangunan Sistem Telekomunikasi Nasional (SISTELNAS).

Sejak berdirinya sampai sekarang ini, PT. Industri Telekomunikasi Indonesia (Persero) berkedudukan di Kompleks Laboratorium PTT di Jl. Moch Toha No. 77 serta di Jl. Moch Toha 225 Bandung. Seiring waktu dan berbagai dinamika yang harus diadaptasi, seperti perkembangan teknologi, regulasi, dan pasar, maka selama lebih dari 30 tahun berkiprah dalam bidang telekomunikasi, INTI telah mengalami berbagai perubahan dan perkembangan.

1. Era 1974 – 1984

Fasilitas produksi yang dimiliki INTI antara lain adalah:

1. Pabrik Perakitan Telepon
2. Pabrik Perakitan Transmisi
3. Laboratorium Software Komunikasi Data
4. Pabrik Konstruksi & Mekanik

Kerjasama Teknologi yang pernah dilakukan pada era ini antara lain dengan Siemen, BTM, PRX, JRC, dan NEC. Pada era tersebut produk Pesawat Telepon Umum Koin (PTUK) INTI menjadi standar Perumtel (sekarang Telkom).

2. Era 1984 – 1994

Fasilitas produksi terbaru yang dimiliki INTI pada era ini, di samping fasilitas-fasilitas yang sudah ada sebelumnya, antara lain adalah Pabrik Sentral Telepon Digital Indonesia (STDI) pertama di Indonesia dengan teknologi produksi Trough Hole Technology (THT) dan Surface Mounting Technology (SMT).

Kerjasama Teknologi yang pernah dilakukan pada era ini antara lain adalah:

1. Bidang sentral (switching), dengan Siemens
2. Bidang transmisi dengan Siemens, NEC, dan JRC
3. Bidang CPE dengan Siemens, BTM, Tamura, Shapura, dan TatungTEL

Pada era ini, PT. Industri Telekomunikasi Indonesia memiliki reputasi dan prestasi yang signifikan, yaitu:

1. Menjadi pionir dalam proses digitalisasi sistem dan jaringan telekomunikasi di Indonesia.
2. Bersama Telkom telah berhasil dalam proyek otomatisasi telepon di hampir seluruh ibu kota kabupaten dan ibu kota kecamatan di seluruh wilayah Indonesia.

3. Era 1994 – 2000

Selama 20 tahun sejak berdiri, kegiatan utama INTI adalah murni manufaktur. Namun dengan adanya perubahan dan perkembangan kebutuhan teknologi, regulasi dan pasar, INTI mulai melakukan transisi ke bidang jasa *engginering*.

Pada masa ini aktivitas manufaktur di bidang switching, transmisi, CPE dan mekanik-plastik masih dilakukan. Namun situasi pasar yang berubah, kompetisi yang makin ketat dan regulasi telekomunikasi yang makin terbuka menjadikan posisi INTI di pasar bergeser sehingga tidak lagi sebagai market leader. Kondisi ini mengharuskan INTI memiliki kemampuan *sales force* dan *networking* yang lebih baik. Kerjasama teknologi masih berlangsung dengan Siemens secara single-source.

4. Era 2000 – 2004

Pada era ini kerjasama teknologi tidak lagi bersifat *single source*, tetapi dilakukan secara multi source dengan beberapa perusahaan multinasional dari Eropa dan Asia. Aktivitas manufaktur tidak lagi ditangani sendiri oleh INTI, tetapi secara spin-off dengan mendirikan anak-anak perusahaan dan usaha patungan, seperti:

1. Bidang CPE, dibentuk anak perusahaan bernama PT. INTI PISMA International yang bekerja sama dengan JITech International, bertempat di Cileungsi Bogor.
2. Bidang mekanik dan plastik, dibentuk usaha patungan dengan PT. PINDAD bernama PT. IPMS, berkedudukan di Bandung.

3. Bidang-bidang switching, akses dan transmisi, dirintis kerja sama dengan beberapa perusahaan multinasional yang memiliki kapabilitas memadai dan adaptif terhadap kebutuhan pasar. Beberapa perusahan multinasional yang telah melakukan kerjasama pada era ini, antara lain:
    a. SAGEM, di bidang transmisi dan selular
    b. MOTOROLA, di bidang CDMA
    c. ALCATEL, di bidang *fixed & optical access network*
    d. Ericsson, di bidang akses
    e. Hua Wei, di bidang switching & akses

5. 2005 – sekarang

Dari serangkaian tahapan restrukturisasi yang telah dilakukan, INTI pada era 2005 – sekarang, kini memantapkan langkah transformasi mendasar dari kompetensi berbasis manufaktur ke *engginering solution*. Hal ini akan membentuk INTI menjadi semakin adaptif terhadap kemajuan teknologi dan karakteristik serta perilaku pasar.

Dari pengalaman panjang INTI sebagai pendukung utama penyediaan infrastruktur telekomunikasi nasional dan dengan kompetensi sumber daya manusia yang terus diarahkan sesuai proses transformasi tersebut, saat ini INTI bertekad untuk menjadi mitra terpercaya di bidang penyediaan jasa profesional dan solusi total yang fokus pada *Infocom System & Technology Integration* (ISTI).

2.2 Landasan Teori

Pada bagian ini akan dijelaskan mengenai teori-teori yang nantinya akan digunakan untuk bab-bab selanjutnya.

2.2.1 Basis Data

Basis Data adalah kumpulan informasi bermanfaat yang diorganisasikan ke dalam tatacara yang khusus. Sedangkan menurut Fabbri dan Schwab, basis data adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan pengulangan data. Menurut Date, basis data dapat dianggap sebagai tempat untuk sekumpulan berkas data terkomputerisasi.

Satu hal yang harus diperhatikan dalam, bahwa basis data bukan hanya sekedar penyimpanan data secara elektronis (dengan bantuan komputer), artinya tidak semua bentuk penyimpanan elektronis dinamakan basis data, karena yang ditonjolkan dalam basis data adalah pengaturan/ pemilihan/ pengelompokan dan pengorganisasian data yang akan disimpan sesuai dengan fungsi dan jenisnya. Pemilihan/ pengelompokan/ pengorganisasian ini dapat berbentuk sejumlah file/ tabel terpisah atau dalam bentuk pendefinisian kolom-kolom atau field-field data dalam setiap file/tabel.

2.2.2 Model Data

Model data didefinisikan sebagai kumpulan perangkat konseptual untuk menggambarkan data, hubungan data, semantik (makna) data dan batasan data. Karena yang ingin ditunjukkan adalah makna dari data dan keterhubungan dengan

data lain, maka model data ini lebih tepat jika disebut “Model data logik”, penyingkatan menjadi model data dilakukan demi penyederhanaan penulisan. Ada sejumlah cara merepresentasikan model data dalam perancangan basis data, yang secara umum dapat dibagi menjadi 2 (dua) kelompok, yaitu model logik data berdasarkan objek yang terdiri dari :

1. Model keterhubungan entitas (*Entity-Relationship Model*)
2. Model berorientasi objek (*Objek-Oriented Model*)
3. Model data semantik (*Semantic data model*)
4. Model data fungsional (*Functional data model*)

Dan model logic data berdasarkan record, yang terdiri dari :

1. Model relasional (Relational Model)
2. Model hirarkis (Hierarchical Model)
3. Model Jaringan (Network Model)

2.2.3 *Entity-Relationship Diagram* (ERD)

Diagram *Entity-Relationship* merupakan batu pertama dari model data. ERD pada mulanya diusulkan oleh Peter Chen untuk desain sistem database relasional yang telah dikembangkan oleh yang lainnya. Serangkaian komponen utama diidentifikasikan untuk ERD: objek data, atribut, hubungan dan berbagai tipe indikator. Tujuan utama dari ERD adalah untuk mewakili objek data dan hubungan mereka.

Objek data diwakili oleh sebuah persegi panjang yang diberi label. Hubungan ditunjukkan dengan garis yang diberi label yang menghubungkan

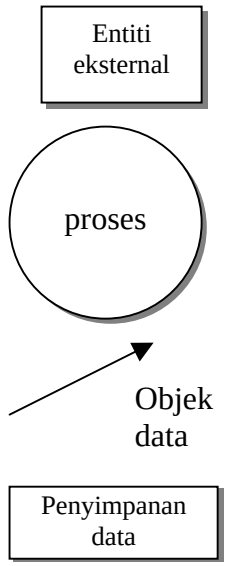
objek. Dalam beberapa variasi ERD, garis yang menghubungkan berisi sebuah permata yang diberi label dengan hubungan tersebut. Sambungan antara data dan objek dan hubungan dibangun dengan menggunakan berbagai simbol khusus yang menunjukan kardinalitas dan modalitas.

2.2.4 *Data Flow Diagram* (DFD)

Diagram alir data atau *Data flow diagram* (DFD) adalah sebuah teknik grafis yang menggambarkan aliran informasi dan transformasi yang diaplikasikan pada saat data bergerak dari input menjadi output. DFD juga dikenal sebagai *grafik aliran data* atau *bubble chart*. DFD dapat digunakan untuk menyajikan sebuah sistem atau perangkat lunak pada setiap tingkat abstraksi. Kenyataannya, DFD dapat dipartisi ke dalam tingkat-tingkat yang merepresentasikan aliran informasi yang bertambah dan fungsi ideal.

DFD tingkat 0, yang disebut juga dengan *model sistem fundamentasi* atau model konteks, merepresentasikan seluruh elemen sistem sebagai sebuah bubble tunggal dengan data input dan output yang ditunjukan oleh anak panah yang masuk dan keluar secara berurutan. Proses tambahan (bubble) dan jalur aliran informasi direpresentasikan pada saat DFD tingkat 0 dipartisi untuk mengungkap detail yang lebih. Notasi dasar yang digunakan untuk menciptakan suatu DFD yaitu sebuah persegi panjang yang digunakan untuk merepresentasikan sebuah *entitas eksternal*, yaitu sebuah elemen sistem atau sistem lain yang menghasilkan informasi bagi transformasi perangkat lunak. Lingkaran merepresentasikan sebuah *proses* atau *transformasi* yang diaplikasikan ke data (atau kontrol) dan

mengubahnya dengan berbagai macam cara. Anak panah melambangkan satu atau lebih data. Garis double merepresentasikan sebuah *penyimpanan data*-informasi tersimpan yang digunakan oleh perangkat lunak.



Prosedur atau konsumer informasi yang ada diluar bound sistem untuk dimodelkan.

Transfer informasi (fungsi) yang ada di luar bound sistem untuk dimodelkan.

Objek data; anak panah menunjukkan arah aliran data

Repositori data yang disimpan untuk digunakan oleh satu atau lebih, proses disederhanakan buffer atau queque, atau serumit database relasional.

Gambar 2.1 Notasi DFD dasar

2.2.5 Penjadwalan Diklat

Teori penjadwalan sangat terkait dengan model matematika yang berhubungan dengan fungsi penjadwalan, dan perkembangan dari pemanfaatan model dan teknik yang telah berlanjut menjadi antarmuka antara teori dan praktek. Teori perspektif adalah sebagian besar dari suatu pendekatan kuantitatif, salah satunya mencoba untuk mengambil struktur masalah ke dalam bentuk matematika yang ringkas. Khususnya, pendekatan kuantitatif ini dimulai dengan suatu translasi dari pengambilan keputusan tujuan kedalam fungsi objektif eksplisit dan pengambilan keputusan batasan ke dalam batasan eksplisit.

Masalah penjadwalan diklat merupakan masalah kompleks yang harus dihadapi. Inti dari penjadwalan diklat adalah bagaimana menjadwalkan sejumlah komponen yang terdiri dari bidang kompetensi, banyaknya instruktur, waktu dan ruang pada setiap harinya dan sejumlah batasan dan syarat (*constraint*) tertentu dengan tujuan agar tidak terjadi bentrok antara jadwal yang satu dengan yang lain.

2.2.6 Struktur Umum Algoritma Genetika

Pada Algoritma ini, pencarian dilakukan sekaligus atas jumlah solusi yang mungkin dikenal dengan istilah *populasi*. Individu yang terdapat dalam suatu populasi disebut dengan *kromosom*. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun secara acak, sedangkan populasi berikutnya merupakan hasil evaluasi kromosom-kromosom melalui iterasi yang disebut *generasi*. Pada setiap generasi kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut dengan *fitness*. Nilai *fitness* dari suatu kromosom akan menunjukan kualitas kromosom dalam populasi tersebut. Generasi berikutnya disebut juga dengan istilah anak (*offspring*), yang terbentuk dari gabungan dua kromosom generasi sekarang yang bertindak sebagai induk (*parent*) dengan suatu kromosom dapat juga dimodifikasi dengan menggunakan operator mutasi.

Populasi generasi yang baru dibangun dengan cara menyeleksi nilai *fitness* dari kromosom induk dan nilai *fitness* dari kromosom anak, serta menolak kromosom-kromosom yang lainnya sehingga ukuran populasi konstan. Setelah melalui beberapa generasi algoritma ini akan konvergen ke kromosom terbaik.

2.2.6.1 Algoritma Genetika

Algoritma Genetika merupakan suatu algoritma pencarian yang bekerja berdasarkan mekanisme dari seleksi alam dan genetik. Pada genetik, *kromosom* terdiri dari susunan gen-gen. Tiap gen mempunyai nilai atau sifat tertentu yang disebut *allele*, sedangkan posisi gen dalam kromosom disebut *locus*. Selanjutnya satu atau beberapa kromosom bergabung membentuk paket genetik yang disebut *genotif*. Interaksi *genotif* ini dengan lingkungannya disebut *fenotif*.

Pada Algoritma Genetika, kromosom bersesuaian dengan string yang dibentuk dari beberapa karakter. Setiap karakter ini mempunyai posisi (*locus*) dan mengandung nilai tertentu (*allele*). Satu atau beberpa string akan bergabung membentuk struktur (*genotif*) dan bila struktur tersebut dikodekan akan memperoleh satu titik yang merupakan satu alternatif solusi (*fenotif*).

2.2.6.2 Komponen Utama Algoritma Genetika

Ada 6 komponen dalam Algoritma Genetika, yaitu :

2.2.6.2.1 Representasi Kromosom

Gen merupakan bagian dari kromosom. Satu gen biasanya mewakili satu variabel. Gen dapat direpresentasikan ke dalam bentuk : string, bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program atau representasikan lainnya yang dapat diimplementasikan untuk operator genetika.

2.2.6.2.2 Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetik yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan dominasi dan solusi kendala permasalahan yang ada.

2.2.6.2.3 Fungsi Evaluasi

Ada dua hal yang harus dilakukan dalam melakukan evaluasi kromosom yaitu : Fungsi obyektif (fungsi tujuan) dan konversi fungsi obyektif ke dalam fungsi *fitness*. Secara umum fungsi *fitness* diturunkan dari fungsi obyektif dengan nilai yang tidak negatif, maka perlu ditambahkan suatu konstanta *x* agar nilai *fitness* yang terbentuk menjadi tidak negatif.

2.2.6.2.4 Seleksi

Ada beberapa metode seleksi yang digunakan dalam Algoritma Genetika, yaitu :

1. *Roulette Wheel Selection*

Metode ini menirukan permainan *Roulette Wheel* dimana masing-masing kromosom menempati potongan lingkaran pada roda *roulette* secara proporsional sesuai dengan nilai *fitness*nya. Kromosom yang memiliki nilai *fitness* yang lebih

besar menempati potongan lingkaran yang lebih besar dibandingkan dengan kromosom bernilai *fitness* rendah.

2. *Rank Selection*

Pada *Rank Selection*, yang pertama dilakukan adalah meranking populasi sehingga setiap kromosom akan menerima nilai *fitness* 1, kedua terburuk bernilai 2 dan seterusnya, dan kromosom terbaik akan memiliki nilai *fitness* N (sejumlah kromosom dalam populasi).

3. *Steady-state Selection*

Cara kerja *Steady-state Selection* adalah sebagai berikut. Pada setiap generasi beberapa kromosom dengan nilai *fitness* tertinggi dipilih untul membentuk *offspring* baru. Kemudian beberapa kromosom yang *fitness*nya terendah dibuang dan *offspring* baru menggantikan tempat mereka. Populasi yang *survive* akan menuju ke generasi selanjutnya.

4. *Elitism*

Elitism adalah suatu metode yang berguna untuk mengcopy kromosom terbaik (atau beberapa kromosom terbaik) pada suatu populasi dan mempertahankannya di populasi yang baru. Sisa populasi yang lain dibangun dengan cara diatas.

2.2.6.2.5 Operator Genetik

Operator Genetik merupakan cara kerja yang dilakukan dalam proses Algoritma Genetika untuk dapat menghasilkan individu baru yang lebih baik dari sebelumnya.

2.2.6.2.5.1 *Crossover*

Pada proses ini akan dipilih secara acak dua individu dan tempat pertukaran, dimana akan menandai kromosom yang berbeda antara kedua tempat pertukaran akan bertukar tempat satu sama lain.

2.2.6.2.5.2 Mutasi

Mutasi adalah operator yang menghasilkan perubahan acak secara spontan pada kromosom. Pada dasarnya akan mengubah secara acak nilai suatu bit pada posisi tertentu, dengan mengganti bit 1 dengan bit 0, atau sebaliknya. Pada mutasi sangat dimungkinkan muncul kromosom baru yang semula belum muncul dalam populasi awal. Faktor terbesar dalam teori evolusi yang menyebabkan suatu kromosom dapat bertahan hidup, mati, melakukan persilangan atau melakukan mutasi adalah lingkungan.

2.2.6.3 Parameter Algoritma Genetika

Yang dimaksud dengan parameter adalah parameter kontrol Algoritma Genetika. Nilai parameter ini ditentukan juga berdasarkan permasalahan yang akan dipecahkan.

2.2.6.3.1 Probabilitas *Crossover* (Pc)

Menunjukan rasio dari anak yang dihasilkan dalam setiap generasi dengan ukuran populasi. Jika *crossover* tidak ada, maka *offspring* akan meniru dari induknya. Tetapi jika *crossover* ada, *offspring* dibuat dari bagian kromosom induk

(jika kemungkinan *crossover* adalah 100% maka semua *offspring* dibuat oleh *crossover*, tetapi jika 0% maka *offspring* dibuat dengan meniru kromosom dari populasi lama secara keseluruhan).

2.2.6.3.2 Probabilitas Mutasi (Pm)

Menunjukan presentasi jumlah total gen pada populasi yang akan mengalami mutasi. Jika tidak ada mutasi, *offspring* langsung mengambil dari bentuk *crossover* (meniru) dengan tidak merubah lagi. Jika mutasi dilakukan, bagian kromosom diubah (jika kemungkinan 100%, seluruh kromosom akan diubah dan jika 0% tidak ada yang diubah). Mutasi dibuat untuk mencegah penurunan kualitas dari kromosom yang telah didapat.

2.2.6.3.4 Population Size

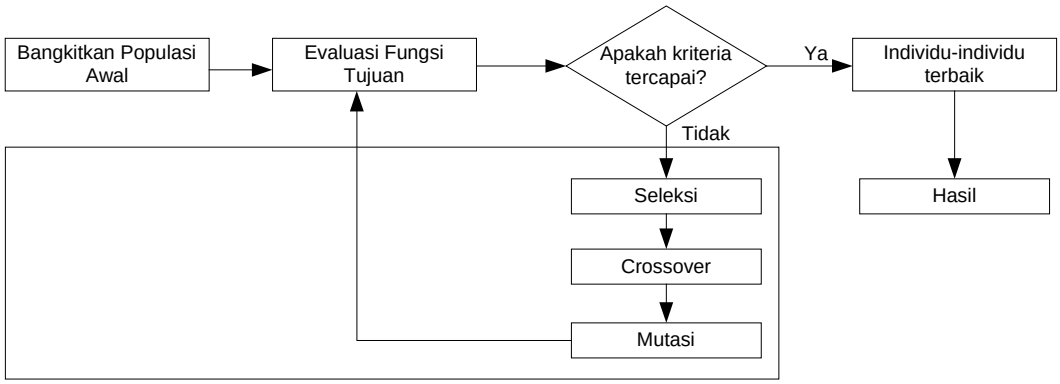
Merupakan kumpulan kromosom dari populasi (1 generasi). Dari setiap generasi ukuran populasi harus sama.

2.2.6.4 Cara Kerja Algoritma Genetika Sederhana

Secara sederhana Algoritma Genetika terdiri dari langkah-langkah :

1. Generasi = 0 (generasi awal)
2. Inisialisasi *populasi* awal, P(generasi) secara acak
3. Evaluasi nilai *fitness* pada setiap individu dalam P(generasi)
4. Kerjakan langkah-langkah berikut hingga generasi mencapai Maksimal generasi.

a. Generasi = generasi + 1 (tambah generasi)
b. Seleksi *populasi* tersebut untuk mendapatkan kandidat induk, P’(generasi)
c. Lakukan *crossover* pada P’(generasi)
d. Lakukan Mutasi pada P’(generasi)
e. Lakukan evaluasi *fitness* setiap individu pada P’(generasi)
f. Bentuk populasi baru: P(generasi) = {P(generasi-1) yang *survive*, P’(generasi)}



Gambar 2.2 Diagram Alir Algoritma Genetika Sederhana

2.2.7 Sekilas tentang Borland Delphi 7.0

Aplikasi merupakan penerapan, pengimplementasian suatu permasalahan pekerjaan ke dalam suatu sarana atau media yang dapat digunakan untuk menerapkan permasalahan tersebut, sehingga menjadi suatu bentuk program tanpa menghilangkan nilai-nilai dasar dari permasalahan yang ada. Aplikasi dapat dibedakan menjadi dua yaitu Aplikasi *Windows* dan Aplikasi *Konsol.* Aplikasi *Windows* adalah aplikasi yang berjalan pada *Windows.* Aplikasi non-*Windows* contohnya yang berjalan pada DOS yang biasa disebut aplikasi *Konsol.*

Secara umum, sebuah aplikasi paling sedikit melibatkan sebuah *form.* Namun biasabya sebuah aplikasi melibatkan banyak *form.* Sebuah *form* umumnya melibatkan komponen lain kotak *Combo Box* dan Tombol *Radio Button* merupakan contoh komponen tersebut. Tetapi tidak semua komponen terlihat secara visual, komponen yang terlihat sevara visual biasa disebut dengan kontrol pada Delphi. Sebuah aplikasi akan diletakkan pada sebuah *Project*. Sebuah *project* dapat membawahi beberapa *form. Form* merupakan inti dari sebuah aplikasi pada Delphi karena dianggap sebagai dasar aplikasi *Windows.*

2.2.7.1 Gambaran Singkat Bagian-bagian *IDE*

IDE *(Intergrated Development Environment)* adalah sebuah lingkungan dimana semua *tools* yang diperlukan untuk mendesain, menjalanklan dan mengetes sebuah aplikasi yang terjadi dan terhubung baik sehingga memudahkan program. Melalui *IDE* ini pemprograman sevara visual merancang tampilan untuk pemakai (antar muka pemakai) dan menuliskan kode.

2.2.7.2 Kelebihan-kelebihan Delphi

Membangun perangkat lunak ini dilakukan dalam lingkungan pengembangan terpadu *(Intergrated Development Environment)* Delphi 7 dari Borland. Pemilihan *tools* pengembangan ini didasarkan pada beberapa pertimbangan sebagai berikut :

1. Delphi mendukung pemrograman berorientasi objek. Dengan demikian, hasil analisis dan desain berorientasi objek yang telah dilakukan pada tahap sebelumnya dapat dimplementasinkan dengan mudah.
2. Delphi memiliki kelas-kelas yang telah terdefinisi, hal ini membuat pemrograman tidak perlu dilakukan dari awal. Pemrograman cukup dilakukan dengan menggunakan kembali atau menurunkan kelas-kelas baru dari kelas yang sudah ada.
3. Delphi dalam pembangunan kelas-kelas kontrol antarmuka visual yang telah terdefinisi. Hal ini memudahkan dalam pembangunan perangkat lunak dengan antarmuka grafis yang *User Friendly*.
4. Delphi memiliki fasilitas *Borland Database Engine (BDE)* yang memberikan lingkungan pemrograman basis data tang tranparan dengan kecepatan pengaksesan yang jauh lebih baik.

2.2.8 MySQL

MySQL (*My Structure Query Language*) 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 Open Source.

Kelebihan lain dari MySQL adalah ia menggunakan bahasa Query standar yang dimiliki SQL (*Structure Query Language*). SQL adalah suatu bahasa permintaan yang terstruktur yang telah distandarkan untuk semua program pengakses database seperti Oracle, Posgres SQL, SQL Server, dan lain-lain.

Sebagai sebuah program penghasil database, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi lain (*interface*). MySQL dapat didukung oleh hampir semua program aplikasi baik yang *open source* seperti PHP maupun yang tidak, yang ada pada platform windows seperti Visual Basic, Delphi dan lain-lain.