# BAB II

# LANDASAN TEORI

## 2.1 Teknologi Telepon Selular

Telepon Seluler (Ponsel) atau *handphone* (HP) pada saat ini sudah menjadi kebutuhan primer. Banyak orang telah mengenal dan menggunakannya sebagai peralatan yang praktis untuk melakukan komunikasi dimanapun mereka berada, tanpa dibatasi ruang dan rentang panjang kabel. Fungsi ponsel kini semakin berkembang, tidak hanya digunakan sebagai alat komunikasi praktis saja. Fungsi ini memang sangat bervariasi tergantung pada model ponsel yang semakin berkembang.

## 2.2 SMS (*Short Message Service*)

### 2.2.1 Sejarah SMS

SMS *(Short Message Service)* adalah salah satu fasilitas dari teknologi GSM yang memungkinkan mengirim dan menerima pesan – pesan singkat berupa text dari MS *(Mobile Station)*. Layanan SMS juga memungkinkan pengiriman pesan dalam bentuk *alphanumeric*, layanan SMS ini banyak diaplikasikan pada sistem komunikasi tanpa kabel (*wireless).* Teknologi wireless dipelopori dari kawasan Eropa yang diawali pada kebutuhan bersama terhadap satu sistem jaringan baru yang dapat menjadi standard jaringan yang berlaku dan dapat diterapkan di seluruh kawasan Eropa. Dalam sistem baru juga harus terdapat kemampuan yang

dapat mengantisipasi mobilitas pengguna serta kemampuan melayani lebih banyak pengguna untuk menampung penambahan jumlah pelanggan baru.

Karena hal ini tidak dapat dilakukan dengan mempertahankan sistem analog, maka kemudian diputuskan untuk merombak sistem dan menggantinya dengan sistem digital. Standard baru diperkenalkan dengan nama GSM *(Global Standard for Mobile Communications)*. GSM pada awalnya adalah kepanjangan dari *Groupe Speciale Mobile*, sebuah badan gabungan dari para ahli yang melakukan studi bersama untuk menciptakan standard GSM tersebut.

Pada bulan Desember 1992, dilakukan pengiriman pesan menggunakan SMS dari sebuah PC *(Personal Computer)* ke telepon *mobile* (bergerak) dalam jaringan GSM milik Vodafone Inggris, kemudian merambah ke benua Amerika yang dipelopori oleh beberapa operator komunikasi bergerak berbasis digital seperti BellSouth Mobility, PrimeCo dan operator lainnya. Teknologi yang digunakan dari pengiriman SMS yaitu *Store* and *forward service*, jadi SMS yang di kirim akan simpan sementara di server SMS center kemudian dialihkan ke nomor tujuan.

### 2.2.2 Arsitektur dan Elemen Jaringan SMS

Layanan SMS dibangun dari berbagai entitas yang saling terkait dan mempunyai fungsi dan tugas masing-masing. Tidak ada satupun dalam sistem SMS yang dapat bekerja secara parsial. Entitas dalam jaringan SMS ini disebut juga elemen SMS.

Elemen-elemen dasar pada jaringan SMS :

a. *SME (Short Message Entity)*, merupakan tempat penyimpanan dan pengiriman pesan yang akan dikirimkan ke MS tertentu.
b. *SC (Service Centre)*, bertugas untuk menerima pesan dari SME dan melakukan *forwarding* ke alamat MS yang dituju.
c. *SMS-GMSC (Short Message Service – Gateway SMC* ), melakukan penerimaan pesan dari SC dan memeriksa parameter yang ada. Selain itu GMSC juga mencari alamat MS yang dituju dangan bantuan HLR, dan mengirimkannya kembali ke MSC yang dimaksud.
d. SMS – IWMSC *(Short Message Service – Interworking MSC* ), berperan dalam SMS *Message Originating*, yaitu menerima pesan dari MSC
e. HLR *(Home Location Register)* merupakan sebuah *database* yang digunakan sebagai tempat penyimpanan permanen data dan profil pelanggan. Bila diminta oleh SMSC, maka HLR dapat memberikan informasi *routing* dari pelanggan tertentu. HLR juga dapat memberikan informasi status tujuan apakah aktif atau tidak.
f. VLR *(Visitor Location Register)* merupakan sebuah *database* tempat menyimpan informasi sementara yang berisi data pelanggan dari sebuah HLR yang *roaming* pada HLR lain.
g. MSC merupakan sebuah sistem yang melakukan fungsi *switching* dan mengontrol panggilan telepon dalam sebuah jaringan komunikasi bergerak. MSC inilah yang akan mengirimkan sebuah SMS ke suatu tujuan tertentu melalui *base station* yang sesuai.

h. BSS *(Base Station System)* Merupakan kesatuan sistem yang bertanggung jawab mengatur transmisi sinyal elektromagnetik untuk membawa data dari MSC ke perangkat telepon bergerak. *Base Station* terdiri dari BSC *(Base Station Controler)* dan BTS *(Base Tranceiver Station)* dan juga dikenal dengan nama *cell cite* atau sederhananya *cell*. Sebuah BSC biasanya menangani satu atau lebih BTS dan bertanggung jawab menangani pelanggan saat berpindah dari satu *cell* ke *cell* lainnya.

i. *Mobile Device* merupakan perangkat yang mempunyai kemampuan mengirimkan dan menerima SMS, biasanya berupa telepon seluler dengan teknologi digital. Akan tetapi, saat ini jenis terminal berkembang sesuai aplikasi dan kebutuhan seperti POS, laptop dan PDA *(Personal Digital Assistant)*.

### 2.2.3 Layanan Aplikasi SMS

Layanan aplikasi SMS merupakan sebuah layanan yang bersifat *none real time* dimana sebuah SMS dapat di *submit* ke suatu tujuan, tidak peduli apakah tujuan tersebut aktif atau tidak. Bila terdeteksi bahwa tujuan tidak aktif, maka sistem akan menunda pengiriman ke tujuan hingga tujuan aktif kembali. Pada dasarnya sistem SMS akan menjamin *delivery* dari suatu SMS hingga sampai tujuan. Kegagalan pengiriman yang bersifat sementara seperti tujuan tidak aktif akan selalu teridentifikasi sehingga pengiriman ulang SMS akan selalu dilakukan kecuali apabila diberlakukuan aturan bahwa SMS yang telah melampaui batas waktu tertentu harus dihapus dan dinyatakan gagal kirim. Berdasarkan mekanisme

distribusi pesan SMS diperoleh aplikasi SMS, terdapat 4 macam mekanisme pengantaran pesan yaitu :

a. *Pull*, yaitu pesan yang dikirimkan ke pengguna berdasarkan permintaan pengguna.
b. *Push – Event based*, yaitu pesan yang diaktivasi oleh aplikasi berdasarkan kejadian yang berlangsung
c. *Push - Schedule*, yaitu pesan yang diaktivasi oleh aplikasi berdasarkan waktu yang telah terjadwal.
d. *Push – Personal Profile*, yaitu pesan yang diaktivasi oleh aplikasi berdasarkan *profile* dan *preference* dari pengguna

### 2.2.4 Prinsip Kerja SMS

Prinsip Kerja dari SMS diantaranya adalah sebagai berikut :

a. Prinsip kerja dari SMS ini adalah bahwa setiap jaringan mempunyai suatu SC *(service center)*. Pesan tidak langsung dikirimkan ke tempat tujuan melainkan di simpan terlebih dahulu di SC menjadi interface antara PLNM *(public land mobile network).*
b. Transmisi SMS dapat terjadi meskipun MS *(mobile station)* sedang melakukan komunikasi dengan MS yang lain. Hal ini di mungkinkan karena kanal radio untuk transmisi *voice* telah ditentukan selama durasi pemanggilan sedangkan SMS merambat pada kanal radio dengan memanfaatkan jalur *signal*
c. Pengiriman SMS yang menggunakan kanal *signal* memiliki dua tipe, yaitu :

1. SMS *Point To point* yaitu menyediakan mekanisme untuk mengirimkan pesan hanya dari datu MS ke MS tertentu, berupa pesan pendek ke dan dari piranti bergerak
2. SMS broadcast *(Point to multipoint)* pengiriman SMS ke beberapa MS sekaligus

d. Pesan yang tidak terkirim, akan diberikan informasi report yang menyatakan pesan SMS gagal terkirim

e. Jika ponsel tidak aktif untuk sms akan tetap masuk dan di simpan di SMS dengan waktu tertentu, jika ponsel aktif sebelum batas waktu maka SMS akan di kirimkan.

## 2.3 PDU *(Protocol Data Unit)*

Langkah-langkah untuk menentukan kode PDU yang dibutuhkan untuk mengirimkan SMS ke SMS *Center* :

a. Nomor SMS Center

Nomor SMS Center terdiri dari tiga *subheader* yaitu:

1. Jumlah pasangan heksa desimal SMS *Center* dalam bilangan heksa.
2. Kode nasional dan kode internasional, nomor kode nasional yang digunakan adalah 81 sedangkan nomor kode internasional adalah 91.
3. Nomor SMS *Center*, dalam pasangan bilangan heksa yang saling dipertukarkan dengan aturan tertentu, jika tersisa satu angka heksa yang tidak memiliki pasangan maka angka tersebut dipasangkan dengan huruf F didepannya.

Contoh konversi nomor ponsel kedalam format kode PDU dengan menggunakan :

1. Kode Nasional

   SMS Center menggunakan nomor 0856000000 maka :

   a. Terdapat 6 pasang angka heksa desimal.

   b. Kode nasional 81, ada 1 pasang

   c. 80-56-00-00-00

   Maka kode PDU yang didapat : 06818056000000

2. Kode Internasional

   SMS Center menggunakan nomor 62811000000, maka:

   a. Terdapat 7 pasang angka heksa desimal

   b. Kode internasional 91, ada 1 pasang

   c. 26-18-01-00-00-F0

   Maka kode PDU yang didapat : 07912618010000F0

Pada tabel 2.1 diperlihatkan beberapa nomor SMS *Center* operator selular di Indonesia :

Tabel 2.1 Tabel Nomor *SMS-Center* (a) Operator Selular dan Kode PDU (Cara I) dan (b) Operator Selular dan Kode PDU (Cara II)

| No | Operator Selular | SMS-*Center* No | Kode PDU |
|---|---|---|---|
| 1 | Telkomsel | 0811000000 | 06818011000000 |
| 2 | Satelindo | 0816125 | 0581806121F5 |
| 3 | Excelcom | 0818445009 | 06818081440590 |
| 4 | Indosat-M3 | 0855000000 | 06818055000000 |

Keterangan : Operator selular dan kode PDU (Cara I)

| No | Operator Selular | SMS-*Center* No | Kode PDU |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 1 | Telkomsel | 6281000000 | 07912618010000F0 |
| 2 | Satelindo | 62816125 | 059126181652 |
| 3 | Excelcom | 62818445009 | 07912618485400F9 |
| 4 | Indosat-M3 | 62855000000 | 07912658050000F0 |

Keterangan : Operator selular dan kode PDU (Cara II)

b. Tipe SMS, untuk pengiriman atau *send* tipe SMS adalah 1 dalam heksa 01

c. Nomor referensi SMS, nomor ini dibiarkan dulu 0 atau dalam heksa 00, ponsel secara otomatis akan memberi nilai nomor referensinya.

d. Nomor ponsel penerima, terdiri atas tiga bagian *subheader* yaitu:

   1. Jumlah bilangan desimal ponsel yang dituju dalam bilangan heksa
   2. Kode nasional atau kode internasional, kode *subheader* nasional adalah 81 dan internasional adalah 91.
   3. Nomor ponsel yang dituju dalam pasangan heksa yang saling dipertukarkan menurut aturan tertentu.

Contoh untuk nomor ponsel yang dituju adalah 085221960301, maka konversi kedalam heksa desimal yaitu dengan cara seperti yang dijabarkan berikut.

1. Kode Nasional :

a. terdapat 12 angka, berarti dalam heksa adalah 0C

b. kode nasional adalah 81

c. 80-25-12-69-03-01

Maka kode *subheader* PDU yang didapat adalah 0C81802512690301

2. Kode Internasional

   a. 62 81 32 12 17 32 6

b. Terdapat 13 angka, berarti dalam heksa adalah 0D

c. Kode internasional adalah 91

d. 261823212732F6

Maka kode *subheader* PDU yang didapat adalah 0D91261823212732F6

e. Bentuk SMS

00 : dikirim sebagai SMS

01 : dikirim sebagai Telekomunikasi

f. Skema *encoding* data *Input* dan *Output* (I/O)

Terdapat dua skema yaitu :

1. Skema 7 bit, ditandai dengan angka 00
2. Skema 8 bit, ditandai lebih besar dari nol (dalam heksa)

g. Jangka Waktu atau validasi sebelum SMS *expired* atau kadaluarsa, jika bagian ini terlewati itu berarti berlakunya SMS tidak dibatasi, sedangkan jika diisi dengan suatu bilangan integer yang kemudian diubah ke dalam bilangan heksa, maka bilangan tersebut mewakili jumlah waktu validasi SMS tersebut. Pada tabel 2.2 diperlihatkan validitas waktu SMS.

Tabel 2.2 Tabel Validitas Waktu SMS

| Integer (INT) | Jangka Waktu Validasi |
|---|---|
| 0-143 | (INT+1)x 5 menit (berarti 5 sampai dengan 12 jam) |
| 144-167 | 12 jam + (INT-143) x 30 menit |
| 168-196 | (INT-166) x 1 hari |
| 197-255 | (INT-192) x 1 minggu |

h. Isi SMS

Terdiri dari dua subheader yaitu:

1. Panjang Isi (Jumlah huruf dalam SMS), misal untuk kata “hello” terdiri dari 5 huruf.
2. Isi berupa pasangan bilangan heksa desimal, skema 7 bit dapat dilihat pada Tabel 2.3 dibawah ini.

Tabel 2.3 Tabel Konversi Heksadesimal

| | | | | **B7** | ***0*** | ***0*** | ***0*** | ***0*** | ***1*** | ***1*** | ***1*** | ***1*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | **B6** | ***0*** | ***0*** | ***1*** | ***1*** | ***0*** | ***0*** | ***1*** | ***1*** |
| | | | | **B5** | ***0*** | ***1*** | ***0*** | ***1*** | ***0*** | ***1*** | ***0*** | ***1*** |
| **B4** | **B3** | **B2** | **B1** | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| ***0*** | ***0*** | ***0*** | ***0*** | **0** | ***@*** | Δ | ***SP*** | ***0*** | ***-*** | ***P*** | | ***p*** |
| ***0*** | ***0*** | ***0*** | ***1*** | **1** | | | ***!*** | ***1*** | ***A*** | ***Q*** | ***a*** | ***q*** |
| ***0*** | ***0*** | ***1*** | ***0*** | **2** | ***$*** | ***Φ*** | ***“*** | ***2*** | ***B*** | ***R*** | ***b*** | ***r*** |
| ***0*** | ***0*** | ***1*** | ***1*** | **3** | | ***ƨ*** | ***#*** | ***3*** | ***C*** | ***S*** | ***c*** | ***s*** |
| ***0*** | ***1*** | ***0*** | ***0*** | **4** | | | | ***4*** | ***D*** | ***T*** | ***d*** | ***t*** |
| ***0*** | ***1*** | ***0*** | ***1*** | **5** | | | ***%*** | ***5*** | ***E*** | ***U*** | ***e*** | ***u*** |
| ***0*** | ***1*** | ***1*** | ***0*** | **6** | | | ***&*** | ***6*** | ***F*** | ***V*** | ***f*** | ***v*** |
| ***0*** | ***1*** | ***1*** | ***1*** | **7** | | | ***.*** | ***7*** | ***G*** | ***W*** | ***g*** | ***w*** |
| ***1*** | ***0*** | ***0*** | ***0*** | **8** | | | ***(*** | ***8*** | ***H*** | ***X*** | ***h*** | ***x*** |
| ***1*** | ***0*** | ***0*** | ***1*** | **9** | | | ***)*** | ***9*** | ***I*** | ***Y*** | ***i*** | ***y*** |
| ***1*** | ***0*** | ***1*** | ***0*** | **10** | ***LF*** | | ******* | ***:*** | ***J*** | ***Z*** | ***j*** | ***z*** |
| ***1*** | ***0*** | ***1*** | ***1*** | **11** | | | + | ***;*** | ***K*** | | ***k*** | |
| ***1*** | ***1*** | ***0*** | ***0*** | **12** | | | ‘ | < | ***L*** | | ***l*** | |
| ***1*** | ***1*** | ***0*** | ***1*** | **13** | ***CR*** | | ***_*** | = | ***M*** | | ***m*** | |
| ***1*** | ***1*** | ***1*** | ***0*** | **14** | | | | > | ***N*** | | ***n*** | |
| ***1*** | ***1*** | ***1*** | ***1*** | **15** | | | ***/*** | ***?*** | ***O*** | | ***o*** | |

Ada 2 langkah yang harus kita lakukan untuk mengkonversikan isi SMS , yaitu :

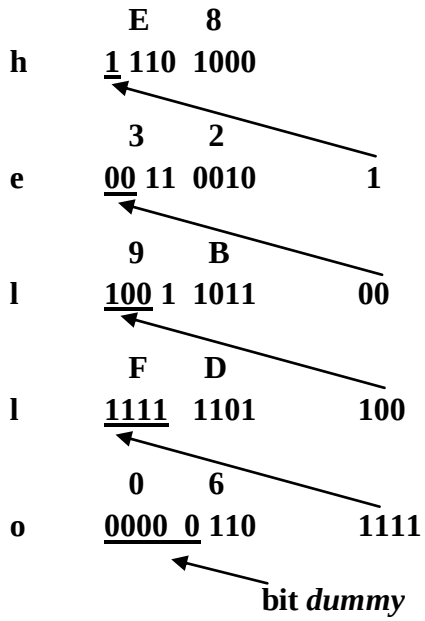
1. Langkah pertama yaitu mengubah menjadi kode 7 bit

   Contoh untuk kata “hello”

| **Langkah pertama** | **Bit** | **7** | **1** |
|---|---|---|---|
| | h | 110 | 1000 |
| | e | 110 | 0101 |
| | l | 110 | 1100 |
| | l | 110 | 1100 |
| | o | 110 | 1111 |

2. Langkah kedua yaitu mengubah kode 7 bit menjadi 8 bit yang diwakili oleh pasangan heksa.

   **Langkah kedua**




Sehingga hasil konversi kata “hello” ke bilangan heksa adalah **E8329BFD06.**

Oleh karena total 7 bit x 5 huruf = 35 bit, sedangkan yang kita perlukan untuk mengubah ke 8 bit adalah 8 bit x 5 huruf = 40 bit, maka diperlukan 5 bit *dummy* yang diisi bilangan 0.

Setelah masing-masing *header* dan *subheader* untuk mengirim pesan pesan dipecahkan, maka *header-header* diatas digabung menjadi sebuah PDU lengkap.

Seperti contoh untuk mengirim kata “Hello”ke MS nomor 628129573337 lewat SMS Center Exelcom tanpa membatasi jangka waktu valid, maka PDU lengkapnya adalah :

**07912618485400F901000C9126I892753373000005E8329BFD06**

i. Tanggal dan waktu SMS di-*stamp* di SMS *Center* diwakili oleh 12 bilangan heksa (6 pasang) yang berarti YY/MM/DD hh:mm:ss

Contoh :

207022512308 02/07/22 12:32:08 22 Juli2002 15:23:08 WIB

## 2.4 AT Command

AT-*Command* atau *Attention Command* yaitu perintah AT (Hayes AT-*Command)* yang digunakan untuk berkomunikasi dengan terminal (modem) melalui gerbang serial pada komputer. Dengan penggunaan perintah AT, dapat diketahui atau dibaca kondisi dari terminal, seperti mengetahui kondisi sinyal, kondisi baterai, mengirim pesan, membaca pesan, menambah item pada daftar telepon, dan sebagainya. Beberapa jenis ponsel memiliki *extended* AT *Command* yang bisa digunakan untuk mengambil informasi jenis, model hp, nomor *Internasional Mobile Station Equipment Identity* (IMEI) , SIM *Subscriber Identification Number* (IMSI), status baterai, kekuatan sinyal, nama operator, lokasi dan *cell* ID. Pada tabel 2.4 berikut diperlihatkan beberapa jenis perintah Hayes yang berhubungan dengan penanganan pesan-pesan AT-Command GSM versi 07.07.

Tabel 2.4 Perintah AT-Command pada GSM Versi 07.07

| **AT-Command** | **Singkatan** | **Fungsi** |
|---|---|---|
| ATE1 | *Activate Command* | Mengaktivasi At-Command 07.07 |
| AT+C..... | *Attention + Command.....* | Awal perintah hayes untuk.... |
| AT+COPS | *Operator Selection* | Memilih operator jaringan |
| AT+CLCK | *Lock* | Menampilkan penguncian telepon |
| AT+CBC | *Battery Charge* | Menampilkan level baterei |
| AT+CSQ | *Signal Quality* | Menampilkan kualitas sinyal |
| AT+CCLK | *Clock Mode* | Pengaturan Jam |
| AT+CPBR | *Phone Book Read* | Membaca daftar buku telepon |
| AT+CPBS | *Phone Book Selection* | Menampilkan isi buku telepon |
| AT+CCSA | *Service Center Address* | Menampilkan Lokasi *Service Center* |
| AT+CMGS | *Message Get Send* | Mengirimkan SMS |
| AT+CMGL | *Message Get List* | Membaca isi *inbox* dan *outbox* SMS |
| AT+CMGD | *Message Get Delete* | Menghapus isi *SMS* |

## 2.5 DBMS *(Database Mangement System)*

DBMS merupakan kumpulan *database* (basis data) beserta aplikasi yang digunakan untuk mengolah basis data tersebut. Berdasarkan orientasi pemakainya, DBMS dapat dikelompokan dalam 2 kategori, yaitu :

1. DBMS yang berorientasi untuk satu atau sedikit pemakai

   Adapun contoh dari DBMS yang di maksud antara lain MS-Access, dBase, foxbase, borland paradox.DBMS ini dapat dengan mudah di pasang di *personal computer* (PC).

2. DBMS yang berorientasi untuk banyak pemakai

   Oracle, borland interbase, MS-SQL-Server, sysbase,infomix merupakan contoh-contoh DBMS yang lebih berorientasi untuk banyak pemakai karena itu lebih ditujukan untuk pemakaian pada sistem jaringan komputer.

DBMS pada umumnya meyediakan beberapa fasilitas untuk operasi pengolahan data atau sering dikenal operasi basis data, antara lain :

1. Penambahan Data.
2. Pencarian Data.
3. Pengubahan Data.
4. Penghapusan data.
5. Pengurutan Data.
6. Penggabungan Data.

## 2.6 *Database*

Semua yang berhubungan dengan informasi hampir tidak terlepas dari *database*. Misalkan buku alamat yang memuat nama, alamat dan nomor telepon. Sebagai kumpulan dari data atau informasi yang teratur, buku alamat dapat digolongkan sebagai *database*. Pada dasarnya suatu *database* adalah kumpulan data atau informasi yang diorganisasikan dan saling berhubungan. Dalam era komputerisasi, *database* menjadi hal yang istimewa. Kumpulan data atau informasi dalam jumlah besar tersebut dengan satu perangkat lunak komputer yaitu program *database*, akan menghasilkan informasi yang sangat penting bagi pengelolaan suatu informasi.

### 2.6.1 Definisi Basis Data

Basis dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul. Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia, barang, hewan, peristiwa, konsep, keadaan, atau yang lainnya yang direpresentasikan dalam bentuk angka, huruf, simbol, *text*, gambar, bunyi, atau kombinasi diantaranya.

Basis Data sendiri dapat didefenisikan dalam sejumlah presfektif yang berbeda, diantaranya :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (*redudansi*) yang tidak perlu untuk memenuhi berbagai kebutuhan.

*Database* sering dianggap sebagai suatu sistem yang harus ditunjang oleh perangkat lunak dan perangkat keras. Perangkat lunak bisa berupa aturan-aturan , tata cara, program (*software*), dan pengolahan data. Sedangkan perangkat keras dapat berupa kertas, buku, seperangkat komputer, atau lemari arsip. Salah satu tugas atau kegunaan *database* adalah membuat model konseptual. Pembuatan model konseptual adalah pembuatan model berdasarkan sistem informasi dari *user*, tanpa tergantung pada *database manajemen sistem* (DBMS) yang akan dipakai.

### 2.6.2 Fungsi Sistem Database

Permasalahan bagaimana mengolah data yang cukup besar dan dapat dipakai dengan cepat, tepat, teliti, serta terjamin kebenarannya dapat diatasi dengan adanya proses komputerisasi pengolahan data, dan dalam hal ini sistem *database*-lah yang akan digunakan.

Tujuan utama dalam pengelolaan data dalam sebuah *database* adalah agar dapat dengan mudah, cepat, dan akurat memperoleh data yang dibutuhkan. Secara lebih lengkap, pemanfaatan *database* dilakukan untuk memenuhi sejumlah tujuan (objektif) seperti berikut :

1. Kecepatan dan Kemudahan

   Pemanfaatan *database* memungkinkan untuk dapat menyimpan atau melakukan manipulasi terhadap data atau menampilkan data tersebut dengan cepat dan mudah.

2. Efesiensi Ruang Penyimpanan Data

   Karena adanya keterkaitan antara data dalam sebuah *database*, maka *redudansi* data pasti aka terjadi. Banyaknya *redudansi* yang ada tentu saja akan memakan banyak ruang penyimpanan. Dengan *database*, efesiensi penggunaan ruang penyimpanan dapat dilakukan dengan cara membuat pengkodean atau membuat relasi-relasi antar kelompok data yang saling berhubungan.

3. Keakuratan Data

   Pemanfaatan pengkodean data atau pembentukan relasi antar data dengan penerapan aturan atau batasan tipe data, domain data, keunikan data, dan

sebagainya, yang secara ketat dapat diterapkan dalam sebuah *database* akan sangat berguna untuk menekan ketidakakuratan pemasukan atau penyimpanan data.

4. Ketersediaan Data

   Pertumbuhan data sejalan dengan berjalannya waktu akan semakin membutuhkan ruang penyimpanan yang semakin besar. Untuk itu data-data yang tidak dibutuhkan lagi dapat dihapus atau dipindahkan ketempat lain, selain itu data juga dapat disebarkan ke berbagai tempat yang terhubung dengan sistem.

5. Kelengkapan Data

   Untuk mengakomodasi kebutuhan kelengkapan data yang semakin berkembang, maka dapat dilakukan proses penambahan *record-record* data, atau dapat melakukan perubahan struktur dalam *database*.

6. Keamanan Data

   Dapat ditentukan *user* mana saja yang boleh mengakses *database* beserta objek-objek didalamnya, dan menentukan jenis operasi apa saja yang yang bisa dilakukan.

7. Kebersamaan Pemakaian

   Karena data yang ada dalam suatu *database* bisa disebar ke berbagai tempat yang terhubung dengan sistem, maka setiap *user* bisa mengakses data secara bersamaan dari tempat yang terpisah.

   Ada beberapa hal yang harus di perhatikan dalam membangun suatu *database*, yaitu :

a. Entitas

Merupakan suatu *object* yang memiliki karakteristik tertentu sebagai suatu atribute, misalnya entitas mahasiswa memiliki atribute nama,alamat dan lain lain.

b. Field

Merepresentasikan suatu atribut dari record yang menunjukkan suatu item dari data, seperti misalnya nama, alamat dan lain sebagainya. Kumpulan dari field membentuk suatu record.

c. Record

Kumpulan dari field membentuk suatu record. Record menggambarkan suatu unit data individu tertentu. Kumpulan dari record membentuk suatu file. Misalnya file personalia, tiap-tiap record dapat mewakili data tiap-tiap karyawan

d. Atribute

Merupakan karakteristik dari sebuah entitas. Nama, alamat merupakan attribute dari entitas mahasiswa.

## 2.7 MySQL

MySQL dikembangkan sekitar sekitar tahun 1994 oleh sebuah perusahaan pengembang *software* sekaligus konsultan basis data bernama MySQL AB yang bertempat di Swedia. Waktu itu perusahaan tersebut masih bernama TcX *DataConsult* AB, dan tujuan awal dikembangkannya MySQL adalah untuk mengembangkan aplikasi berbasis *website* pada *client.*

MySQL merupakan basis data yang dikembangkan dari bahasa SQL. Ada beberapa alasan mengapa MySQL menjadi program basis data yang sangat populer dan digunakan oleh banyak orang. Alasan-alasan tersebut diantaranya sebagai berikut :

1. MySQL adalah basis data yang memiliki kecepatan yang tinggi dalam melakukan pemrosesan data, dapat diandalkan, dan mudah digunakan serta mudah dipelajari.
2. MySQL didukung banyak bahasa pemrograman seperti C, C++, Perl, Python, Java dan PHP. Pengguna dapat menggunakan bahasa pemrograman tersebut untuk berinteraksi maupun berkomunikasi dengan MySQL *server*, atau dapat juga digunakan sebagai komponen pembentuk antarmuka (*interface*) dari suatu basis data MySQL.
3. MySQL dapat melakukan koneksi dengan *client* menggunakan protokol TCP/IP, *Unix socket* (Unix), atau *Named Pipes* (NT).
4. Dalam hal menampung data, MySQL dapat menangani basis data dengan skala yang cukup besar dengan jumlah *record* mencapai lebih dari 50 juta, dapat menampung 60 ribu tabel, dan juga bisa menampung 5 milyar baris data. Selain itu, batas *index* pada tiap tabel menampung mencapai 32 *index*.
5. Dalam hal relasi antartabel pada suatu basis data, MySQL menerapkan metode yang sangat cepat, yaitu dengan menggunakan metode *one-sweep multijoin* yaitu efisiensi pengelolaan informasi yang diminta oleh user dari beberapa tabel sekaligus.

6. *Multiuser,* yaitu dalam satu basis data *server* pada MySQL dapat diakses oleh beberapa user dalam waktu yang sama tanpa mengalami konflik atau kemacetan sistem.
7. Keamanan yang dimiliki basis data MySQL dikenal baik, karena memiliki lapisan sekuritas seperti *level subnetmask*, nama *host* dan izin akses user dengan sistem perizinan yang khusus serta *password* yang dimiliki setiap user dalam bentuk terenkripsi.
8. MySQL adalah *software* basis data yang bersifat *free* atau gratis.

MySQL termasuk DBMS yang memiliki pengembangan tipe data yang cukup beraneka ragam, mulai dari pengembangan tipe data angka, karakter maupun tanggal. Berbagai tipe data yang didukung oleh MySQL untuk lebih lengkapnya dapat dilihat pada Tabel 2.5 dengan deskripsi pada masing-masing tipe data.

Tabel 2.5 Tipe data pada MySQL

| **Tipe Data** | **Deskripsi** |
|---|---|
| TinyInt | Ukuran 1 *byte*. Bilangan bulat terkecil, dengan jangkauan untuk bilangan bertanda: -128 sampai dengan 127 dan untuk yang tidak bertanda: 0 sampai dengan 255. Bilangan tak bertanda ditandai dengan kata *Unsigned*. |
| SmallInt | Ukuran 2 *byte*. Bilangan bulat dengan jangkauan untuk bilangan bertanda: -32768 sampai dengan 32767 dan untuk yang tidak bertanda: 0 sampai dengan 65535. |
| MediumInt | Ukuran 3 *byte*. Bilangan bulat dengan jangkauan untuk bilangan bertanda: -8388608 sampai dengan 8388607 dan untuk yang tidak bertanda: 0 sampai dengan 16777215. |
| Int | Ukuran 4 *byte*. Bilangan bulat dengan jangkauan untuk bilangan bertanda: -2147483648 sampai dengan 2147483647 |

| Tipe Data | Deskripsi |
|---|---|
| | dan untuk yang tidak bertanda: 0 sampai dengan 4294967295 |
| Integer | Ukuran 4 *byte*. Sinonim dari Int. |
| BigInt | Ukuran 8 *byte*. Bilangan bulat terbesar dengan jangkauan untuk bilangan bertanda: -9223372036854775808 sampai dengan 9223372036854775087 dan untuk yang tidak bertanda: 0 sampai dengan 18446744073709551615. |
| Float | Ukuran 6 *byte*. Bilangan pecahan. |
| Double | Ukuran 8 *byte*. Bilangan pecahan. |
| Double Precision | Ukuran 8 *byte*. Bilangan pecahan berpresesi ganda. |
| Real | Ukuran 8 *byte*. Sinonim dari Double. |
| Decimal(m,d) | Ukuran M *byte*. Bilangan pecahan. Misalnya, Decimal(5,2) dapat digunakan untuk menyimpan bilangan -99,99 sampai dengan 99,99. |
| Timestamp | Sebuah timestamp otomatis |
| Date | Tipe data untuk menyimpan informasi tanggal dengan menggunakan "YYYY-MM-DD" sintaksis, tapi dapat diperbaharui dengan sebuah angka atau sebuah huruf. |
| Time | Tipe data untuk menyimpan informasi jam dengan menggunakan "HH:MM:SS" sintaksis, tapi dapat diperbaharui dengan sebuah angka atau sebuah huruf. |
| DateTime | Tipe data untuk menyimpan data tanggal dan jam dengan format "YYYY-MM-DD HH:MM:SS". |
| Numeric(m,d) | Ukuran M *byte*. Sinonim dari Decimal. |
| MediumBlob, MediumText | L+3 *byte*, dengan $L<2^{24}$. Tipe teks atau Blob dengan panjang maksimum 16777215 karakter. |
| Varchar | Sebuah *variable* huruf yang dapat menyimpan 1-255 karakter. |

| Tipe Data | Deskripsi |
|---|---|
| LongBlob, LongText | L+4 *byte*, dengan $L<2^{32}$. Tipe Text atau Blob dengan panjang maksimum 4294967295 karakter |
| Enum(‘nilai1’, ’nilai2’,...) | Ukuran 1 atau 2 *byte* tergantung jumlah nilai enumerasinya (maksimum 65535 nilai) |
| Set(‘nilai1’, ’nilai2’,...) | 1, 2, 3, 4, atau 8 *byte*, tergantung jumlah anggota himpunan (maksimum 64 anggota) |

## 2.8 UML *(Unified Modelling Language)*

### 2.8.1 Sejarah UML

Sejarah UML sendiri cukup panjang. Hingga era tahun 1990, seperti telah diketahui, puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. 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 bekerjasama dengan group atau perusahaan lain yang menggunakan metodologi yang berlainan.

Metodologi yang paling banyak digunakan dalam OOAD adalah OMT *(Object modeling Technique)* dari Rumbaugh, OOAD *(Object Oriented Analysis/Design)* dari Shlaer-Mellor, Metode Booch, *Responsiblity-Driven Design/-Class/Responsibility Coad/Yourdon dan Jacobson* OOSE *(Object Oriented Software Engineering).*

### 2.8.2 Definisi UML

*Unified modeling language* adalah bahasa standar yang digunakan untuk menuliskan *blueprint* (cetak-biru) perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan artifak dari sebuah sistem perangkat lunak intensif. Artifak adalah sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa perangkat lunak.

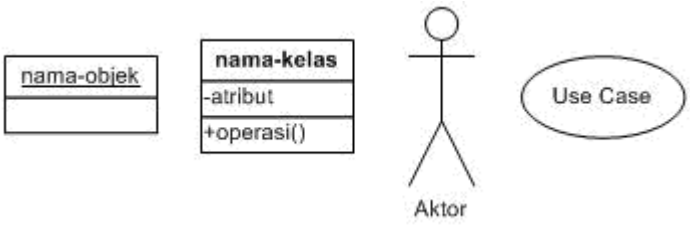
Dalam konteks ini, menentukan berarti membuat model yang tepat, tidak ambigu, dan lengkap. Biasanya, UML berfokus pada semua spesifikasi keputusan analisis, desain dan implementasi penting yang harus dibuat dalam membangun dan menerapkan sistem perangkat lunak.

UML merupakan bahasa yang sangat ekspresif, membuat semua *view* (tampilan) yang dibutuhkan dalam membangun sistem yang berskala besar. UML hanyalah sebuah bahasa dan juga hanya merupakan salah satu bagian dari metode pembangunan perangkat lunak. UML tidak tergantung pada proses, walaupun optimalnya, UML harus digunakan dalam sebuah proses yang bersifat *use-case driven*, *architecture-centric*, iteratif, dan inkremental.

*Use-case driven* berarti bahwa proses dalam *use case* digunakan sebagai artifak primer untuk membuat *behavior* yang diinginkan dalam sistem, memastikan dan memvalidasi arsitektur sistem, menguji, dan berkomunikasi antar pihak dalam proyek. *Architecture-centric* berarti bahwa proses yang berfokus pada pembangunan sebelumnya dan berdasar pada arsitektur perangkat lunak menggunakan arsitektur sistem sebagai artifak primer untuk mengonsep, membangun, mengatur, dan mengubah sistem yang sedang dibangun.

### 2.8.3 Notasi UML

UML Merupakan notasi-notasi baku untuk melakukan pemodelan visual. Pemodelan visual merupakan proses penggambaran informasi-informasi secara grafis. Notasi-notasi UML terbentuk atas kerjasama dan upaya Graddy Booch, DR. James Rumbaugh, serta Ivar Jacobson.




*Gambar 2.1 Beberapa Notasi UML*

Beberapa notasi UML yang sering digunakan antara lain ditunjukkan pada Gambar 2.1 Aktor merupakan entitas yang berada di luar sistem, bersifat eksternal. Salah satu aktor sistem adalah pengguna (*user*).

*Use case* berfokus pada perilaku sistem, misalnya *use case* "Pengaktifan modem". Kelas merupakan prototipe suatu objek sehingga memiliki atribut (variabel) dan operasi (fungsi atau prosedur).

Dalam UML terdapat beberapa jenis diagram yang dapat digunakan dalam pemodelan sistem, diantaranya adalah:

1. *Activity Diagram*. Bersifat dinamis, memodelkan fungsi-fungsi dalam suatu sistem dan menekankan pada aliran kendali antar objek.
2. *Use Case Diagram*. Bersifat statis, menggambarkan hubungan (*relation*) antara aktor dan *use case*.

3. *Sequence Diagram*. Bersifat dinamis, merupakan diagram interaksi yang menekankan pada pengiriman pesan (*message*) secara berurutan berdasarkan waktu.
4. *Class Diagram*. Bersifat statis, menggambarkan himpunan kelas-kelas, antarmuka dan relasi antar kelas.

## *2.9 Borland Delphi 7*

*Borland Delphi 7* adalah salah satu bahasa pemrograman yang bekerja dalam lingkup *Microsoft windows*. *Borland Delphi 7* dapat memanfaatkan kemampuan yang dimilikinya untuk dapat bekerja dalam lingkungan *windows* dengan baik dan. kemampuannya dapat dipakai untuk merancang program aplikasi yang berpenampilan seperti program aplikasi lainnya yang berbasis *Microsoft windows.*

*Borland Delphi 7* dapat memamfaatkan fasilitas dan kecanggihan yang ada pada sistem operasi *Microsoft Windows*. Kemampuan *Borland Delphi 7* ditunjang dengan ketersediaan komponen yang lengkap baik komponen yang umum maupun komponen khusus yang dapat di gunakan setelah diinstalasi sehingga membuat program aplikasi yang dibuat lebih menarik baik dari segi tampilan maupun kehandalanya terutama pada lingkungan system operasi *Microsoft Windows.*

Komponen standar yang ada pada *Borland Delphi 7* adalah sebagai berikut:

1. *Project*

   *Project* adalah sekumpulan modul. Jadi *project* (proyek) adalah program aplikasi itu sendiri. *Project* disimpan dalam *file* berakhiran .DPR. *File* ini

menyimpan seluruh komponen program, termasuk pilihan proyek, pilihan *Environment*, pilihan file EXE dan segala sesuatu yang berhubungan dengan proyek.

2. *Form Designer*

   *Form Designer* adalah suatu objek yang dipakai sebagai tempat bekerja program aplikasi. *form* berbentuk jendela dan dapat dibayangkan sebagai kertas atau meja kerja yang dapat dilukisi atau diletakkan ke dalamnya objek-objek lain.

3. *Object Inspector*

   *Object Inspector* adalah kotak alat yang berisi *icon-icon* untuk memasukkan objek tertentu ke dalam jendela *form*. Ketika anda mulai membuat suatu proyek , *Delphi 7* akan otomatis menyediakan *icon-icon* objek yang sering dipakai, sesuai kategori aplikasi yang anda buat, misalnya DLL dsb.

4. *Properties*

   *Properties* digunakan untuk menentukan konfigurasi suatu objek. Suatu objek biasanya mempunyai beberapa properti yang dapat diatur langsung dari jendela *properties* atau lewat kode program. Setting properti akan menentukan cara kerja dari objek yang bersangkutan saat program aplikasi dijalankan, misalnya menentukan warna objek, bingkai objek, pengambilan data dan lain-lain.

5. *Code Editor*

   *Code Editor* adalah serangkaian tulisan perintah yang akan dilaksanakan jika suatu objek dijalankan. Kode program ini akan mengontrol dan menentukan jalannya suatu objek.

6. *Event*

   *Event* adalah peristiwa atau kejadian yang diterima oleh suatu objek, misalnya klik kanan tunjuk dan lain-lain. *Event* yang diterima objek akan memicu *Borland Delphi 7* menjalankan kode program yang ada didalamnya.

7. *Metoda*

   *Metoda* adalah suatu set perintah seperti halnya fungsi dan prosedur, tetapi sudah tersedia didalam suatu objek. Seperti halnya *property* (yang juga terdapat pada suatu objek), suatu metoda dapat dipanggil dengan menyebut nama objek diikuti tanda titik dan nama metodanya.

8. *Module*

   *Module* dapat disejajarkan dengan *form*, tetapi tidak mengandung objek dan bentuk standar. *Module* dapat berisi beberapa kode program atau prosedur yang dapat digunakan dalam program aplikasi. Sebenarnya suatu form juga adalah sebuah *module* (disebut sebagai *form module*), tetapi mempunyai objek berupa *form*.