Mata Kuliah

ANALISIS & DESAIN SISTEM (ADS)

System Analysis & Design

Dosen:
Budi Nugroho, S.Kom, M.Kom

Materi 8

UML: Class Diagram

Program Studi S1 Informatika

Fakultas Ilmu Komputer UPN "Veteran" Jawa Timur

TA 2019/2020 Semester Genap

Class Diagram

- Menggambarkan hubungan statis antar class dalam sistem
  - ✓ Struktur logis dari class-class yang membangun sistem (logical Static View dalam UML).

- Class Diagram dan Interaction Diagram :
  - ✓ Class Diagram dibuat setelah Interaction Diagram (recommended).
    - ⇘ Interaction Diagram dibuat untuk setiap fungsi sistem (use case)
      → Class Diagram dibuat berdasarkan Interaction Diagram pada setiap use case.
    - ⇘ Mengidentifikasi class-class dari obyek-obyek pada Interaction Diagram dan bagaimana pola interaksi antar class tersebut.
    - ⇘ Kerugian : waktu pengerjaan lebih lama.
      Keuntungan : tahapan pengerjaan lebih sistematis → hasil lebih optimal.
  - ✓ Class Diagram dibuat sebelum Interaction Diagram.
    - ⇘ Memanfaatkan class-class yang ada pada Class Diagram untuk membuat obyek-obyek dan relasi dinamisnya (urutan prosedur atau pola interaksi antar elemen untuk setiap fungsi sistem) pada Interaction Diagram.
    - ⇘ Umumnya tidak menggunakan klasifikasi class (boundary, control, entity) pada pemodelan UML yang direkomendasikan.
    - ⇘ Kerugian : tahapan pengerjaan kurang sistematis → hasil kurang optimal.
      Keuntungan : waktu pengerjaan lebih cepat.

Relasi

- Menjelaskan bagaimana antar class saling berkaitan.
  - ☞ Bagaimana sebuah class mengetahui atribut, operasi, dan hubungan dengan class lain.

- Ketika sebuah class mengirimkan message ke class lain pada Interaction Diagram, maka harus terdapat relasi pada kedua class tersebut.

- Jenis Relasi :
  - ✓ Relasi Asosiasi
  - ✓ Relasi Dependensi
  - ✓ Relasi Agregasi
  - ✓ Relasi Generalisasi
  - ✓ Relasi Realisasi

Relasi Asosiasi

- Menggambarkan suatu class yang mengirimkan pesan ke class lain.
- Memungkinkan suatu class mengetahui atribut & operasi yang mempunyai visibilitas public dari class lain.
- Ada 3 tipe :
  1. Relasi Asosiasi Bidirectional
  2. Relasi Asosiasi Unidirectional
  3. Relasi Asosiasi Reflexive (Unidirectional Reflexive)

Relasi Asosiasi

1. Relasi Asosiasi Bidirectional

- ✓ Relasi asosiasi 2 arah
- ✓ Kedua class bisa saling mengirimkan pesan.
- ✓ Model : link / anak panah 2 arah atau garis lurus.
- ✓ Pengkodean : kedua class memiliki atribut dengan tipe data dari class lainnya.


```
public class A
{       …
        public B theB;
        …
}

public class B
{       …
        public A theA;
        …
}
```

Relasi Asosiasi

2. Relasi Asosiasi Unidirectional

- ✓ Relasi asosiasi 1 arah
- ✓ Sebuah class bisa mengirim pesan ke class lain, tidak sebaliknya.
- ✓ Model : link / anak panah 1 arah.
- ✓ Pengkodean : class pengirim pesan memiliki atribut dengan tipe data class lain.

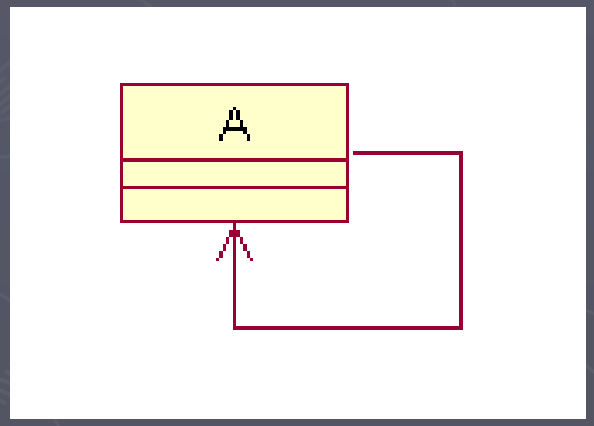

```
public class A
{       …
        public B theB;
        …
}

public class B
{       …
        …
}
```

Relasi Asosiasi

3. Relasi Asosiasi Reflexive (Unidirectional Reflexive)

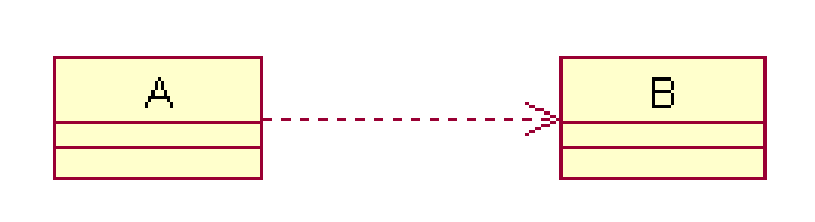
- ✓ Relasi asosiasi unidirectional dengan dirinya sendiri.
- ✓ Sebuah class bisa mengirim pesan dirinya sendiri.
- ✓ Model : link / anak panah 1 arah (mengarah ke class itu sendiri).
- ✓ Pengkodean : class memiliki atribut dengan tipe data dirinya sendiri.




```
public class A
{       …
        public A theA;
        …
}
```

Relasi Dependensi

- Menggambarkan suatu class yang mengacu ke class lain
  - ✓ Perubahan spesifikasi dalam class yang diacu mempengaruhi class pengacunya.
  - ✓ Model : link / anak panah putus-putus 1 arah.
  - ✓ Pengkodean : Suatu class menjadi parameter dari suatu method pada class lain.

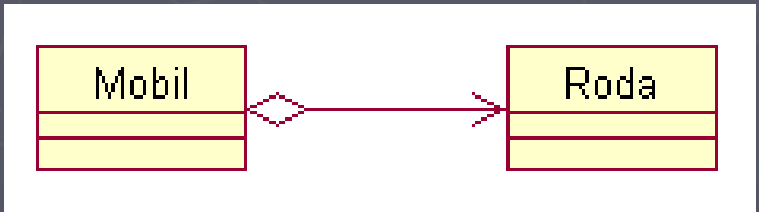



```
public class A
{       …
        public fungsiX(B theB)
        {
            …
        }
        …
}

public class B
{       …
        …
}
```

Relasi Agregasi

- Relasi antara suatu class dengan class lain yang secara konseptual / semantic menjadi bagian pembentuknya (Relasi antara "keseluruhan" dengan "bagian".)
  - ✓ Secara teknis, implementasi pengkodeannya sama dengan Relasi Asosiasi Unidirectional.

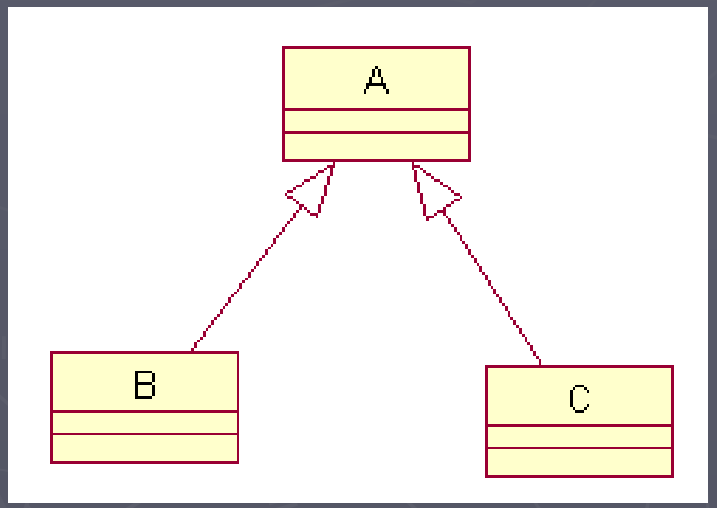



```
public class Mobil
{       …
        public Roda theRoda;
        …
}

public class Roda
{       …
        …
}
```

Relasi Generalisasi

- Relasi yang menggambarkan pewarisan class (inheritance).



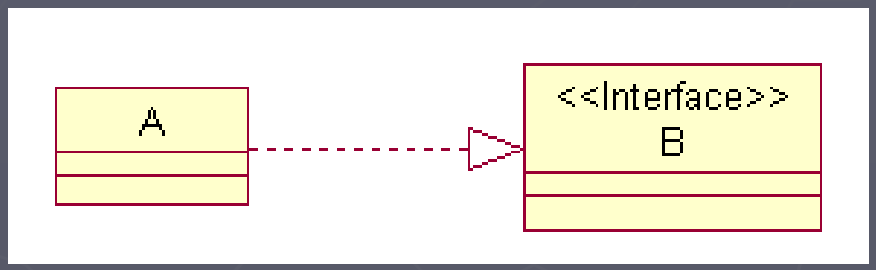

```
public class A
{      …
       …
}

class B extends A
{      …
       …
}

class C extends A
{      …
       …
}
```

Relasi Realisasi

- Relasi antara interface dan class implementasinya.




```
class A implements B
{       …
        …
}
```

Identifikasi Relasi

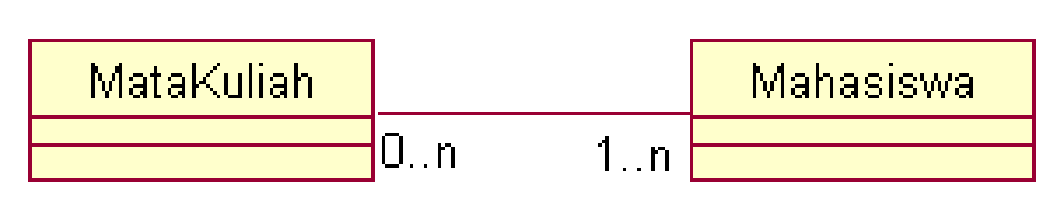
- Cek Interaction Diagram.
  - Jika class A mengirimkan pesan ke B, maka relasi yang diperlukan adalah Asosiasi atau Dependensi.
  - Jika class A dan class B saling mengirimkan pesan, maka relasi yang diperlukan adalah Asosiasi Bidirectional.

- Analisa antar class untuk menemukan hubungan konseptual / semantic "keseluruhan" dan "bagian".
  - Relasi yang diperlukan adalah Agregasi → implementasi teknisnya seperti Relasi Asosiasi Unidirectional.

- Analisa antar class untuk mencari kesamaan sifat dan beberapa sifat lain yang berbeda.
  - Relasi yang diperlukan adalah generalisasi.

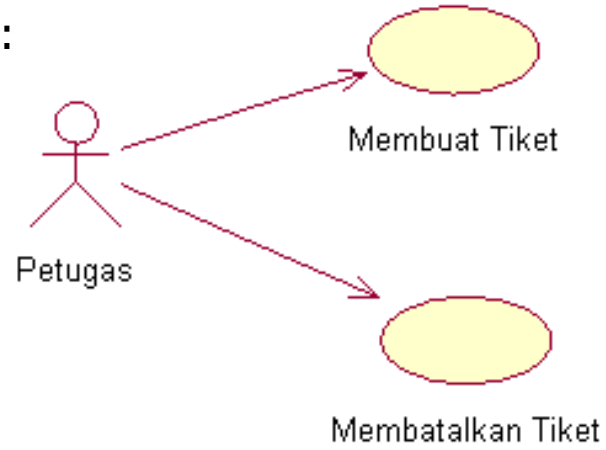
Multiplicity

- Menjelaskan berapa banyak obyek dari suatu class yang memiliki relasi ke obyek tunggal dari class lainnya dalam satu waktu.

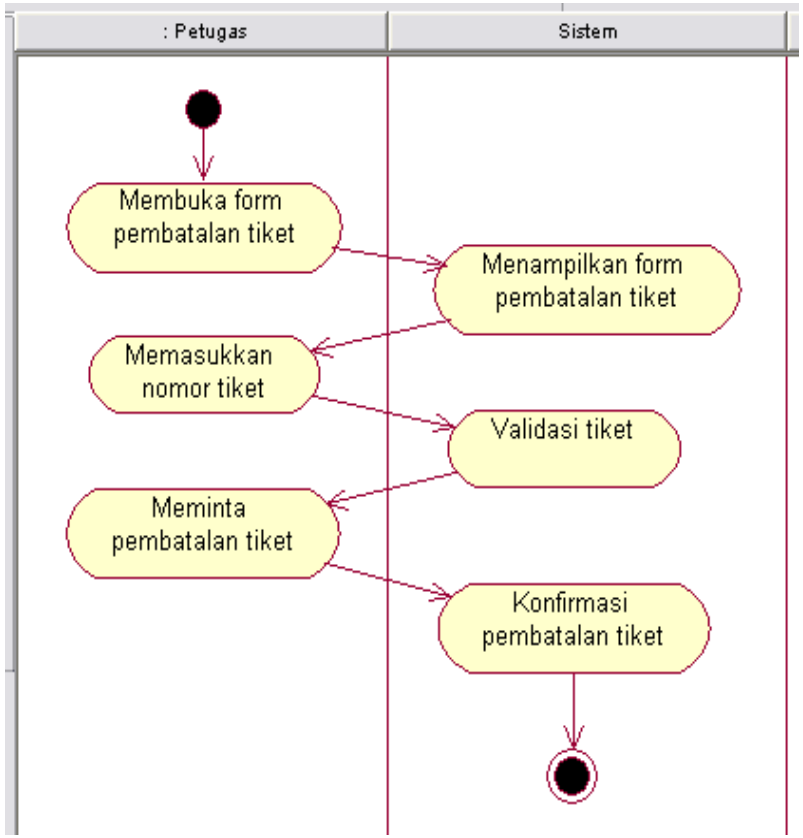


- Contoh diatas (antara 2 entity class) :
  - Nama peranan relasi : MataKuliah mempunyai Mahasiswa atau Mahasiswa mengambil MataKuliah.
  - Setiap Mahasiswa dapat mengambil 0 (tidak ada yang diambil) atau n (banyak) MataKuliah dan setiap MataKuliah dapat mempunyai 1 atau banyak Mahasiswa.

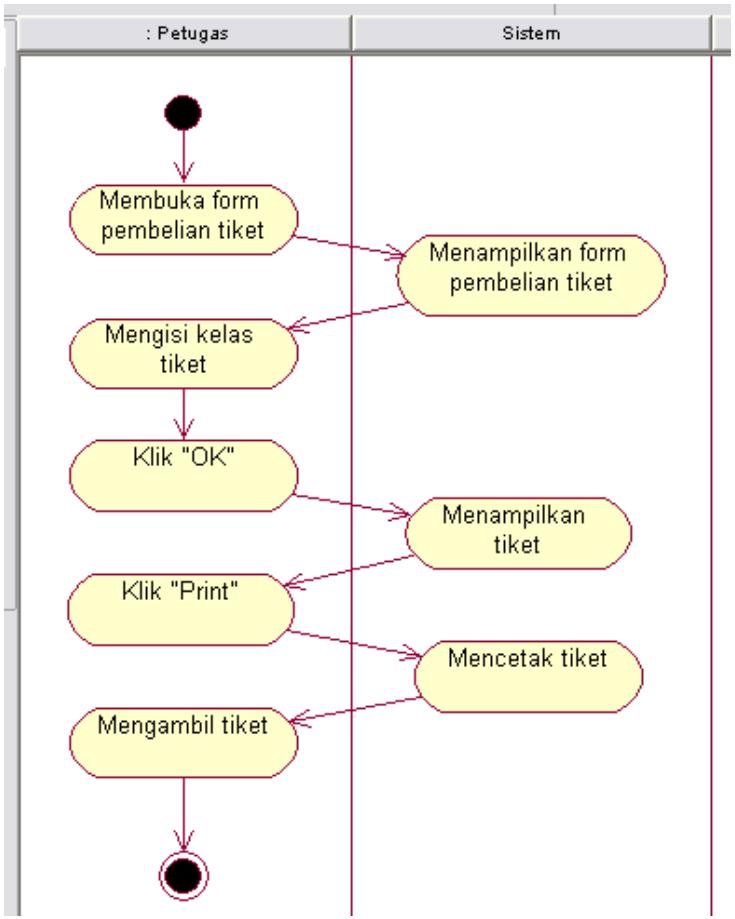
Use Case Diagram :
Pembelian Tiket



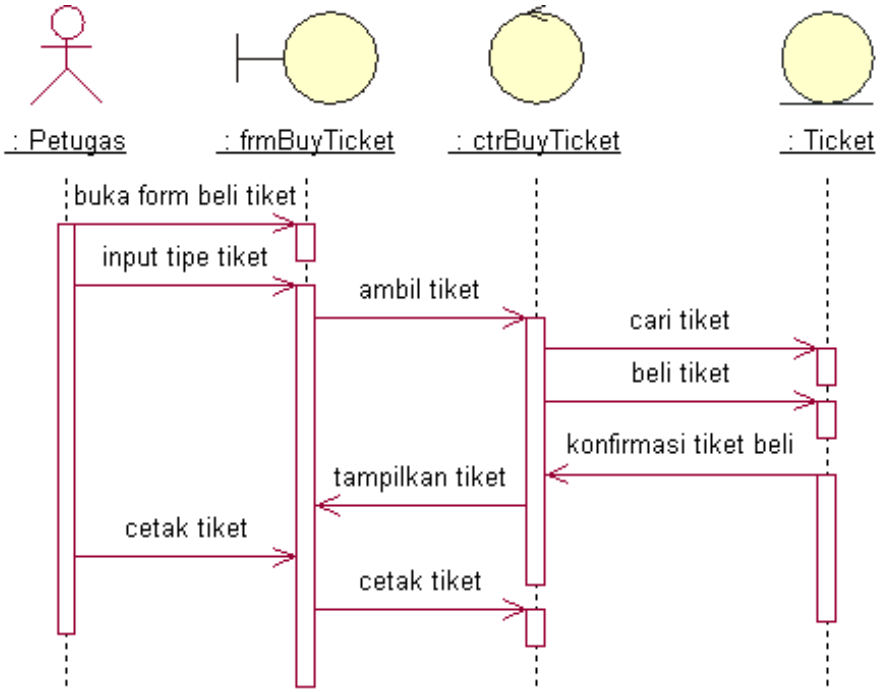
Activity Diagram : Membatalkan Tiket



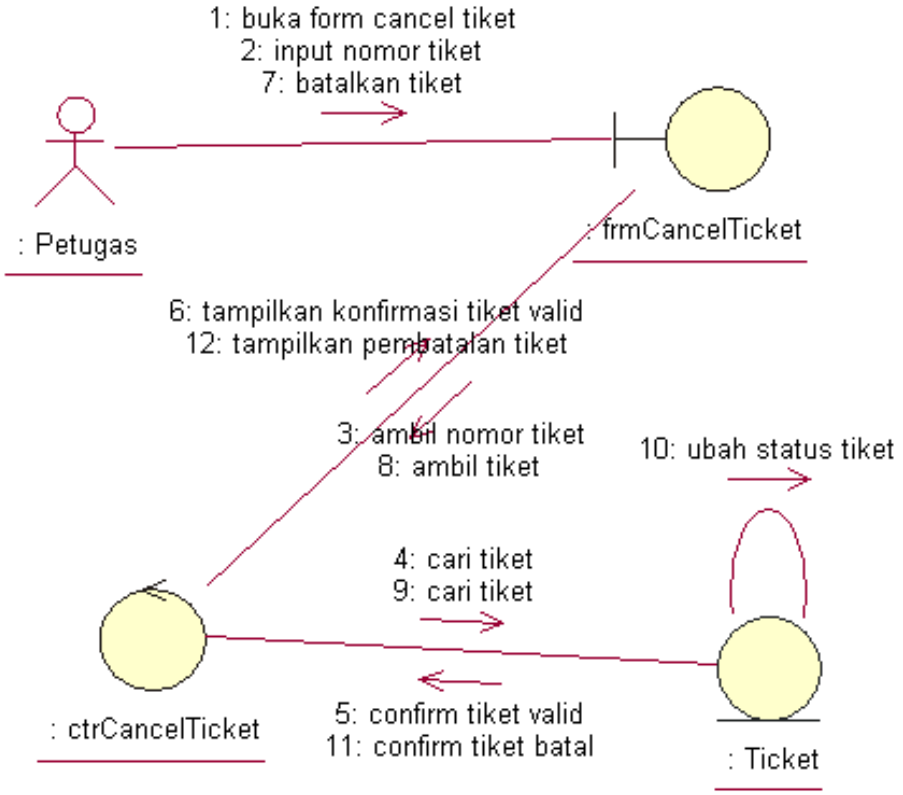
Activity Diagram : Membuat Tiket



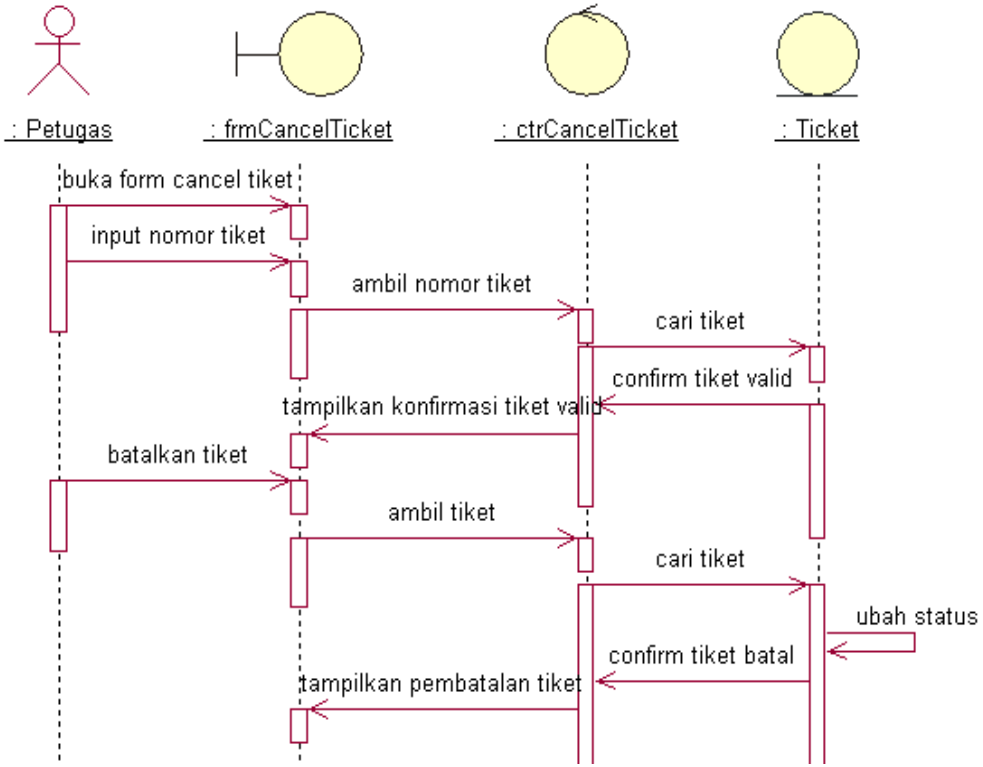
Sequence Diagram : Membuat Tiket



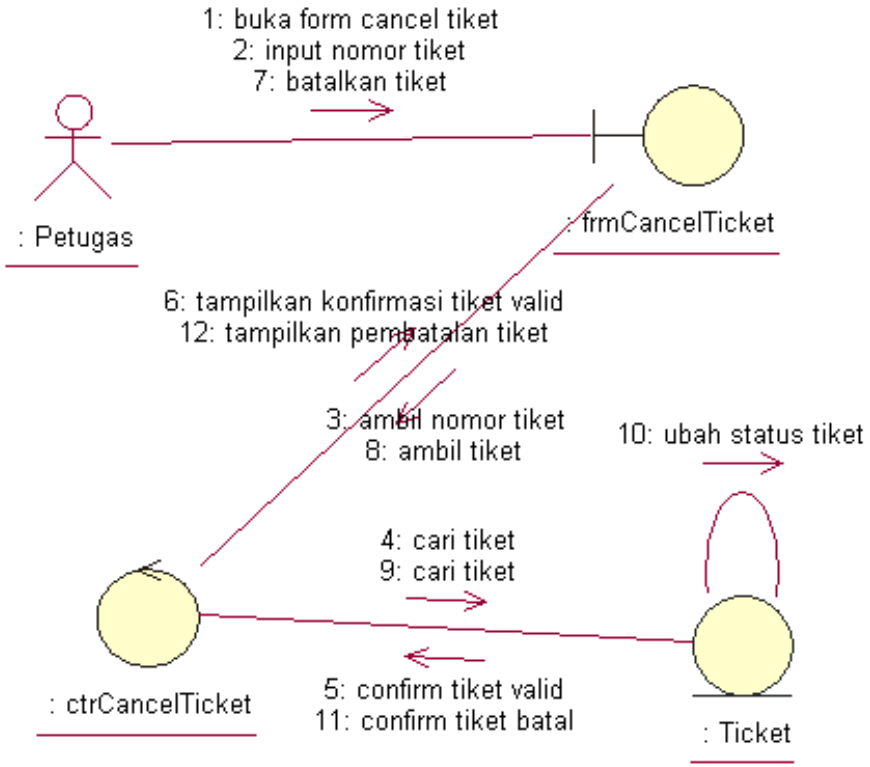
Collaboration Diagram : Membuat Tiket



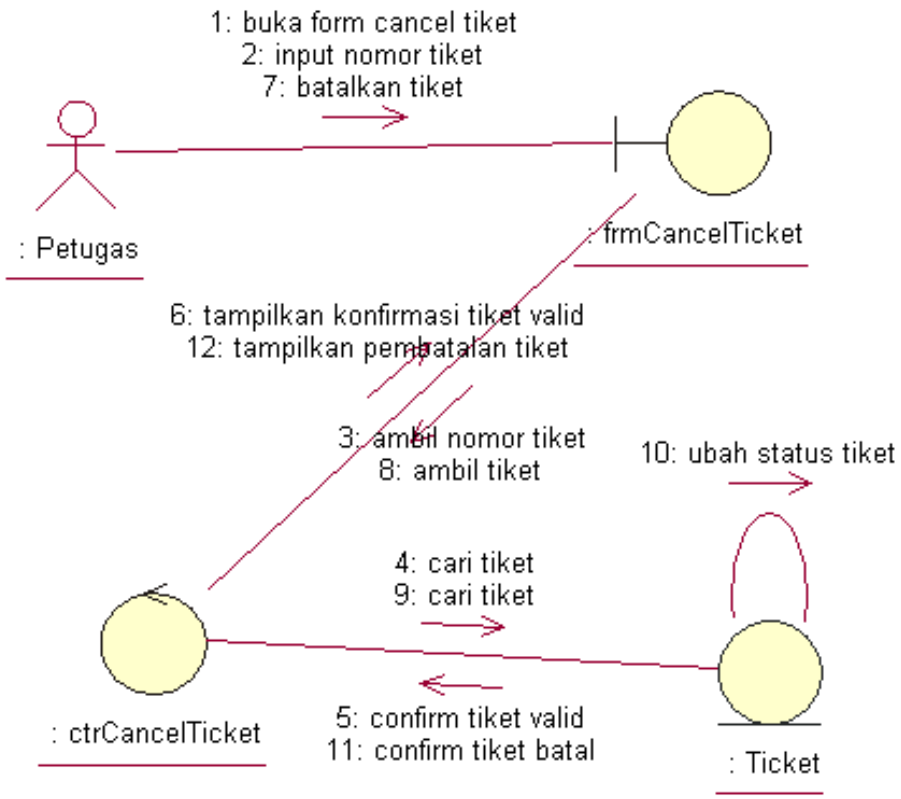
Sequence Diagram : Membatalkan Tiket



Collaboration Diagram : Membatalkan Tiket

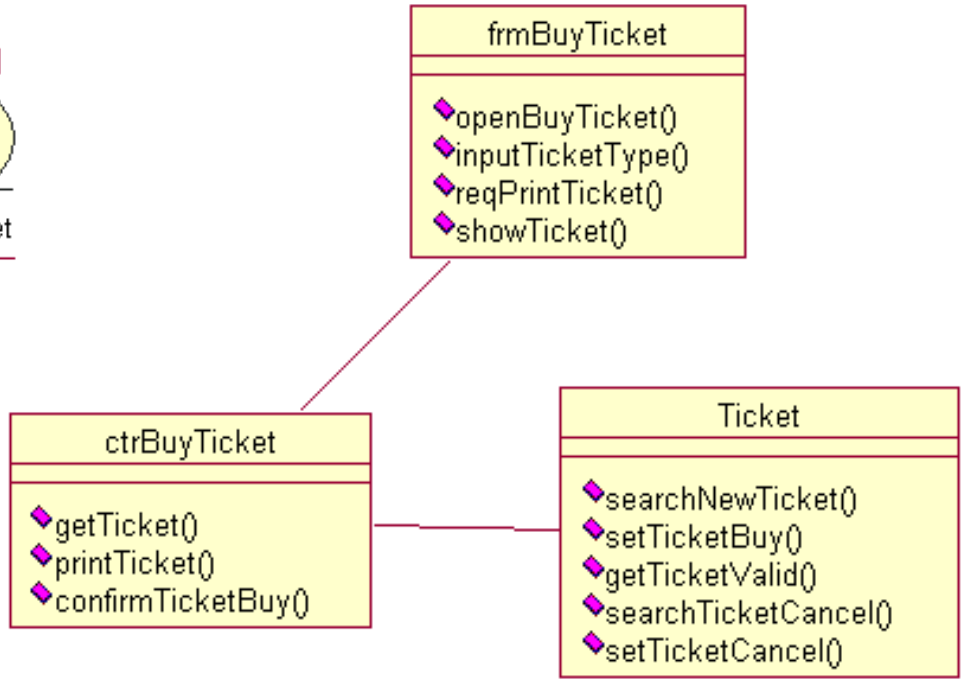


Collaboration Diagram : Membuat Tiket

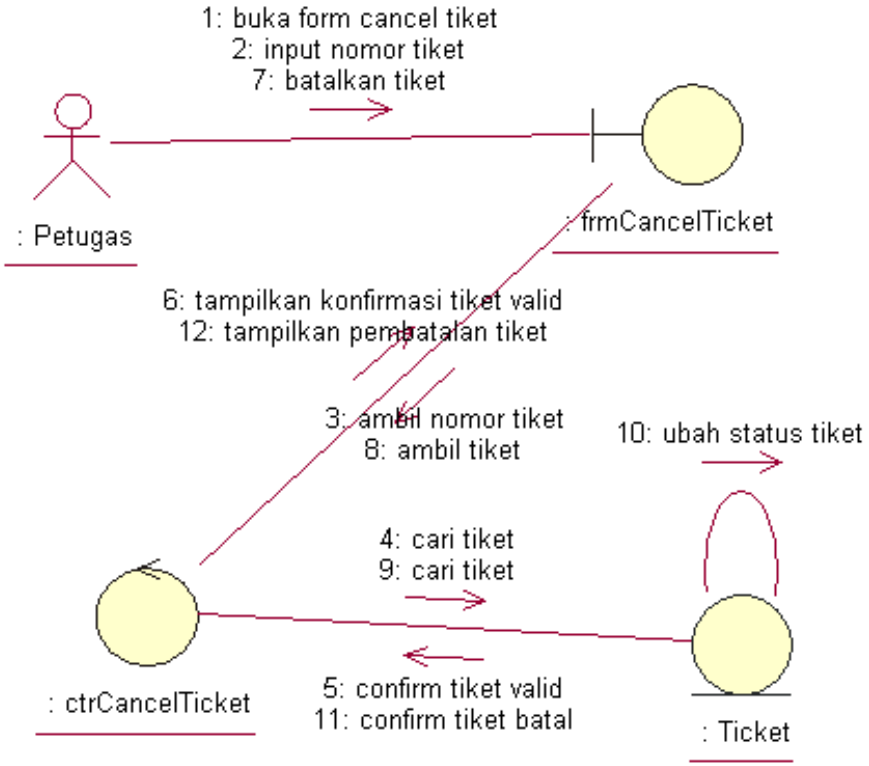


- ✓ Messages pada Interaction Diagram dipetakan (mapping) menjadi Operasi / Method Class pada Class Diagram.

Class Diagram : Membuat Tiket

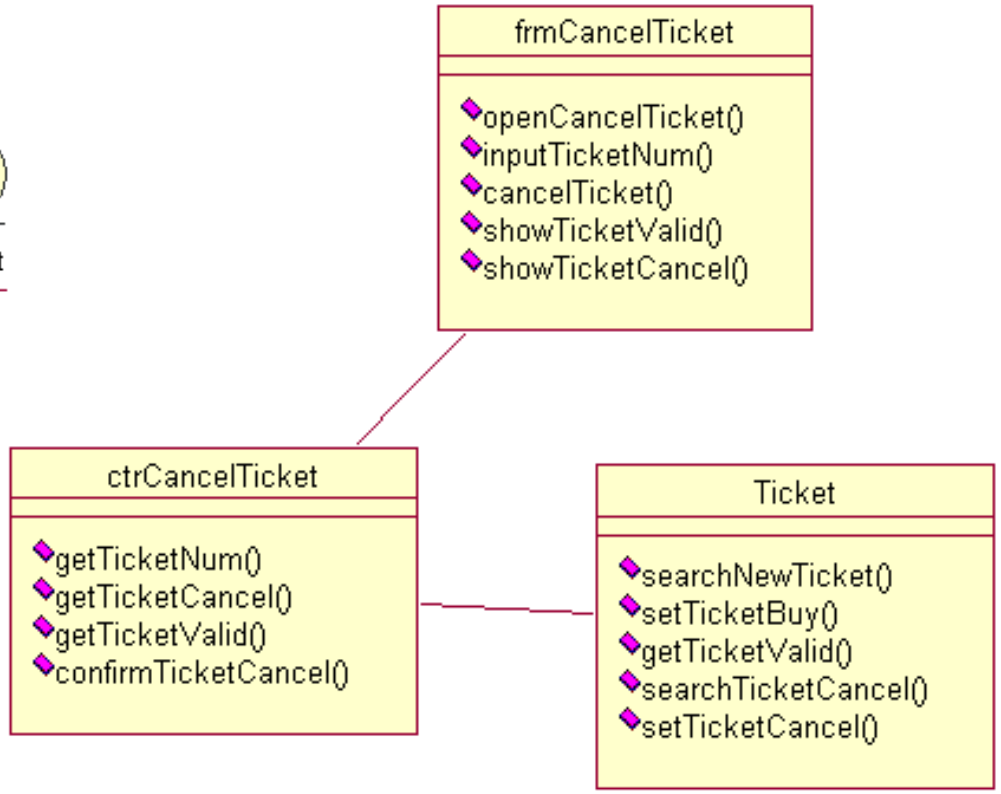


Collaboration Diagram : Membatalkan Tiket

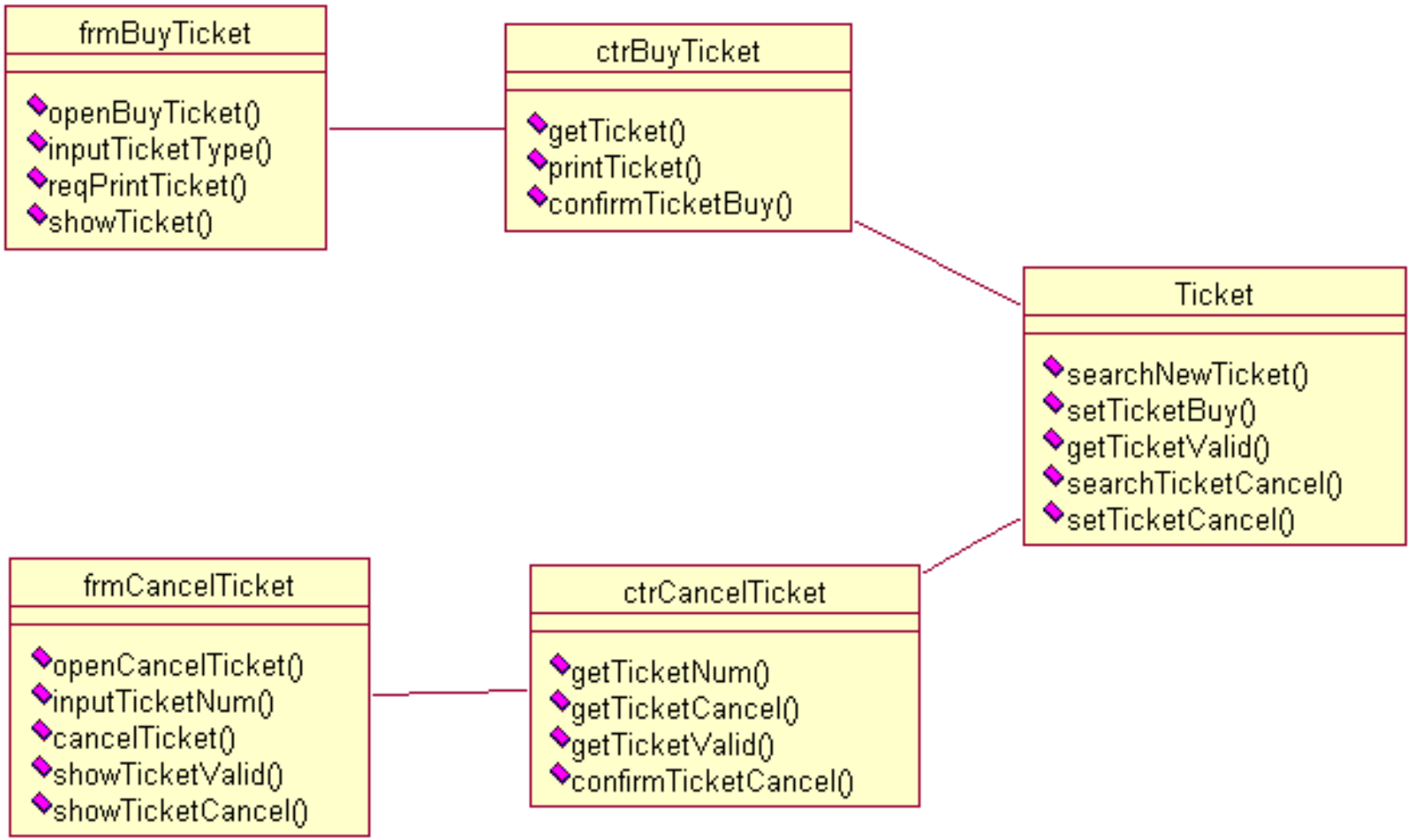


- ✓ Messages pada Interaction Diagram dipetakan (mapping) menjadi Operasi / Method Class pada Class Diagram.

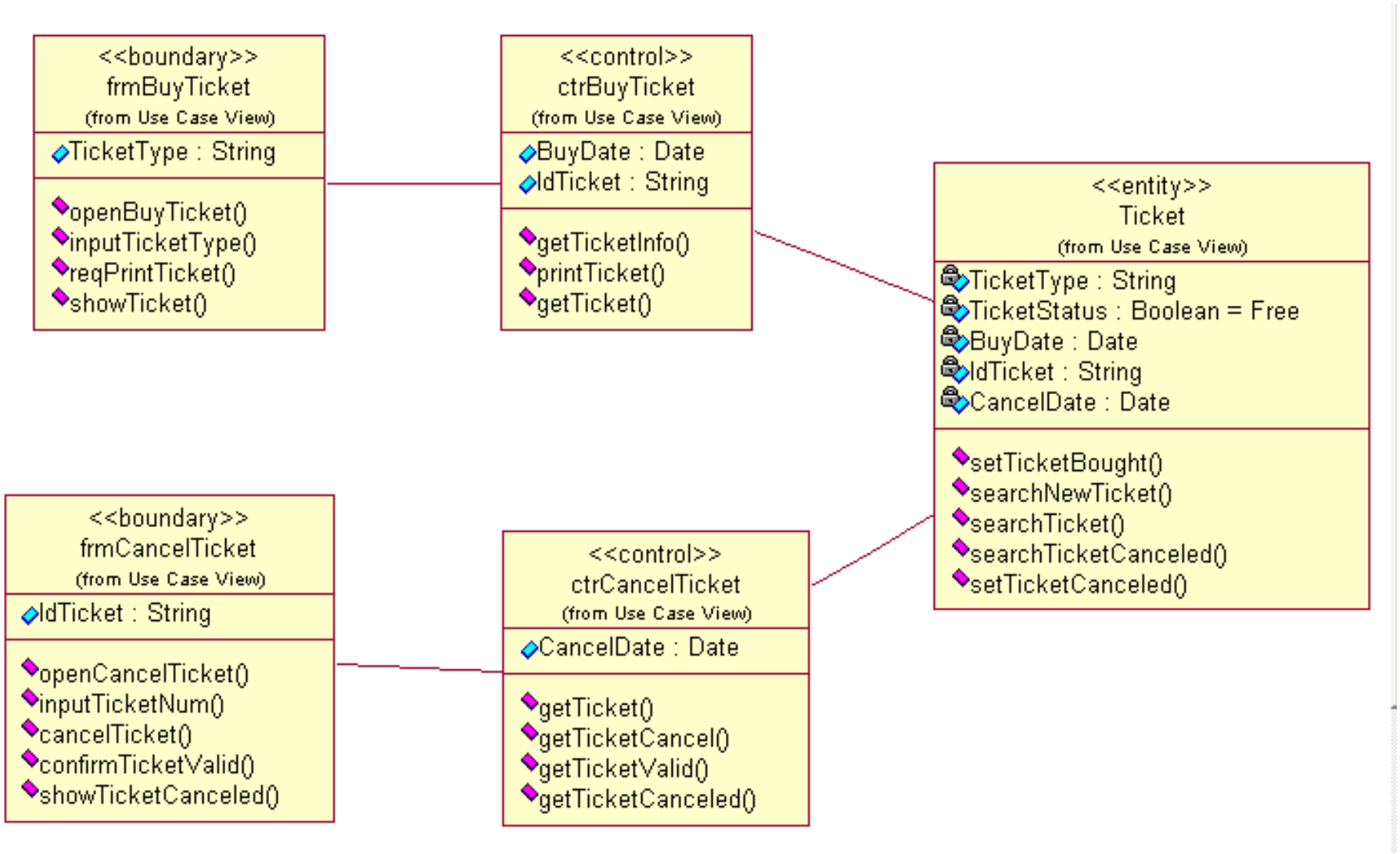
Class Diagram : Membatalkan Tiket



Class Diagram : Pembelian Tiket



Class Diagram : Pembelian Tiket



Thank You!