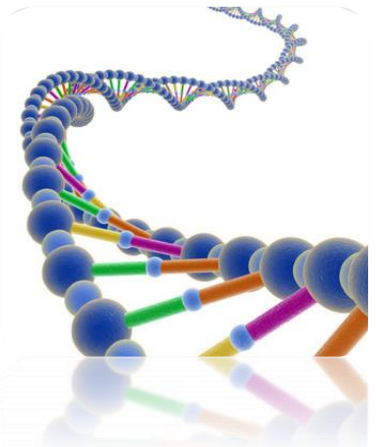
BAGIAN II
PEMODELAN STRUKTURAL



POKOK BAHASAN

- Class
- Relationships
- Class Diagram
- Object Diagram
- Component Diagram
- Package
- Deployment Diagram
- Mekanisme Umum

BAB 3

CLASS

1. Konsep Model

Memodelkan suatu sistem dengan pendekatan berorientasi objek adalah diawali dengan mengidentifikasi objek-objek yang memiliki peranan penting dalam sistem. Objek ini dapat berupa objek konkrit yaitu; benda, manusia, unit kerja, dan objek abstrak; konsep, proses, event dan hal-hal lainya.

Langkah selanjutnya adalah mengidentifikasi atribut-atribut yang melekat pada masing-masing objek. Atribut-atribut ini adalah data atau nilai yang mencerminkan karakteristik dari objek. Misalkan objek mobil akan memiliki atribut; merk, model, warna, tahun keluaran, dan kapasitas mesin.

Gambar 3.1. Identifikasi Atribut dan Operasi Pada Objek

Kemudian setelah mengidentifikasi atribut, proses selanjutnya adalah mengidentifikasi operasi-operasi yang akan menjadi tingkah laku dari objek-objek. Operasi-operasi ini akan memainkan peranan di dalam kosakata sistem nantinya. Untuk mengidentifikasi operasi-operasi pada suatu objek tidaklah mudah, tergantung pada fungsi dan peranan objek tersebut. Apa yang anda inginkan objek tersebut lakukan. Misalkan sebuah mobil tentunya akan memiliki fungsi move forward, move backward, move left, move right, dan break.

Dalam konteks pemodelan proses identifikasi ini disebut abstraksi tingkat atas. Dari objek yang sudah teridentifikasi atribut dan operasinya maka dapat ditransformasikan menjadi sebuah class dalam

pemodelan. Class ini akan menjadi abstraksi tingkat menengah.

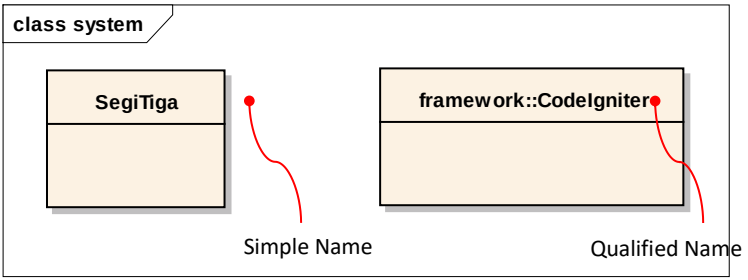
2. Notasi

2.1. Class

Class atau kelas mempunyai beberapa definisi, diantaranya; kelas adalah deskripsi dari satu set objek-objek yang berbagi atribut-atribut, operasi-operasi, relasi-relasi, dan semantik-semantik yang sama (Rumbaugh, Jacobson, & Booch, The Unified Modeling Language User Guide Second Edition, 2005). Kelas bukanlah objek yang individual, tetapi lebih kepada representasi seluruh set objek-objek.

2.2. Penamaan Kelas

Setiap kelas harus memiliki nama yang membedakannya dari kelas-kelas yang lain. Nama kelas adalah bersifat tekstual dan selalu diawali dengan huruf besar. Jika terdiri dari dua suku kata maka harus disambung penulisannya tidak boleh menggunakan spasi dan suku kata berikutnya juga diawali huruf besar. Penamaan kelas dapat berbentuk nama sederhana, atau nama yang *qualified* yaitu diawali dengan nama *package* asal kelas tersebut.



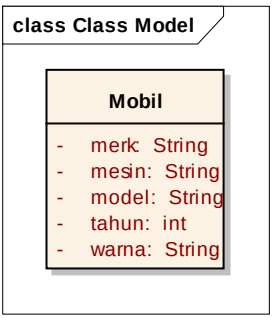
Gambar 3.2. Penamaan Kelas

2.3. Attribute

Attribute (baca: atribut) adalah properti yang memiliki nama dari suatu kelas yang menjadi karakteristik dari kelas tersebut. Sebuah kelas mungkin memiliki banyak atribut-atribut atau tidak memiliki atribut sama sekali. Atribut ini merepresentasikan properti dari hal yang sedang anda modelkan. Dari abstraksi tingkat atas sebelumnya sudah dicontohkan misalkan kelas mobil akan memiliki atribut; merk, model, warna, tahun keluaran, dan kapasitas mesin, yang jika dimodelkan sebagai kelasakan menjadi seperti gambar berikut.

Sebuah atribut dalam kelas haruslah memiliki type data sesuai dengan bahasa pemrograman yang akan mengimplementasikan kelas tersebut. Nama atribut biasanya diawali dengan huruf kecil dan suku kata berikutnya diawali dengan huruf besar. Atribut juga

memiliki jangkauan visibility, baik itu bersifat private (-), public (+), protected (#), dan package (~).



Gambar 3.3. Kelas Mobil Dengan Atribut

Implementasi kode:

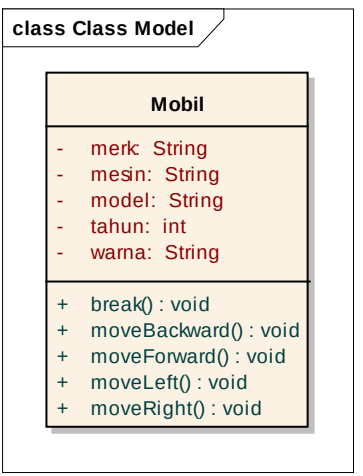
```
public class Mobil {

        private String merk;
        private String mesin;
        private String model;
        private int tahun;
        private String warna;
}
```

2.4. Operation

Operation (baca: operasi) adalah implementasi dari layanan yang dapat diminta dari objek manapun dari kelas. Dengan kata lain operasi adalah perilaku dari

objek yang mempengaruhi status dari objek. Sebuah kelas dapat memiliki beberapa operasi atau tanpa operasi sama sekali. Dalam konteks pemrograman operasi pada dasarnya adalah suatu fungsi atau prosedur yang dimiliki objek terkait. Misalkan seperti contoh diatas sebuah objek mobil dalam abstraksi tingkat atas dapat memiliki operasi; `moveForward`, `moveBackward`, `moveLeft`, `moveRight`, dan `break`.

Sebuah operasi dapat mengembalikan nilai sesuai dengan type data yang kita tentukan dan tentunya relevan terhadap bahasa pemrograman yang digunakan atau tidak mengembalikan nilai sama sekali. Operasi juga memiliki jangkauan visibility sama seperti atribut.



Gambar 3.4. Kelas Mobil Dengan Operasi

Implementasi kode:

```
public class Mobil {

        private String merk;
        private String mesin;
        private String model;
        private int tahun;
        private String warna;

        public Mobil(){
        }

        public void finalize() throws Throwable {

        }
        public void break(){

        }

        public void moveBackward(){

        }

        public void moveForward(){

        }

        public void moveLeft(){

        }

        public void moveRight(){

        }
}
```

2.5. Visibility

Visibility (baca: visibilitas) adalah spesifikasi yang dapat dimiliki oleh kelas, atribut, dan operasi, yang

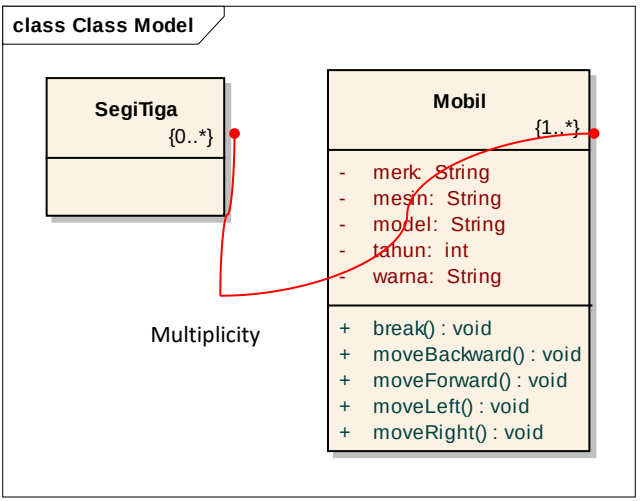
menentukan hak akses atau jangkauan atribut dan operasi tersebut. Visibilitas menentukan tingkatan akses atribut dan operasi. Ada empat macam visibilitas yaitu:

1. (-) private, hanya kelas yang bersangkutan yang dapat menggunakan fitur tersebut.
2. (+) public, kelas-kelas diluar kelas yang bersangkutan dapat mengakses dan menggunakannya.
3. (#) protected, setiap kelas turunannya dapat mengakses dan menggunakannya.
4. (~) package, hanya kelas yang berada dalam satu package yang sama yang dapat mengakses dan menggunakannya.

2.6. Multiplicity

Ketika anda menggunakan suatu kelas cukup beralasan mengasumsikan bahwa mungkin ada beberapa instansiasi dari kelas tersebut. Kadangkala mungkin anda perlu membatasi berapa banyak kelas tersebut dapat di-instansiasi misalkan null instansiasi, minimal satu, atau banyak instansiasi. Jumlah instansiasi yang dapat dimiliki suatu kelas itulah yang disebut dengan *multiplicity* (baca: multiplisitas). Multiplisitas adalah spesifikasi jangkauan kardinalitas yang diijinkan yang dimiliki suatu entitas.

Di UML anda dapat menspesfikasikan multiplisitas suatu kelas dengan menuliskan ekspresi multiplisitas disudut kanan atas icon kelas.



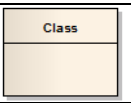
Gambar 3.5. Contoh Multiplicity Suatu Kelas

2.7. Classifier

Ketika anda melakukan pemodelan, anda akan menemukan abstraksi-abstraksi dari hal-hal di dunia nyata dan hal-hal di dalam pemodelan. Sebagai contoh ketika anda membangun sistem e-commerce, kosakata yang ada dalam proyek anda sepertinya akan melibatkan kelas Customer (yang merepresentasikan orang yang memesan produk) dan sebuah kelas Order (abstraksi dari pencatatan

order). Masing-masing dari abstraksi-abstraksi ini akan memiliki instansiasi. Ada beberapa hal dalam UML yang tidak memiliki instansiasi misalnya, package. Secara umum, semua elemen-elemen yang bisa memiliki instansiasi disebut *classifier* (baca: pengklasifikasi). Sebuah pengklasifikasi memodelkan sebuah konsep tersendiri yang menjelaskan hal-hal atau objek-objek yang memiliki identitas, status, tingkah laku, hubungan, dan struktur internal yang bersifat opsional.

Tabel III.1. Macam-macam Classifier

Classifier	Fungsi	Notasi
Actor	Pengguna dari luar sistem	
Artifact	Bagian yang bersifat fisik (file) dari sistem informasi	
Class	Sebuah konsep dari sistem yang dimodelkan	
Collaboration	Sebuah hubungan tertentu antara objek-objek yang memainkan peranan	
Component	Suatu bagian dari sistem yang bersifat modular dengan