

Bab 8: Pemodelan dan Analisis Berorientasi Obyek menggunakan UML



**ANALISIS DAN PERANCANGAN SISTEM
INFORMASI**

MONICA A. KAPPIANTARI - 2009

*Source: Whitten, J.L., L.D. Lonnie and K.C. Dittman, Systems Analysis and Design Methods,
6th ed., McGraw-Hill, Boston, 2004.*

Bab 8

2

Referensi

- Whitten, Bab 11
- users.encs.concordia.ca/~gregb/home/PPT/Io5IntrotoOO-class.ppt
- www.ece.ualberta.ca/~reform/ece521w2009/lab/lab_uml_part03.ppt

Topik

1. Object Orientation: Basic Concepts
2. UML Diagram
3. Class Diagram

1. Object orientation: basic concepts

3

Basic Concepts of Object Orientation

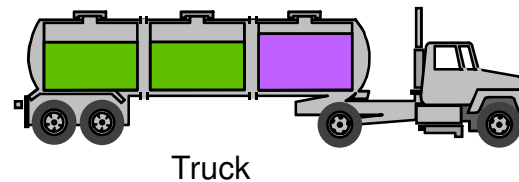
4

1. Object
2. Class
3. Attribute
4. Operation
5. Interface (Polymorphism)
6. Relationships

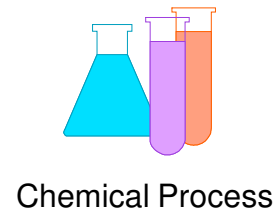
1. What is an Object?

- Informally, an object represents an entity, either physical, conceptual, or software

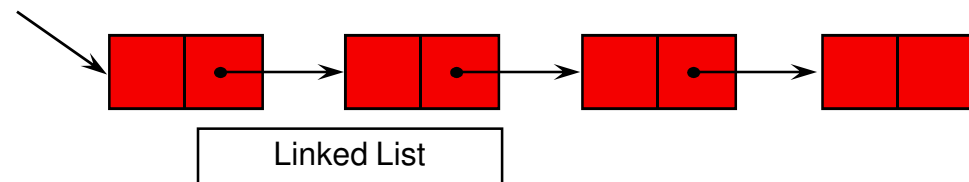
- Physical entity



- Conceptual entity



- Software entity



A More Formal Definition

6

- An object is a concept, abstraction, or thing with sharp boundaries and meaning for an application
- An object is something that has:
 - State
 - Behavior
 - Identity

Representing Objects

7

- An object is represented as rectangles with underlined `

: Professor

Class Name Only

ProfessorClark :
Professor

Class and Object Name

ProfessorClark

Object Name Only



Professor Clark

2. What is a Class?

8

- A class is a description of a group of objects with common properties (attributes), behavior (operations), relationships, and semantics
 - An object is an instance of a class
- A class is an abstraction in that it:
 - Emphasizes relevant characteristics
 - Suppresses other characteristics

OO Principle: Abstraction

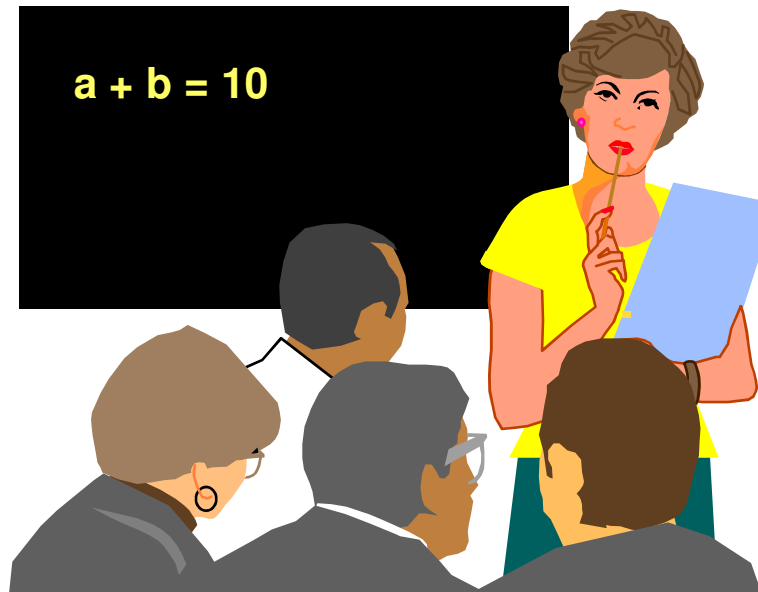
Sample Class

9

Class Course

Properties

Name
Location
Days offered
Credit hours
Start time
End time



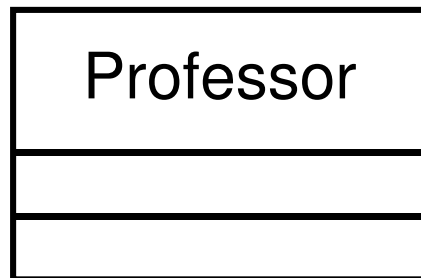
Behavior

Add a student
Delete a student
Get course roster
Determine if it is full

Representing Classes

10

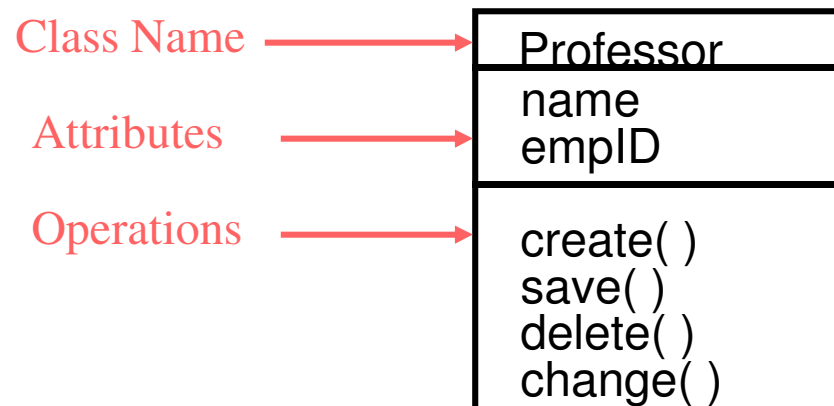
- A class is represented using a compartmented rectangle



Class Compartments

11

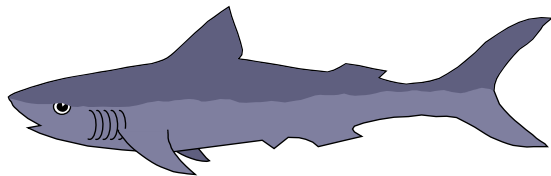
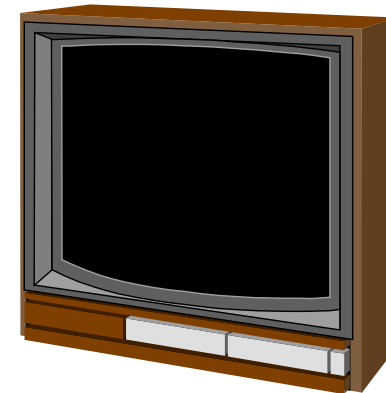
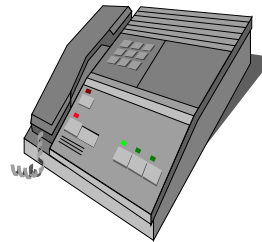
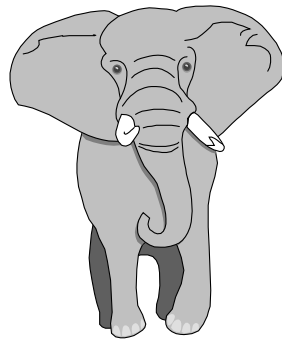
- A class is comprised of three sections
 - The first section contains the class name
 - The second section shows the structure (attributes)
 - The third section shows the behavior (operations)



Classes of Objects

12

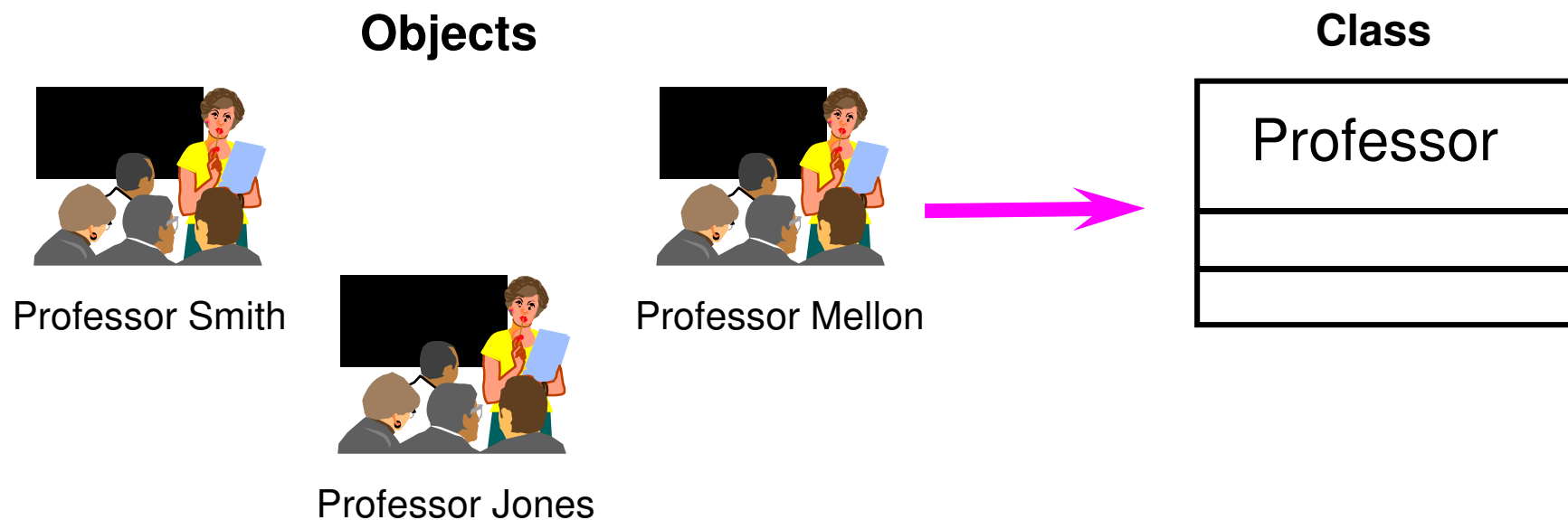
- How many classes do you see?



The Relationship Between Classes and Objects

13

- A class is an abstract definition of an object
 - It defines the structure and behavior of each object in the class
 - It serves as a template for creating objects
- Objects are grouped into classes

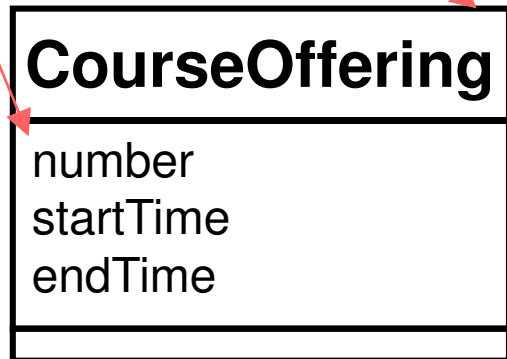


3. What is an Attribute?

14

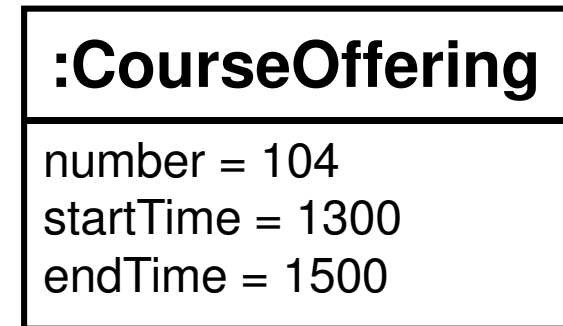
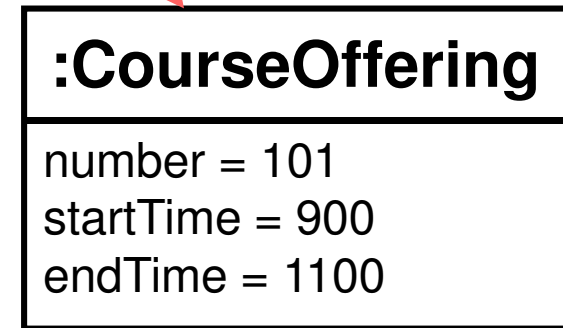
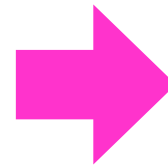
Class

Attribute



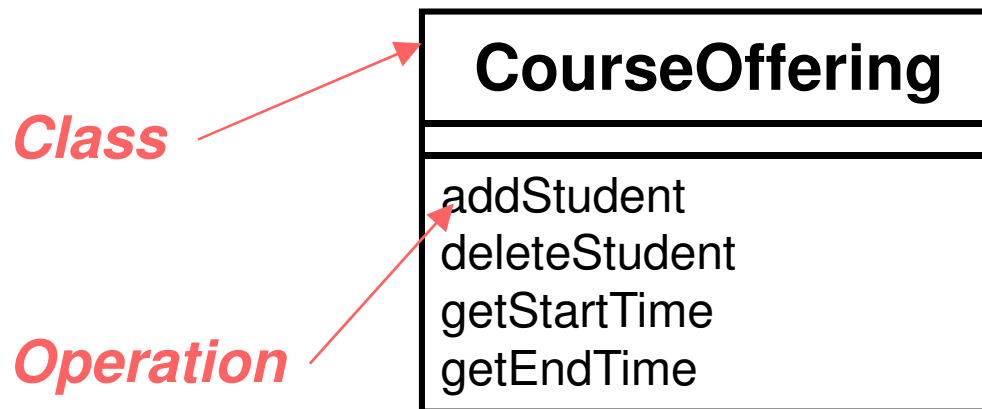
Object

Attribute Value



4. What is an Operation?

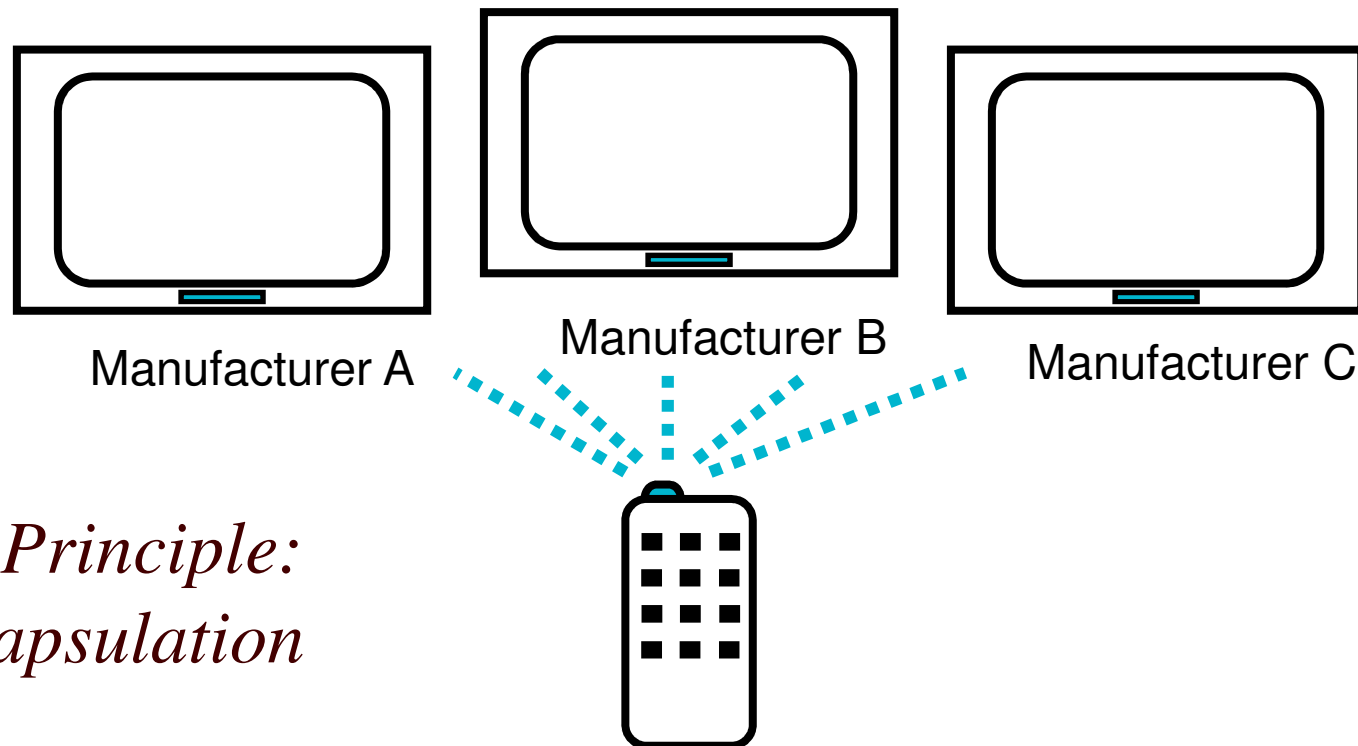
15



5. What is Polymorphism?

16

- The ability to hide many different implementations behind a single interface

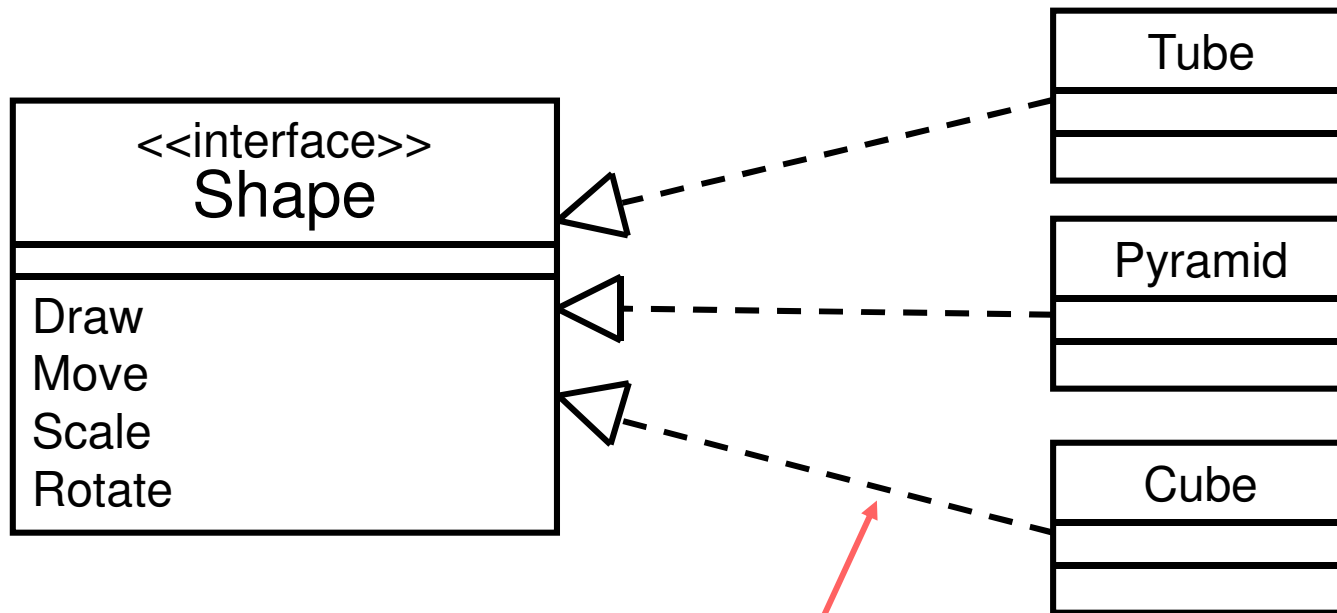


*OO Principle:
Encapsulation*

What is an Interface?

17

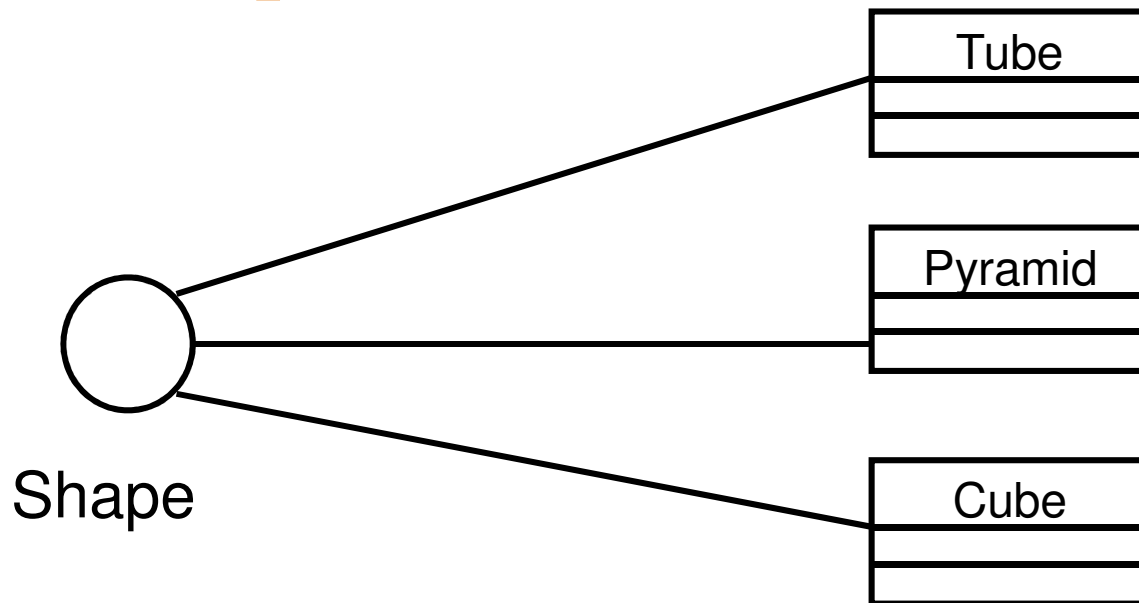
- Interfaces formalize polymorphism
- Interfaces support “plug-and-play” architectures



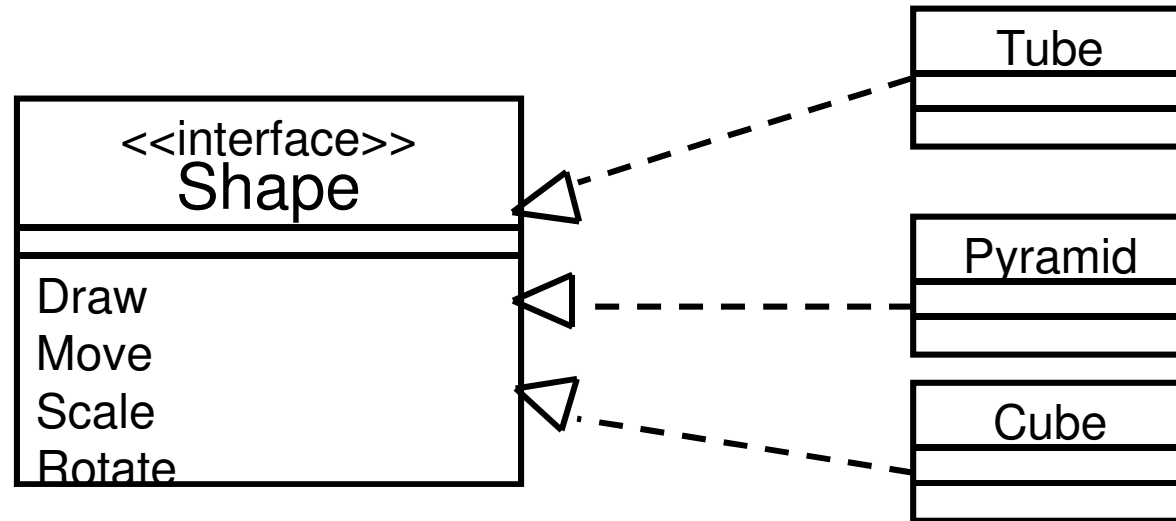
Realization relationship

Interface Representations

Elided/Iconic
Representation
("lollipop")



Canonical
(Class/Stereotype)
Representation



6. Relationships

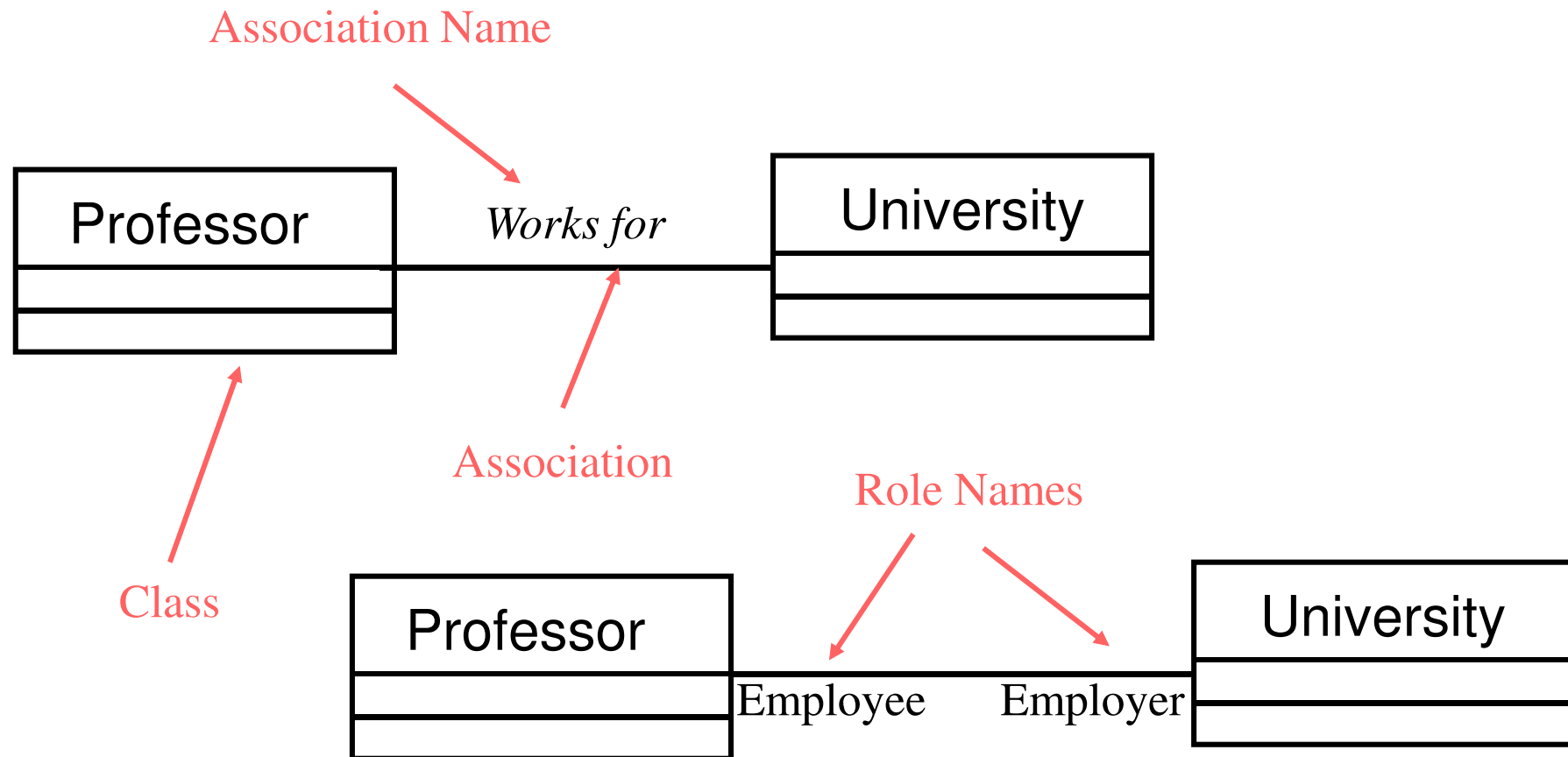
19

- Association
 - Aggregation
 - Composition
- Dependency
- Generalization
- Realization

Relationships: Association

20

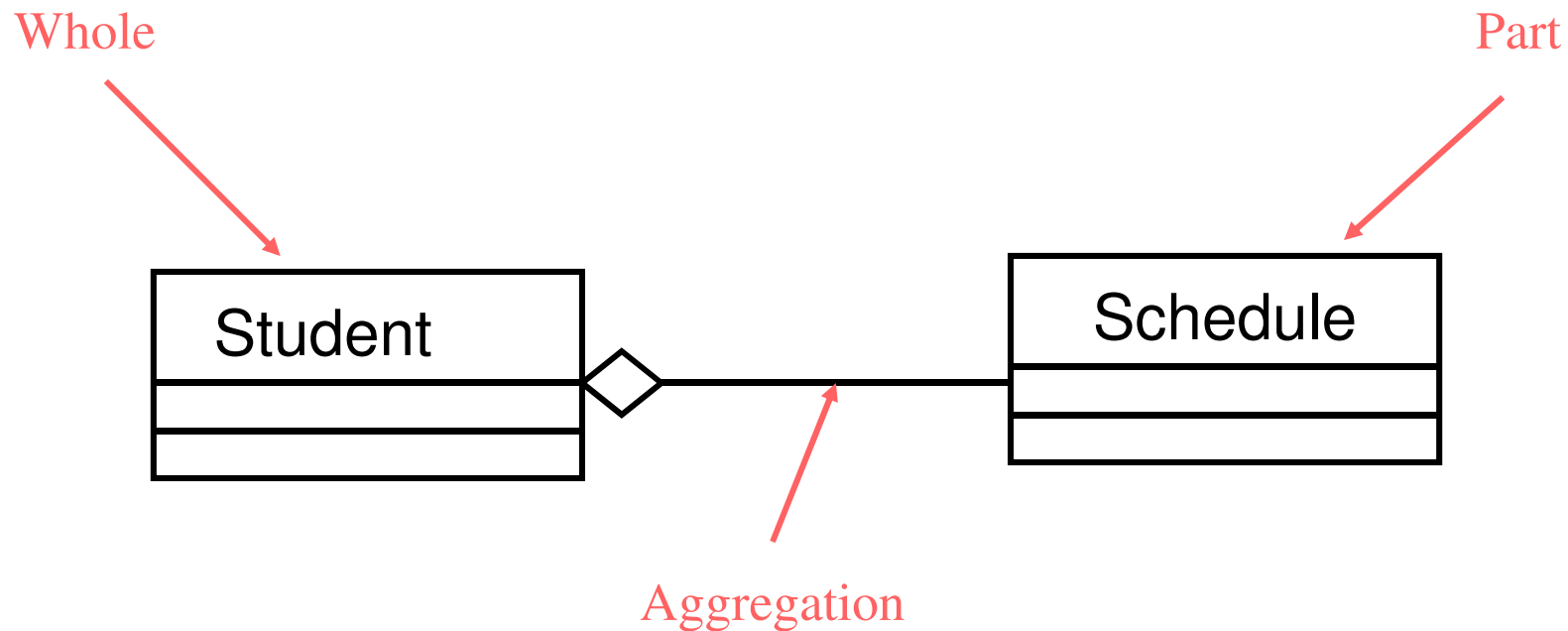
- Models a semantic connection among classes



Relationships: Aggregation

21

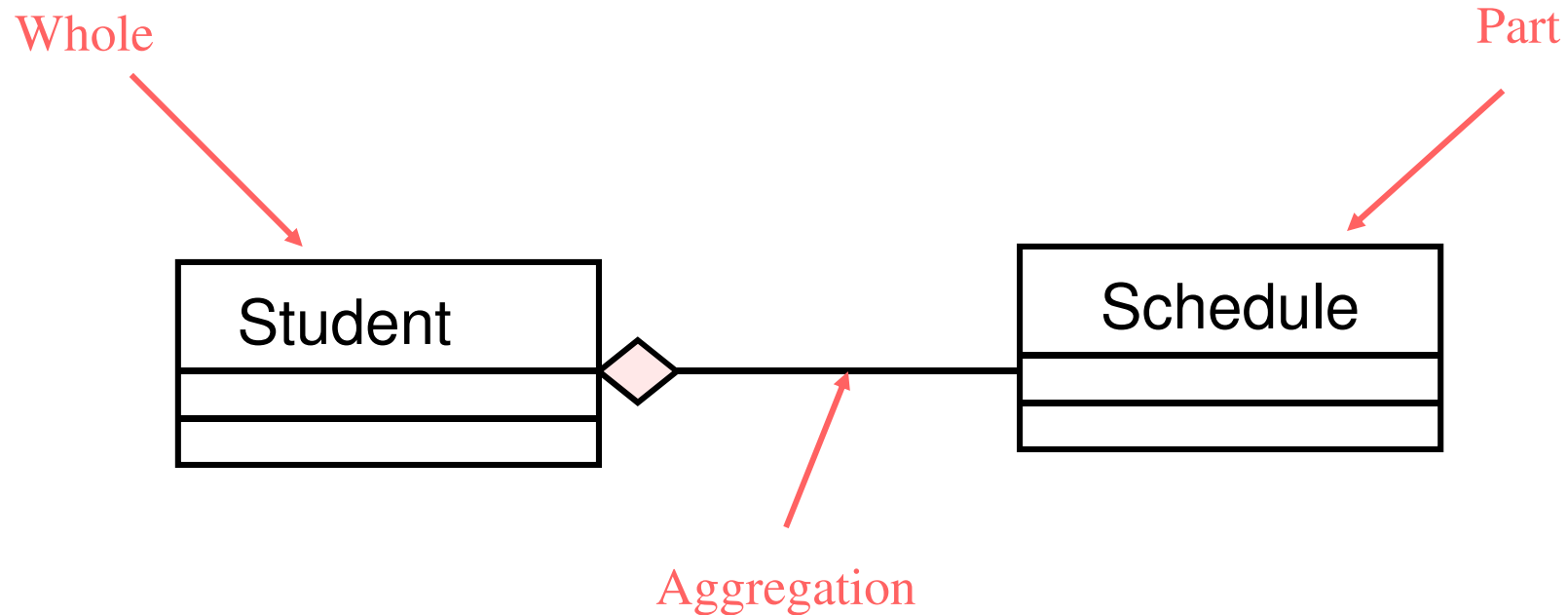
- A special form of association that models a whole-part relationship between an aggregate (the whole) and its parts



Relationships: Composition

22

- A form of aggregation with strong ownership and coincident lifetimes
 - The parts cannot survive the whole/aggregate



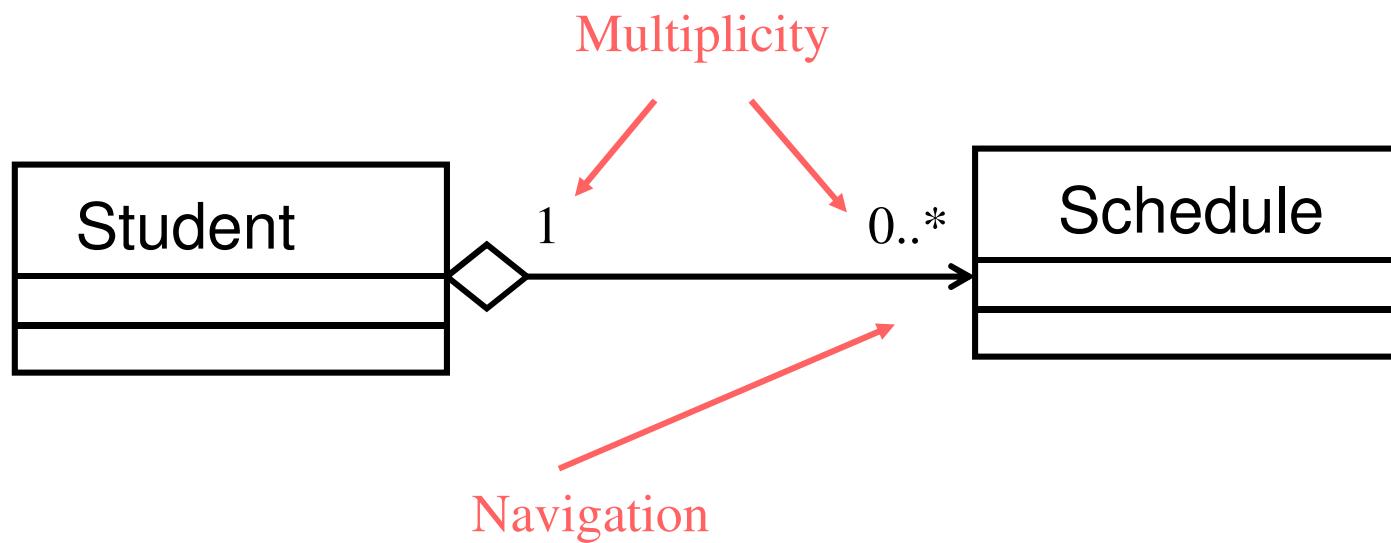
Association: Multiplicity and Navigation

23

- Multiplicity defines how many objects participate in a relationships
 - The number of instances of one class related to ONE instance of the other class
 - Specified for each end of the association
- Associations and aggregations are bi-directional by default, but it is often desirable to restrict navigation to one direction
 - If navigation is restricted, an arrowhead is added to indicate the direction of the navigation

Example: Multiplicity and Navigation

24

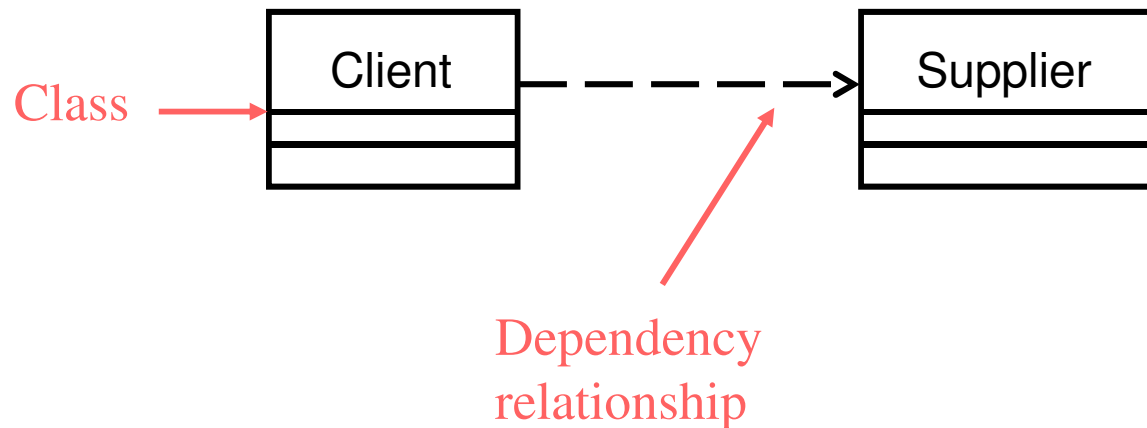


Multiplicity	UML Multiplicity Notation	Association with Multiplicity	Association Meaning
Exactly 1	1 or <i>leave blank</i>		An employee works for one and only one department.
Zero or 1	0..1		An employee has either one or no spouse.
Zero or more	0..* or *		A customer can make no payment up to many payments.
1 or more	1..*		A university offers at least 1 course up to many courses.
Specific range	7..9		A team has either 7, 8, or 9 games scheduled

Relationships: Dependency

26

- A relationship between two model elements where a change in one **may** cause a change in the other
- Non-structural, “using” relationship



Relationships: Generalization

27

- A relationship among classes where one class shares the structure and/or behavior of one or more classes
- Defines a hierarchy of abstractions in which a subclass inherits from one or more superclasses
 - Single inheritance
 - Multiple inheritance
- Generalization is an “is-a-kind of” relationship

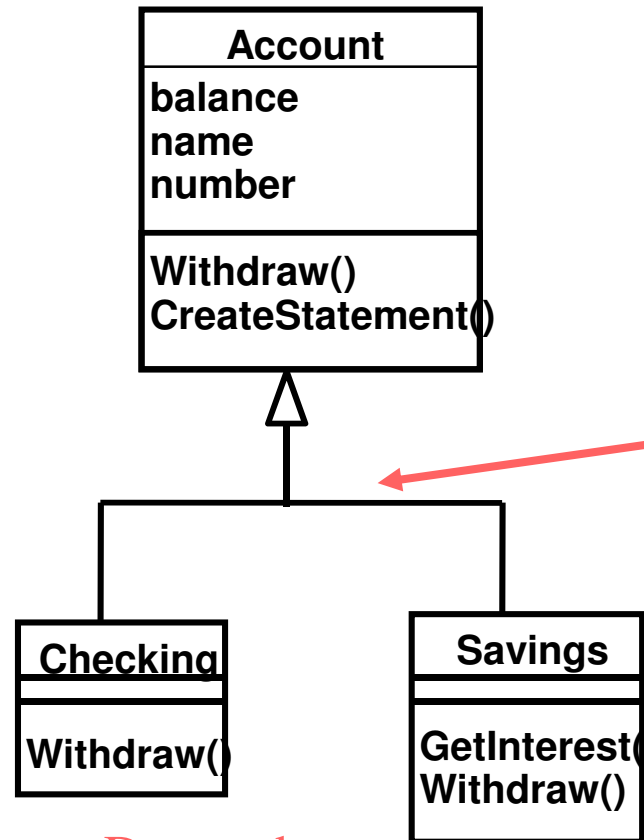
Example: Single Inheritance

28

One class inherits
from another

Superclass
(parent)

Ancestor



Generalization
Relationship

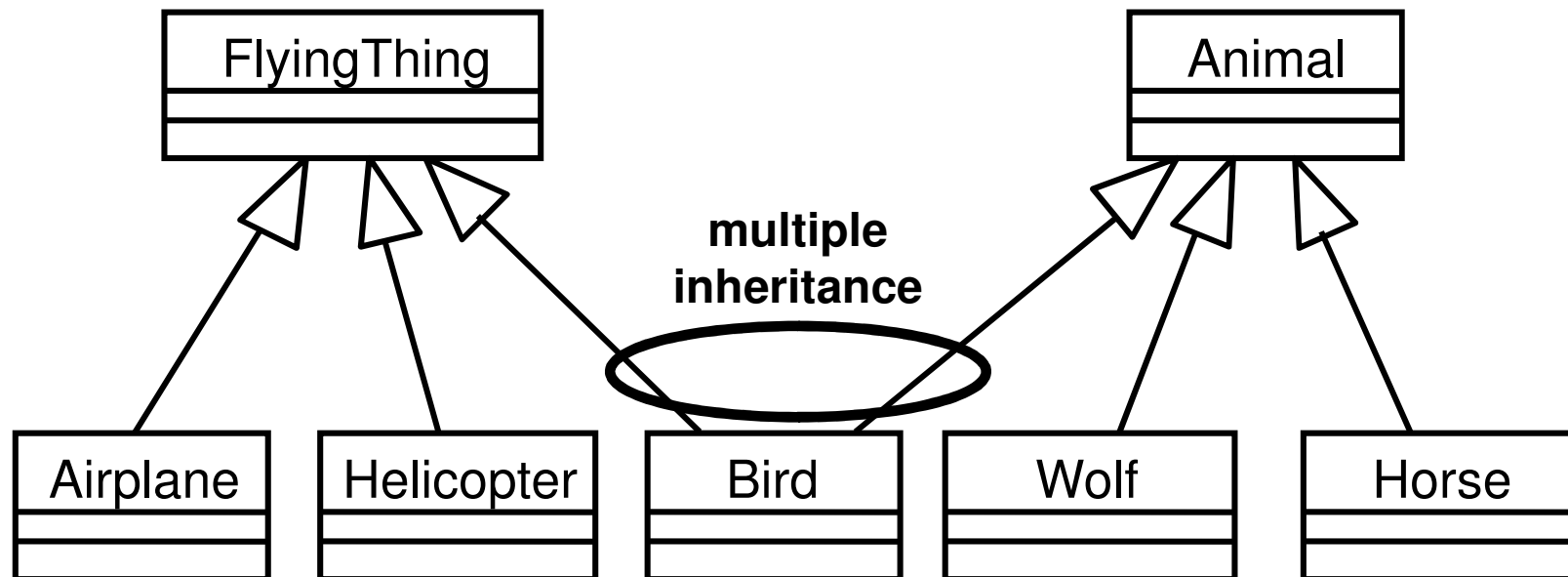
Subclasses

Descendants

Example: Multiple Inheritance

29

- A class can inherit from several other classes



***Use multiple inheritance only when needed, and
always with caution !***

What Gets Inherited?

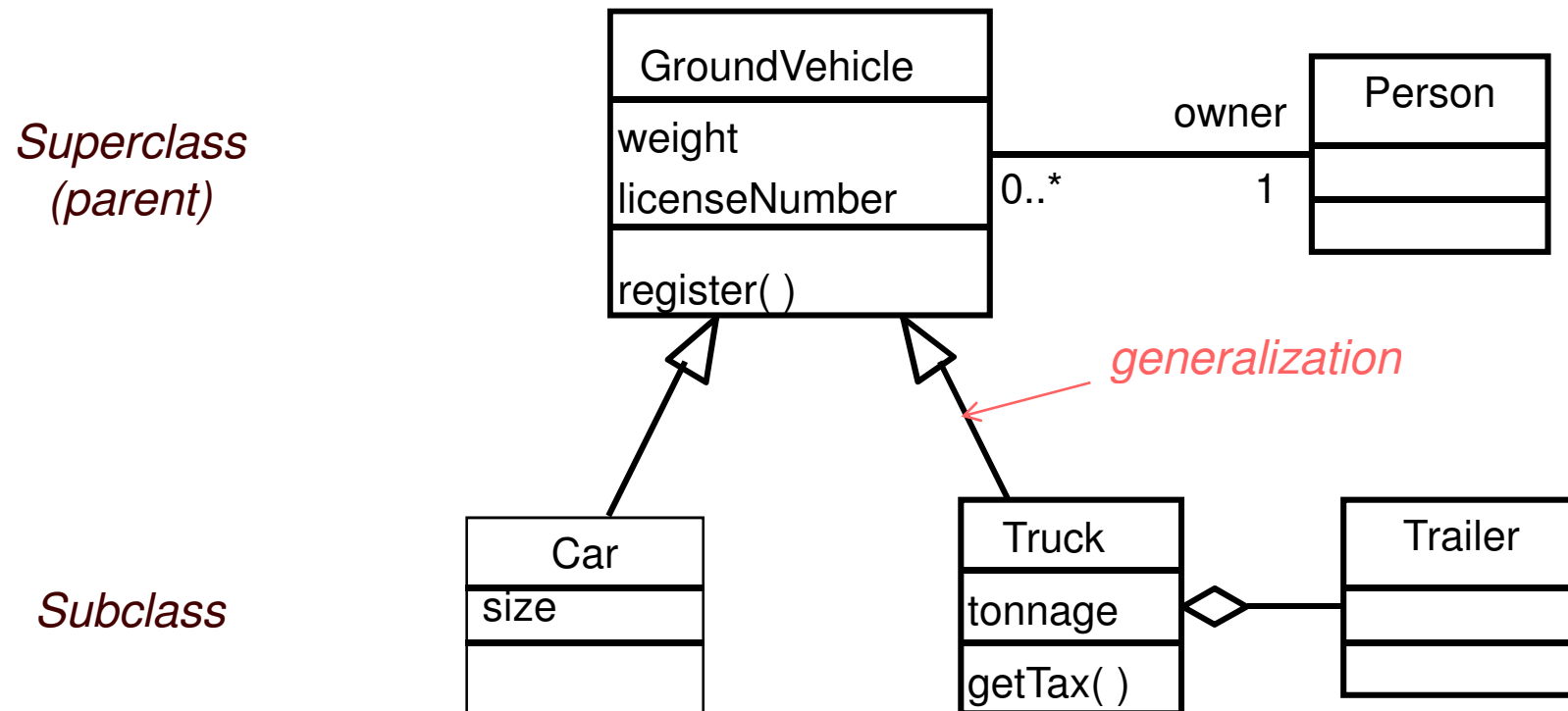
30

- A subclass inherits its parent's attributes, operations, and relationships
- A subclass may:
 - Add additional attributes, operations, relationships
 - Redefine inherited operations (use caution!)
- Common attributes, operations, and/or relationships are shown at the highest applicable level in the hierarchy

Inheritance leverages the similarities among classes

Example: What Gets Inherited

31



2. UML Diagram

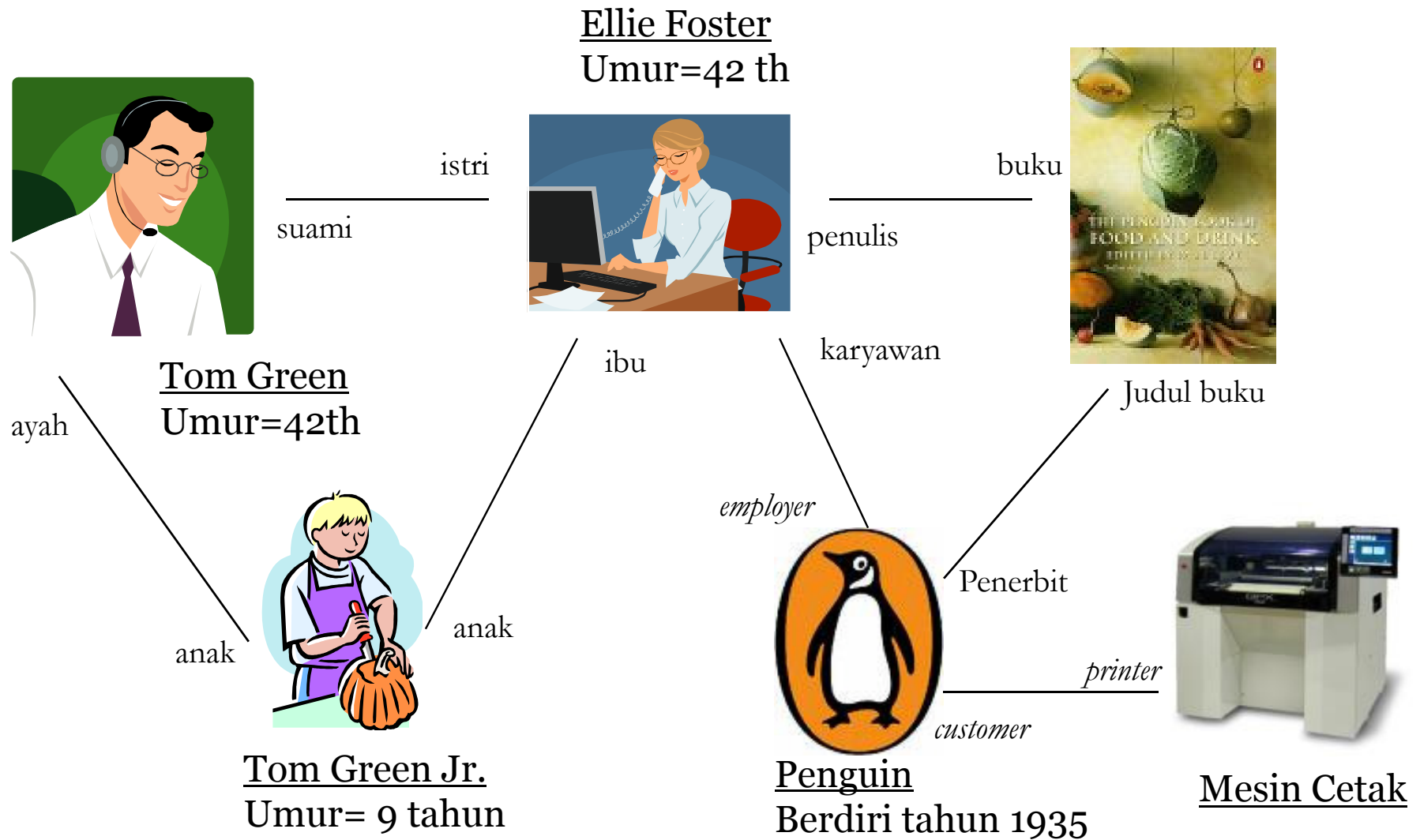
32

Unified Modeling Language (UML)

33

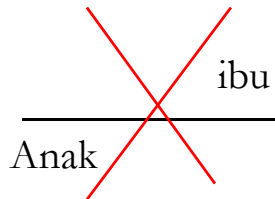
- **Unified Modeling Language (UML)** is a standardized general-purpose modeling language in the field of software engineering (http://en.wikipedia.org/wiki/Unified_Modeling_Language)
- UML digunakan untuk menggambarkan obyek, atributnya (fakta yang perlu kita ketahui tentang obyek tersebut), dan relasinya dengan obyek lain pada saat tertentu

UML



UML

Ellie Foster
Umur=42 th



ayah

Tom Green
Umur=42th

ibu

anak



anak

Tom Green Jr.
Umur= 9 tahun

UML Diagram

36

- ★ • **Use-Case Model Diagrams**
- **Static Structure Diagrams**
- ★ ○ *Class diagrams*
- *Object diagrams*
- **Interaction Diagrams**
- *Sequence diagrams*
- *Collaboration diagrams*
- **State Diagrams**
- *Statechart diagrams*
- *Activity diagrams*
- **Implementation Diagrams**
- *Component diagrams*
- *Deployment diagrams*

3. Class Diagram

37

Class Diagram

38

- Class diagrams are static diagrams consisting of pieces that make up the systems or subsystem.
- Class diagrams begin with a high-level overview of the functionality of the system.
- As the diagrams mature – classes become more detailed.
- Class diagrams are used to indicate where data resides and where functionality is available through the use of class attributes and operations

Class Diagram

39

- Class diagrams also describe the various kinds of static relationships that exist among classes. There are two principle kinds of static relationships:
 - **association** (for example, a customer may rent a number of videos)
 - **subtypes** (a nurse is a kind of person)

Class Diagram: an example

40

