

Class Diagram

Class

- Pengertian dari class sama dengan class pada OOP.
- Merupakan sebuah pengelompokan atau abstraksi dari *attribute* dan operasi / *behaviour (method)* yang nantinya direalisasikan dalam bentuk *object*.
- Nama pada suatu *class* berbeda dengan nama *class* lainnya.

Notasi *Class*

<Nama_Class>
<Attribute 1> <Attribute 2> ... <Attribute N>
<Method 1> <Method 2> ... <Method N>

Notasi *Modifier*

- + : public
- - : private
- # : protected
- ~ : package

Contoh Notasi *Class* (1)

Dokter
<ul style="list-style-type: none">- kodeDokter : String- namaDokter : String- spesialisasi : String []- tglLahir : Date
<ul style="list-style-type: none">- setKodeDokter(String) : void- getKodeDokter() : String- setNamaDokter(String) : void- getNamaDokter() : String- setSpesialisasi(String []) : void- getSpesialisasi() : String []- getTglLahir(Date) : void- setTglLahir() : Date

Contoh Notasi *Class* (2)

Pegawai
<ul style="list-style-type: none">- noPegawai : String- namaPegawai : String- alamat : String- tglLahir : Date
<ul style="list-style-type: none">- setNoPegawai(String) : void- getNoPegawai() : String- setNamaPegawai(String) : void- getNamaPegawai() : String- setAlamat(String) : void- getAlamat() : String- setTglLahir(Date) : void- getTglLahir() Date- hitungGaji() : double

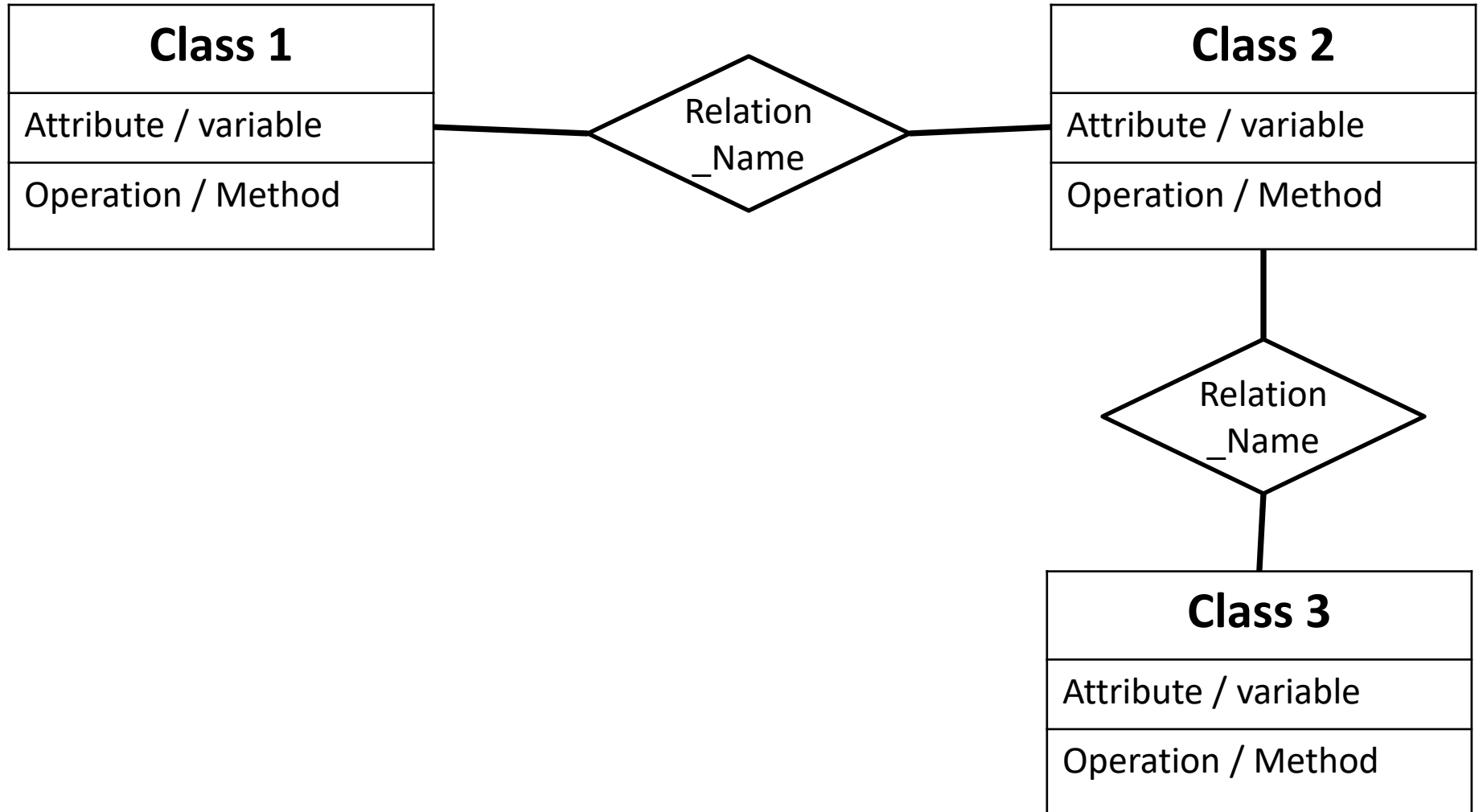
Contoh Notasi *Class* (3)

Store
<ul style="list-style-type: none">- storeCode : String- storeName : String- address : String- shortName : String
<ul style="list-style-type: none">- void setStoreCode(String)- String getStoreCode()- void setStoreName(String)- String getStoreName()- void setAddress(String)- String getAddress()- void setShortName(String)- String setString()- Object doSale(ArryList [])





Class Diagram

- Merupakan sebuah diagram *static structure* yang mendeskripsikan struktur sebuah sistem, dengan menampilkan *class* dan *relationship*.
- Seluruh metode analisis dan desain berbasis OOP menyediakan cara untuk menggambarkan *static structure*.
- Mencakup elemen : *class, relationship*.

Notasi *Class Diagram*



Class Relationship

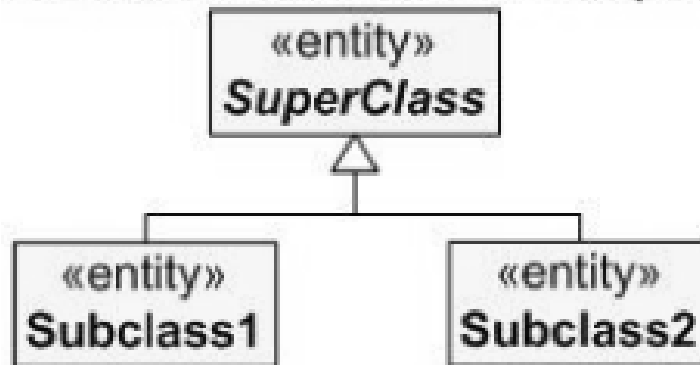
Relationship	Description	Arrow	Description
Dependency	“uses a”		One class uses another
Association	“has a”		One class retains relationship to the other over an extended period of time. The lifeline of the two is not tied.
Aggregation	“owns a”		Ownership & some sort of relationship between the two life lines.
Composition	“is part of”		The two life lines is (nearly) always linked.

Relation Type

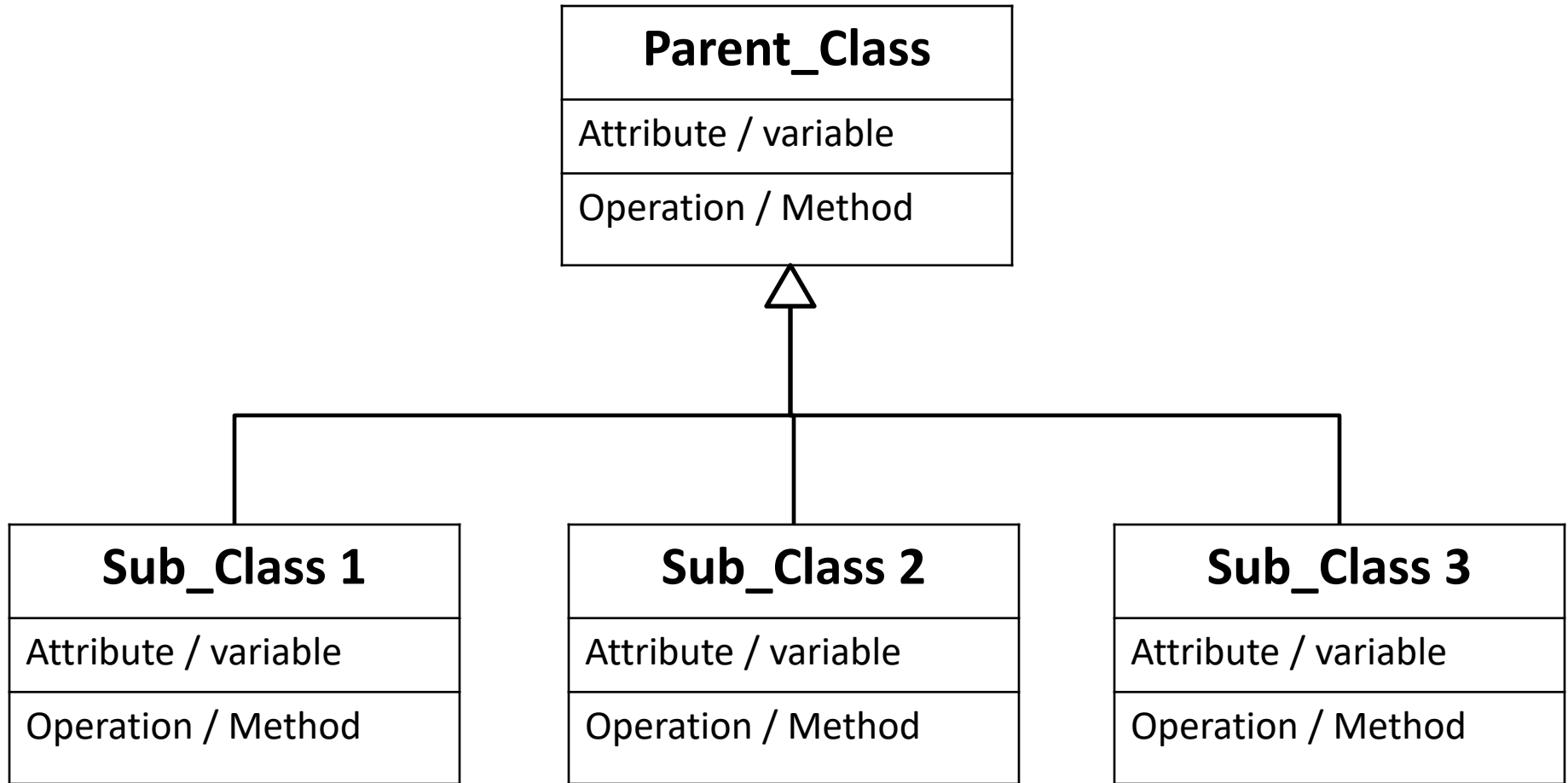
- Inheritance (or Generalization)
- Association
 - Simple association
 - Aggregation
 - Composition
 - Dependency

Notasi *Generalization* Pada *Class Diagram*

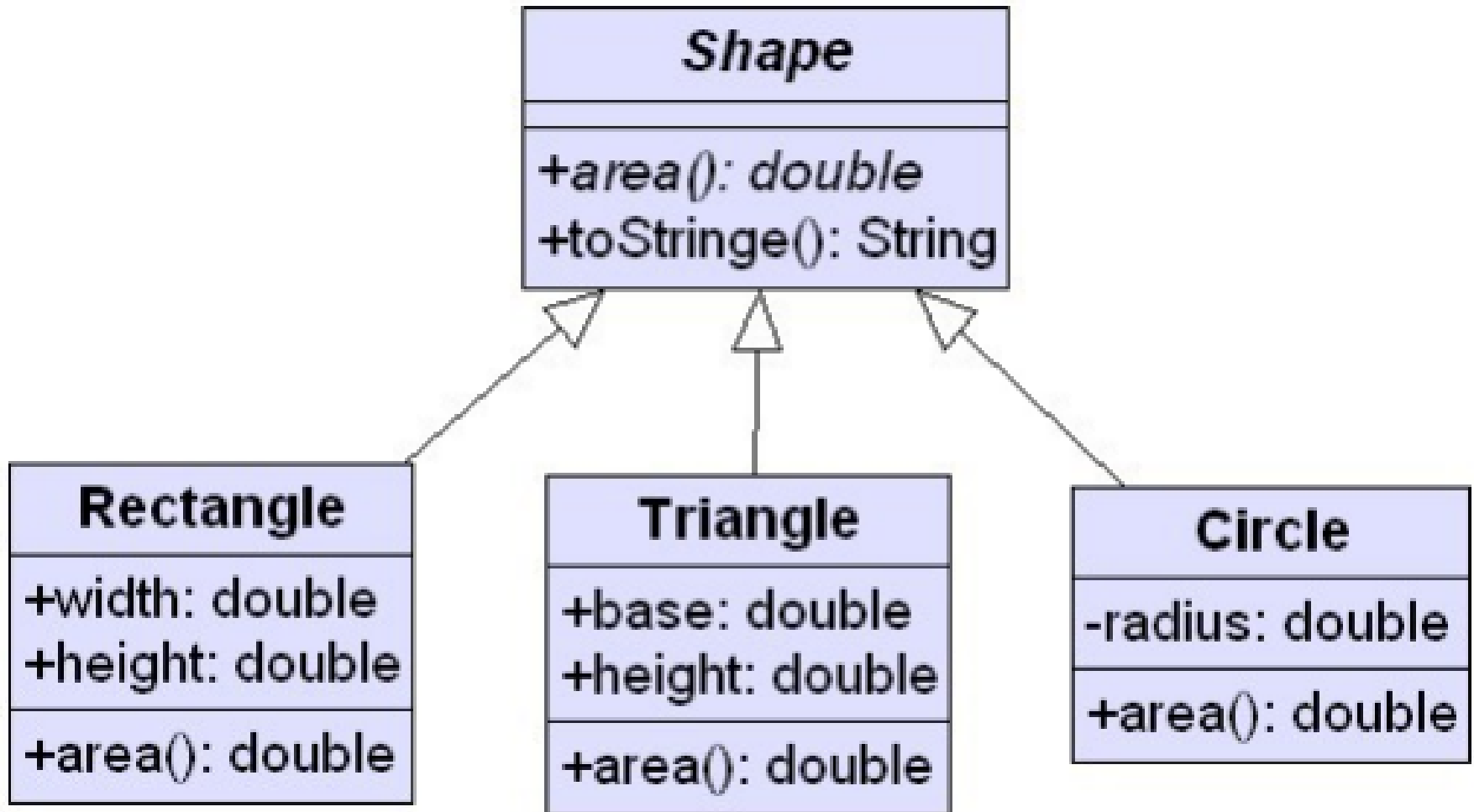
- *Generalization* berhubungan dengan konsep *inheritance* pada OOP.
- Berlaku istilah Parent-Child atau SuperClass-SubClass.
- Merepresentasikan hubungan : “is-a”.
- Nama *abstract class* ditulis dengan *italic*.

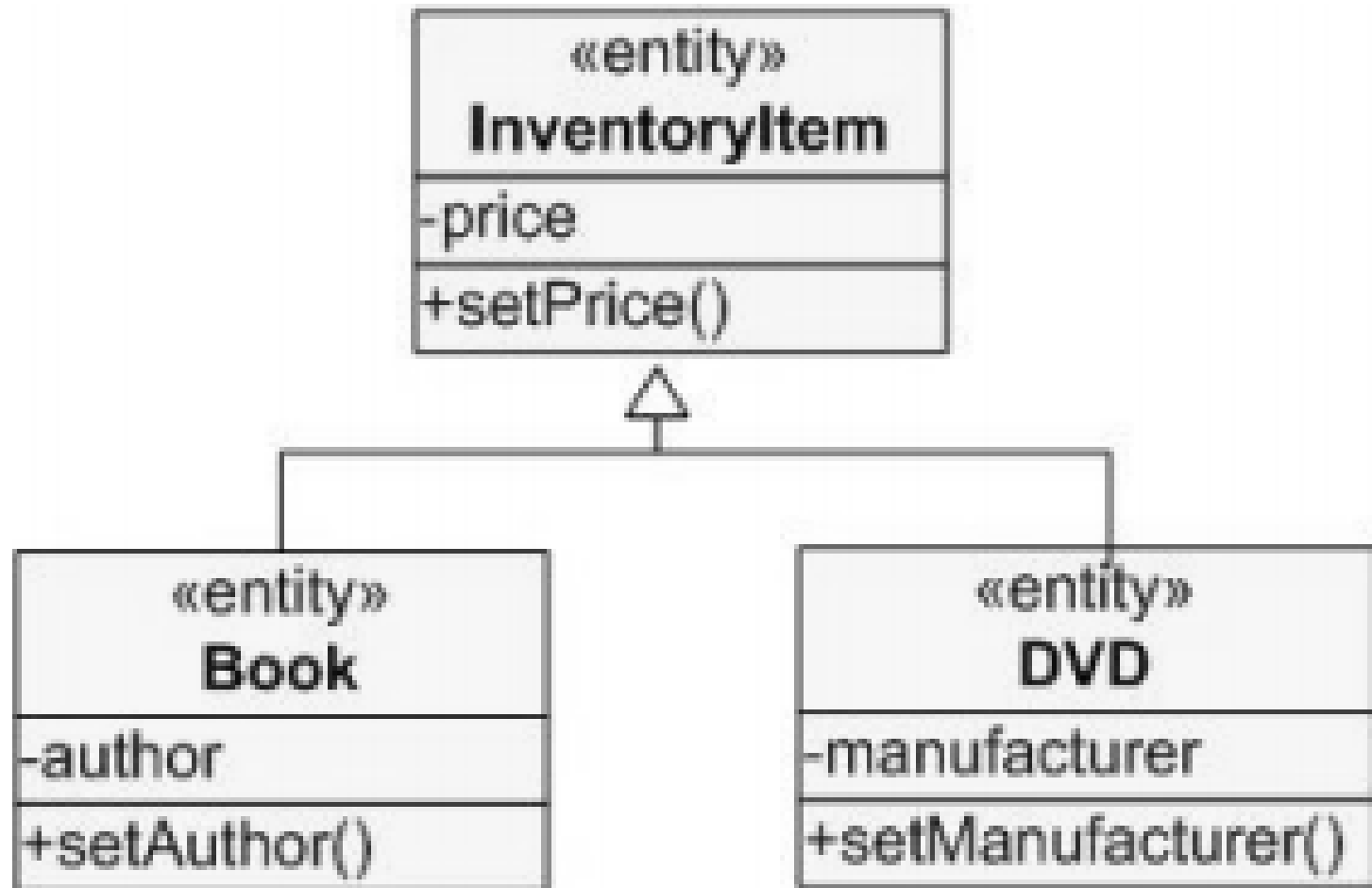


Notasi *Generalization* Pada *Class Diagram*



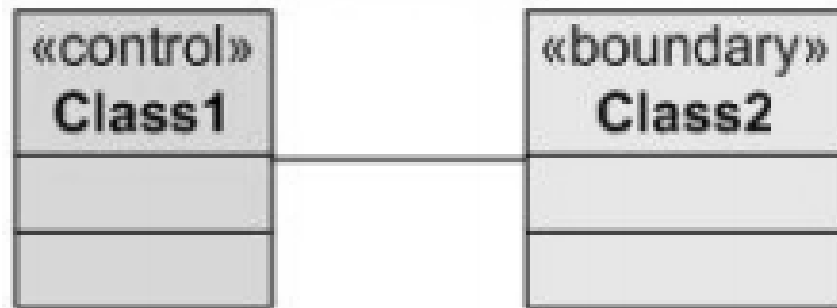
Contoh *Generalization* Pada *Class Diagram* (1)





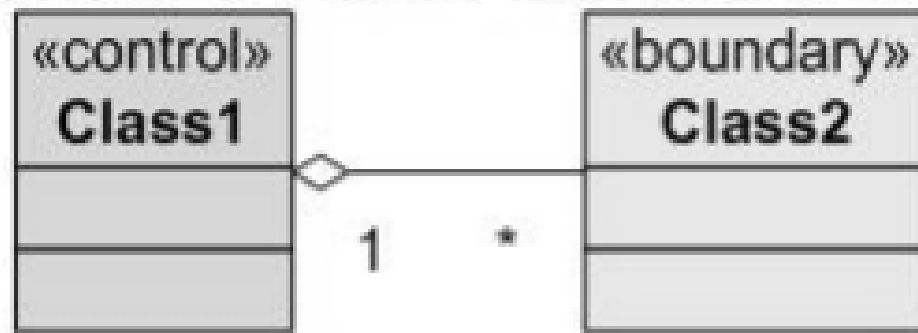
Simple Association

- A structural link between two peer classes.
- There is an association between Class1 and Class2.



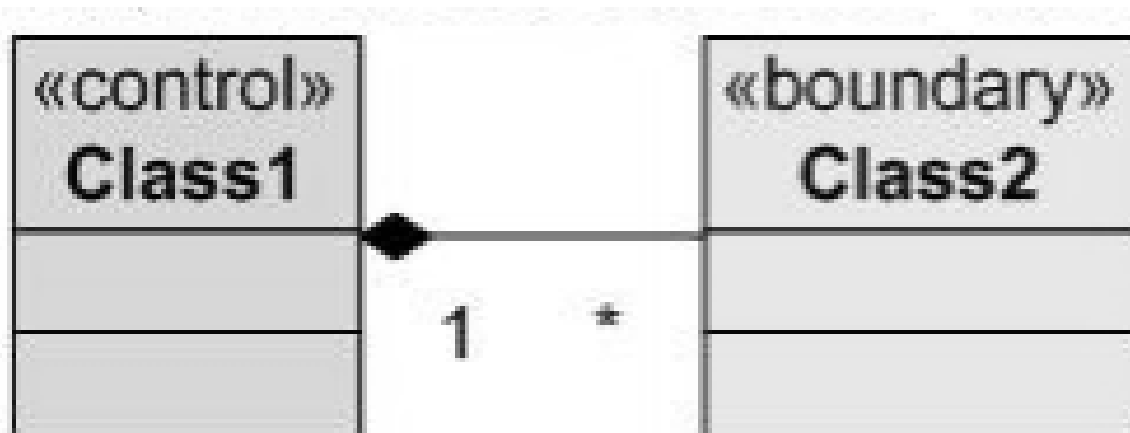
Aggregation

- A special type of association. It represents a “part of” relationship.
- Class2 is part of Class1.
- Objects of Class1 and Class2 have separate lifetimes.



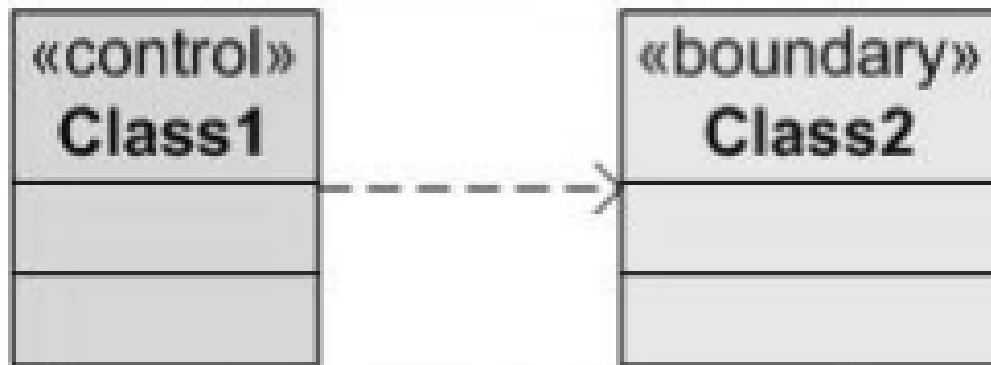
Composition

- A special type of aggregation where parts are destroyed when the whole is destroyed.
- Objects of Class2 live and die with Class1.
- Class2 cannot stand by itself.

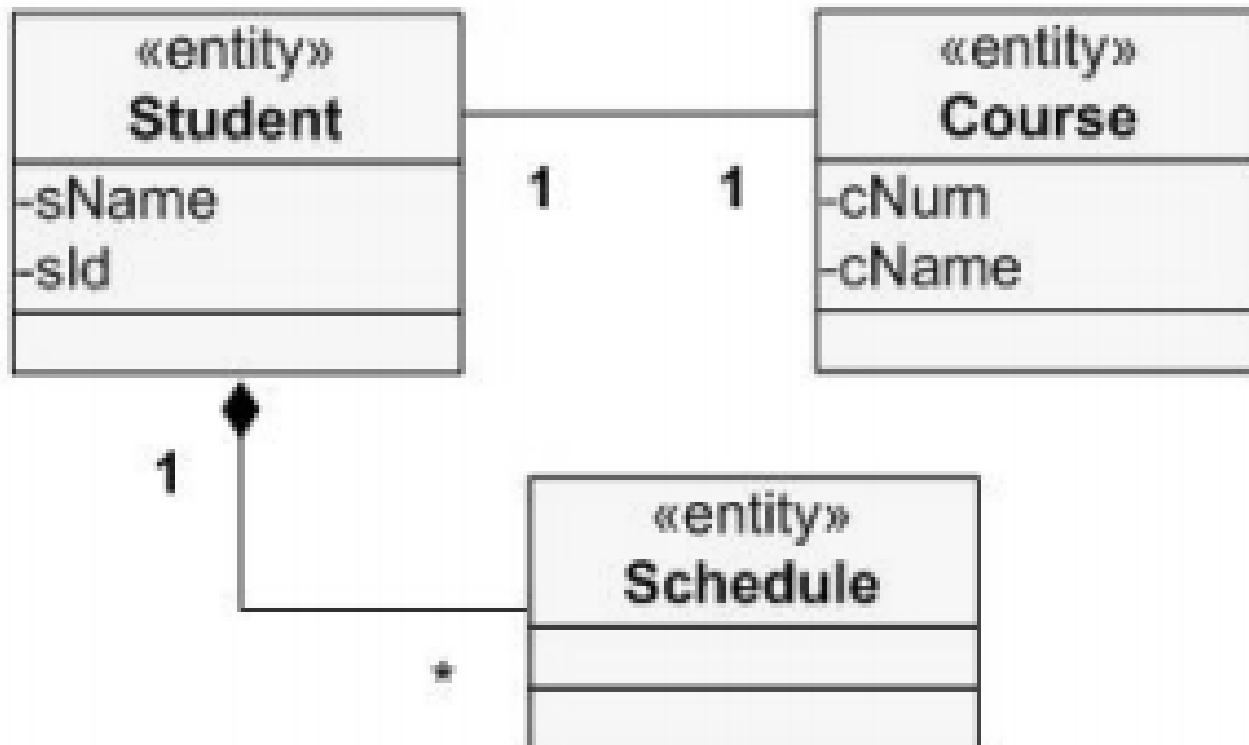


Dependency

- Exists between two classes if changes to the definition of one may cause changes to the other (but not the other way around).
- Class1 depends on Class2.



Contoh *Relationship-Association* dan Composition



Contoh *Relationship-Aggregation*



Contoh *Relationship-Depencency*

