

UML1.5

Details of Class Definition

Koichiro OCHIMIZU

School of Information Science

JAIST

Schedule(2/3)

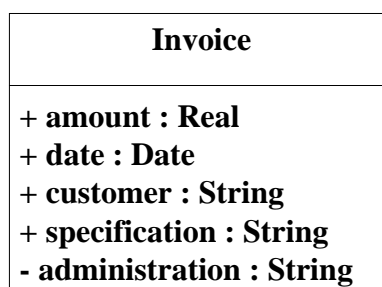
- Feb. 27th
 - 13:00 Introduction to Java Programming
 - 14:30 **Outline of UML: Static Modeling**
(usecase modeling, details of class definition)
- Feb. 28th
 - 13:00 Outline of UML: Dynamic Modeling
(state machine)
 - 14:30 Outline of UML: Dynamic Modeling
(communication diagram, sequence diagram)

A Simple model of an insurance business



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Visibility



- + public can access it from the other classes
 - It violates information hiding principle
- - private cannot access it from other classes
- # protected can access it from sub classes

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Default values

Invoice
+ amount : Real + date : Date = Current date + customer : String + specification : String - administration : String = "Unspecified"

- Assigned at the same time an object of the class is created

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

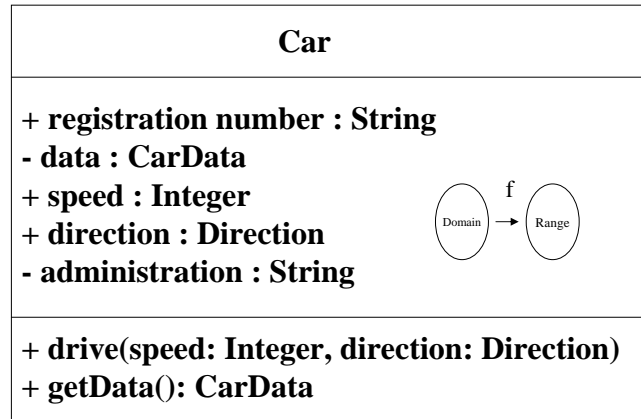
A class-scope attribute or a class variable

Invoice
+ amount : Real + date : Date = Current date + customer : String + specification : String - administration : String = "Unspecified" - <u>number of invoices : Integer</u>

- A class variable (underlined) is shared by all objects of the class

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

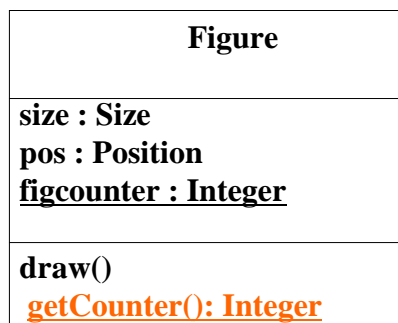
Signature



- Signature: a return-type, a name, zero or more parameters

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Class-scope operation



- Access class-scope attributes
- Creation of objects

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Visibility of Operation

Figure
size : Size pos : Position
+ draw() + scaleFigure(percent: Integer = 25) + returnPos(): Position

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Java implementation

Figure
- x: Integer = 0 - y: Integer = 0
+ draw()

```
public class Figure
{
    private int x = 0;
    private int y = 0;

    public void draw()
    {
        // Java code for drawing the figure
    }
};
```

```
Figure fig1 = new Figure();
Figure fig2 = new Figure();
fig1.draw();
fig2.draw();
```

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

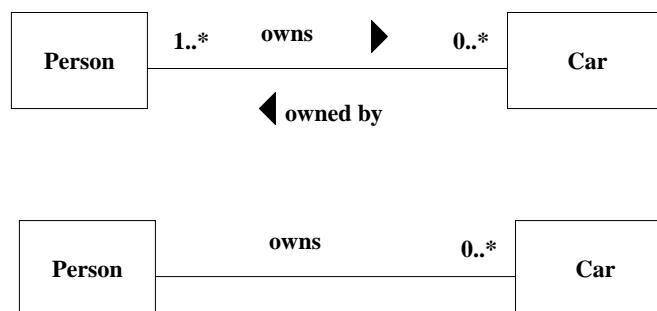
UML notations

Representation of Relationships between Classes

- Association
- > Navigable Association
-> Dependency
- > Generalization
- ◇ Aggregation: whole-part association
- ◊ Composition: the same life span

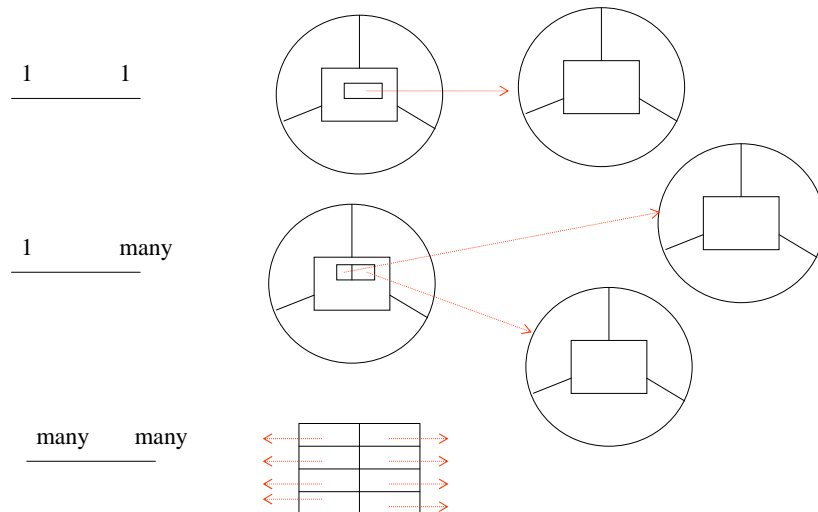
G.Booch, J.Rumbaugh, I. Jacobson , "The Unified Modeling Language User Guide", Addison Wesley, 1999.

Multiplicity

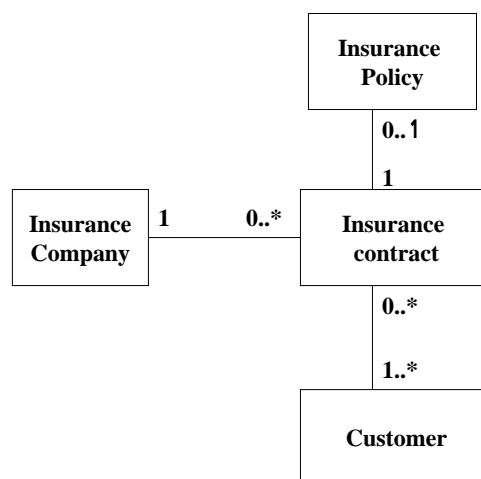


H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Implementation of Association

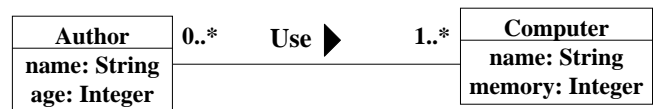


A class diagram describing an Insurance business



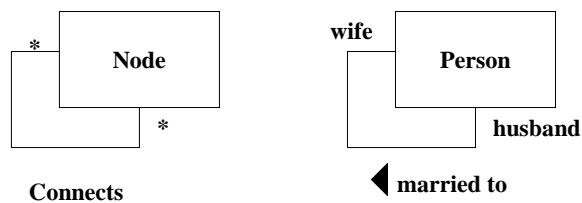
H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Object Diagram



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

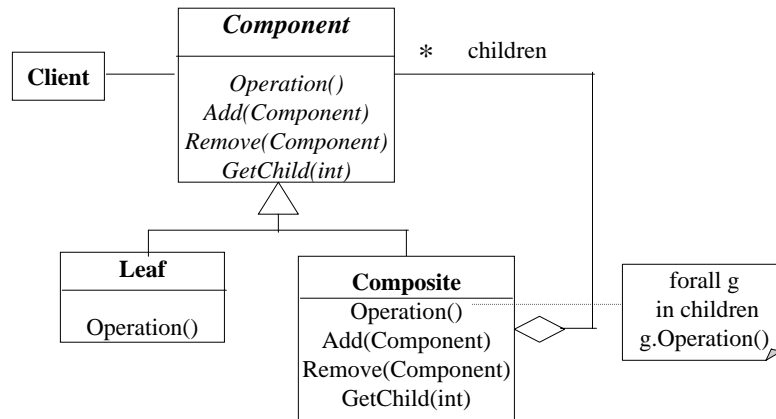
Recursive Association



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

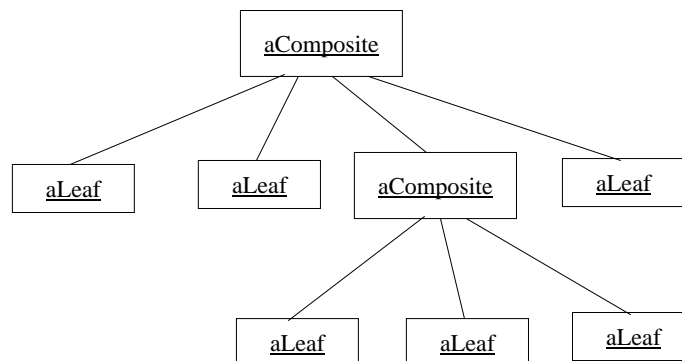
Composite

Clients use the Component class interface to interact with objects in the composite structure. If the recipient is a Leaf, then the request is handled directly. If the recipient is a Composite, then it usually forwards requests to its child components, possibly performing additional operations before and/or after forwarding.

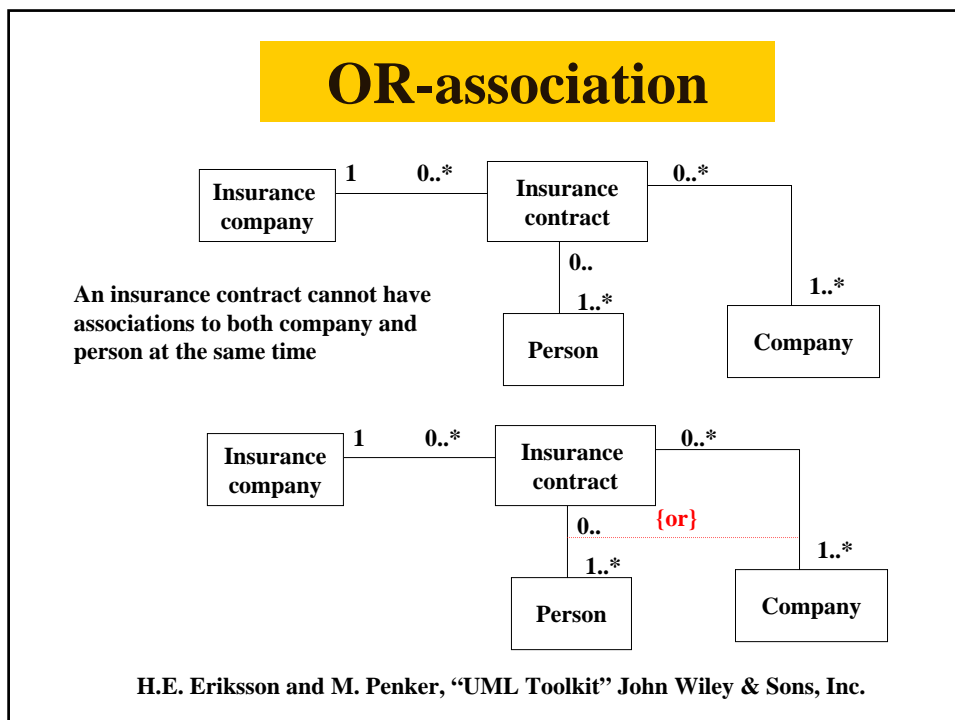
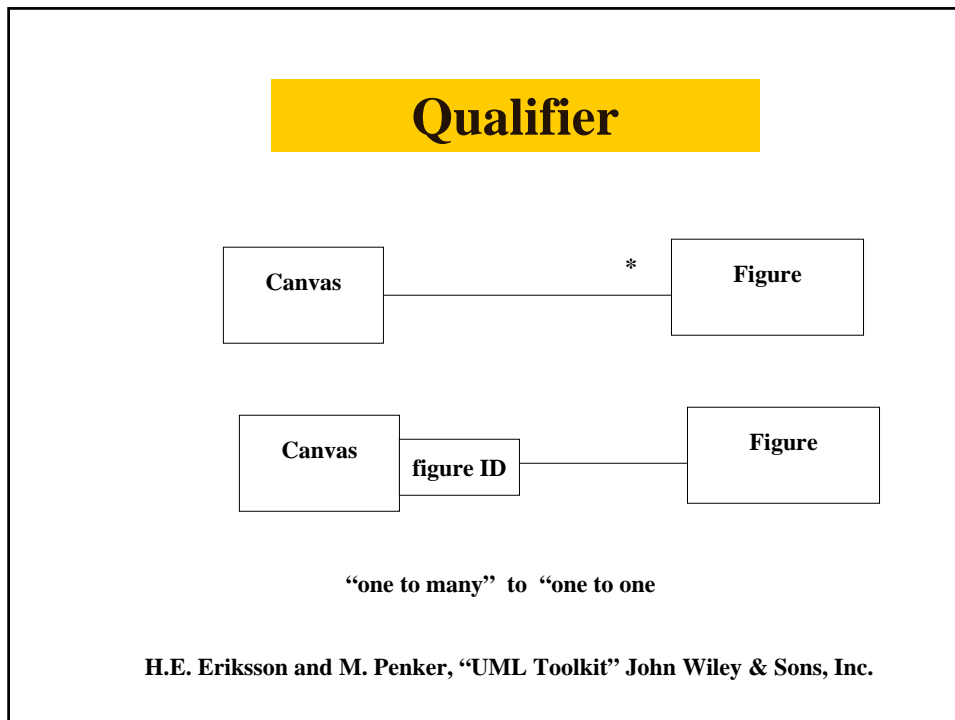


E.Gamma, R.Helm, R.Johnson, J.Vlissides, "Design Patterns", ADDISON-WESLEY, 1995.

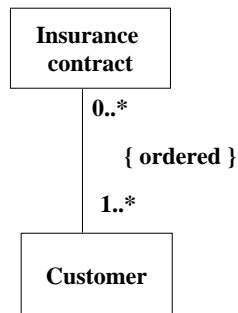
Object Diagram



E.Gamma, R.Helm, R.Johnson, J.Vlissides, "Design Patterns", ADDISON-WESLEY, 1995.



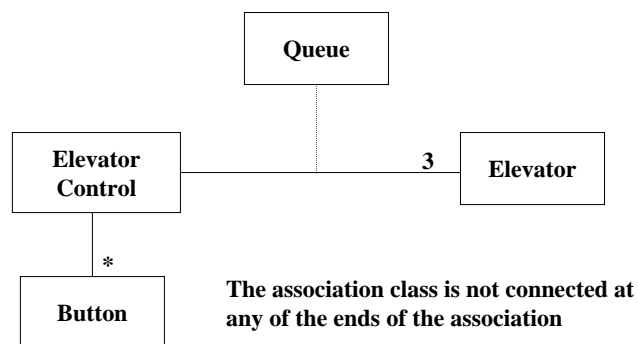
Ordered Association



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

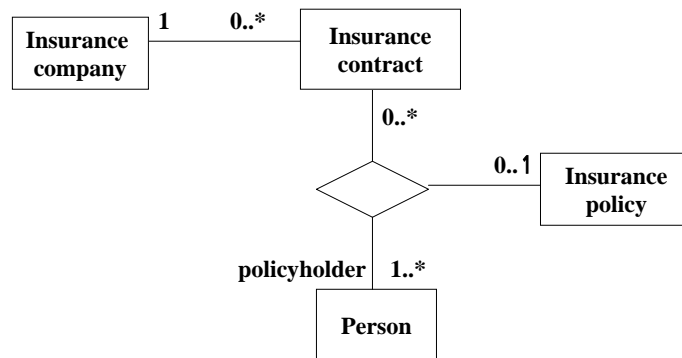
Association Class

A class can be attached to an association



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Ternary Association

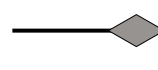


H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

What is an Aggregation ?

- Aggregation is a special case of association
- “whole-part”, “is-part-of”
- Special kinds of aggregation
 - aggregation
 - Shared Aggregation
 - Composition Aggregation

 **Aggregation**

 **Composition:** The part has the same life span with the whole

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

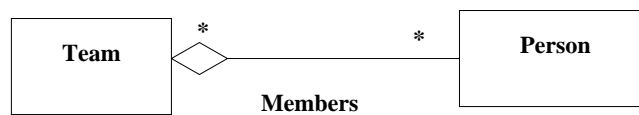
Aggregation



This example is not so good.

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Shared Aggregation

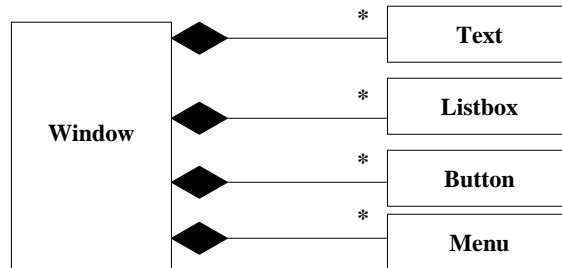


A team is composed of team members.

One person could be a member of many teams.

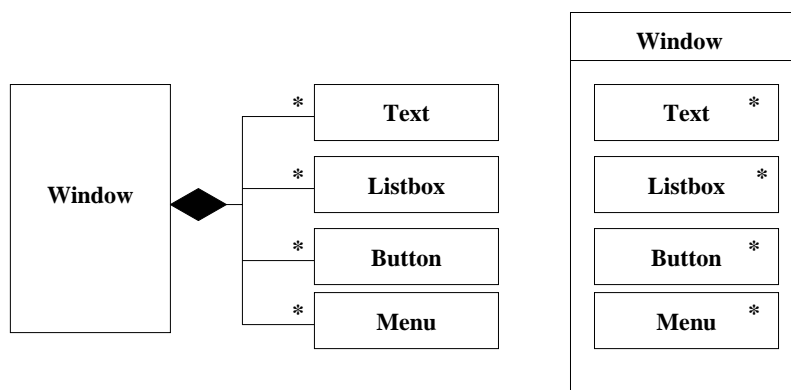
H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Composition Aggregation

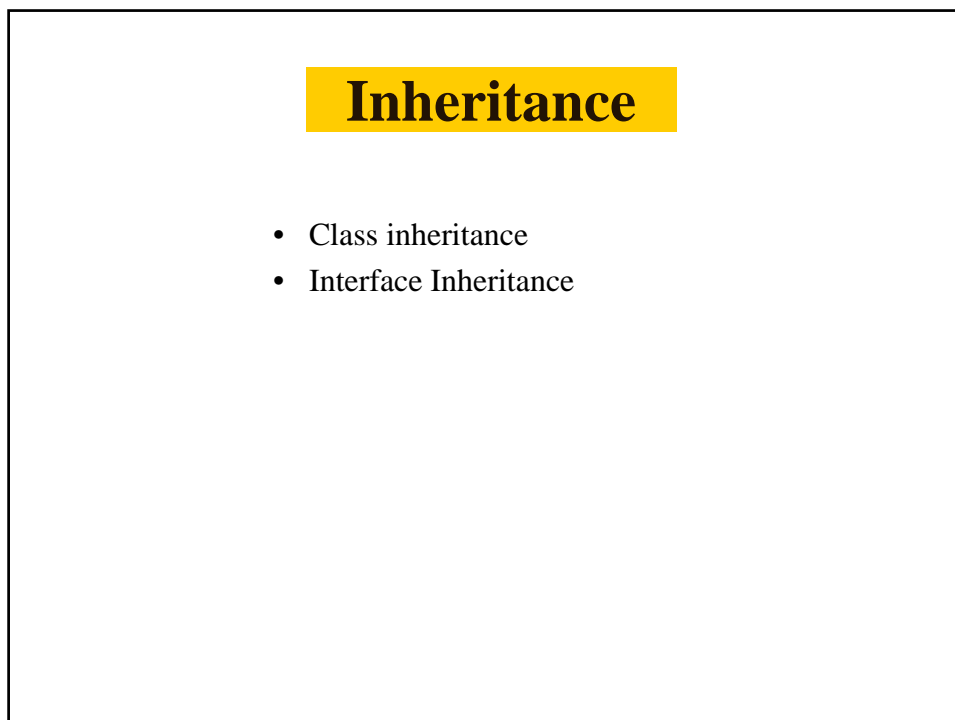
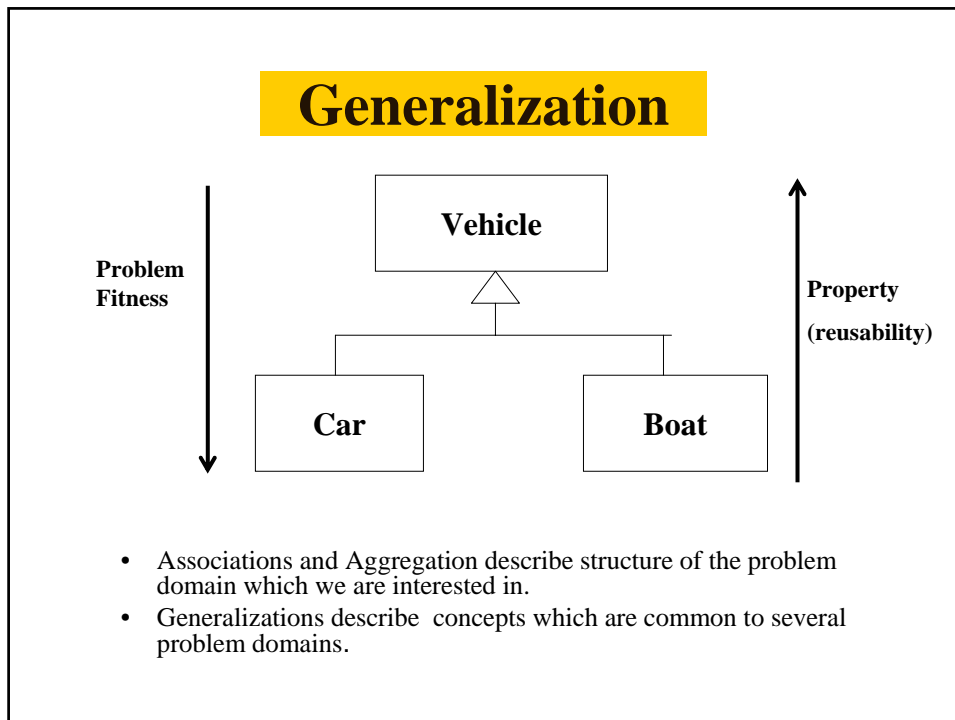


H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

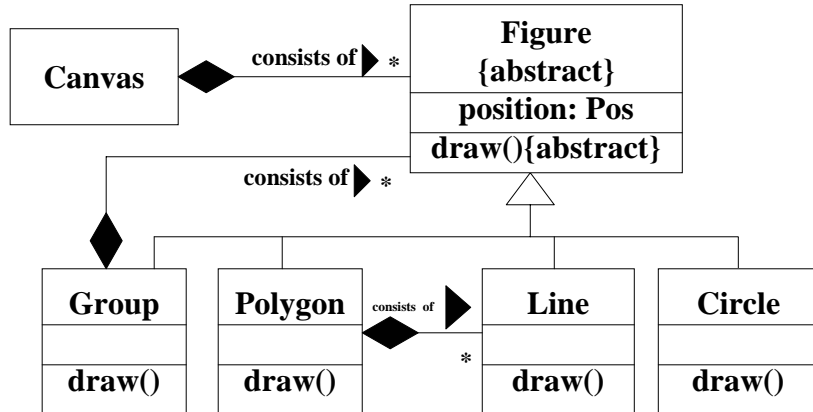
Another Expression of Aggregation



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

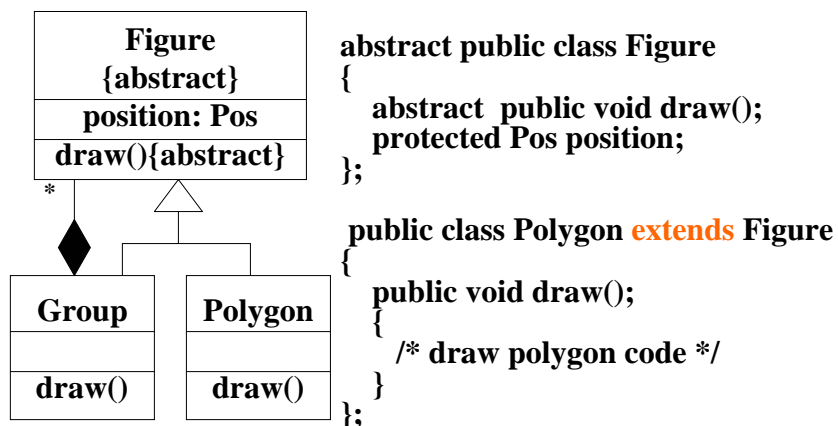


Combination of Inheritance and Aggregation



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

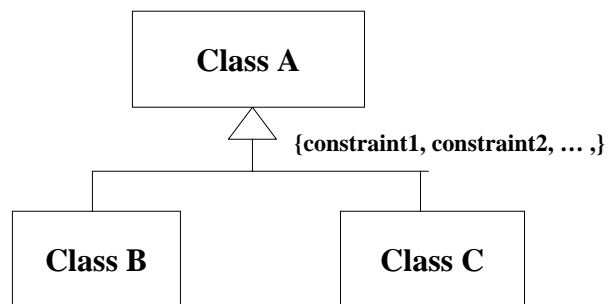
Java Implementation



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

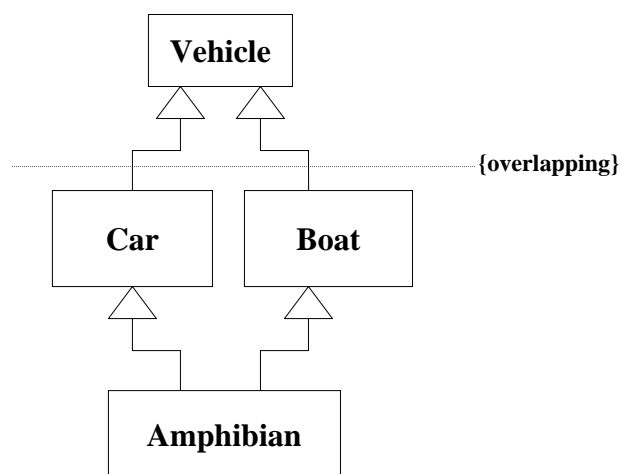
Constrained Generalization

Overlapping
Disjoint
Complete
Incomplete



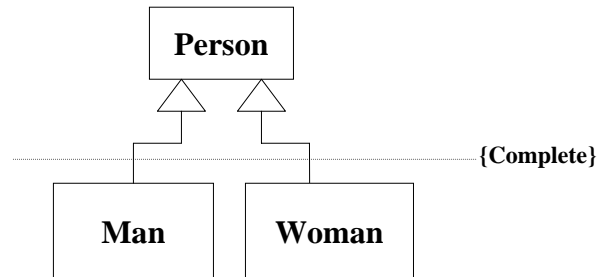
H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Overlapping and Disjoint



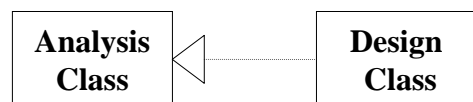
H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Complete and Incomplete



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

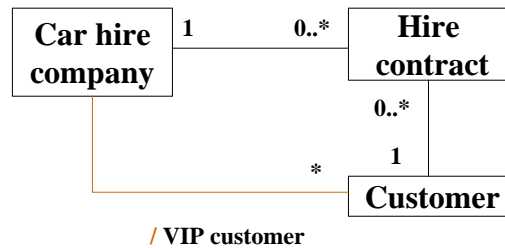
Refinement relationship



- A refinement is a relationship between two descriptions of the same thing, but at different levels of abstraction.
- It can be also used to model different implementations of the same thing.
- Support Configuration Management.
- Support traceability In the model.

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Derivation



- can be computed from other associations and attributes.
- The VIP customers are a derived association when company makes contracts with many customers.

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

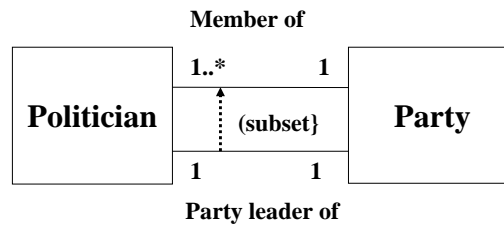
Derived Attribute



{ profit = sales price – cost price }

H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Subset Constraint



H.E. Eriksson and M. Penker, "UML Toolkit" John Wiley & Sons, Inc.

Exercise

- Review the content of my lecture by answering the following simple questions. Please describe the definition of each technical term.
 1. What is a navigable association?
 2. Please explain the difference of usage between association and generalization.
 3. What is a dependency?
 4. What is a multiplicity?
 5. How can we implement many to many correspondence between objects in a program?