

**Introduction to Object-Oriented Technologies**

**Basic Concepts on Reuse**

**Koichiro Ochimizu  
Japan Advanced Institute of  
Science and Technologies  
School of Information Science**

**Schedule(1/3)**

- Feb. 20th
  - 13:00 Scope and Goal (History of SPMs and SDMs, History of OO-technologies)
  - 14:30 Basic Concepts on Representing the World (object, class, association, aggregation...)
- Feb. 21th
  - 13:00 Basic Concepts on Interaction (message passing, operation and method, polymorphism)
  - 14:30 **Basic Concepts on Reuse** (super class, class inheritance, interface inheritance)

## Basic Concepts for Reuse

- Super Class and Sub Class
- Inheritance
- Class Library
- Multiple Inheritance
- Abstract Class
- Class Inheritance and Interface Inheritance
- Delegation and Object Composition

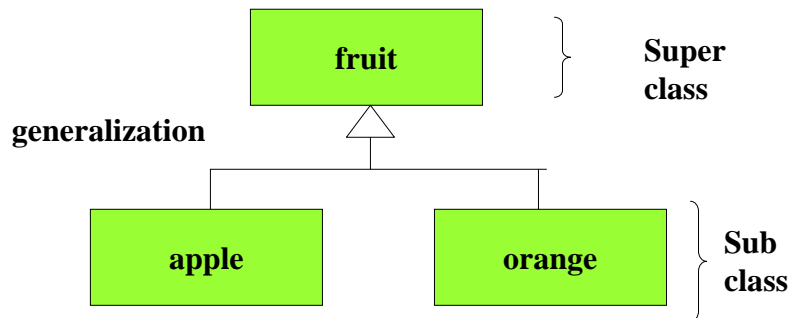
## How can we promote reuse ?

- **Class Inheritance or Sub-classing**
- **Interface Inheritance or Sub-typing**
- **Delegation and Object Composition**

## Super class and Sub class

A generalization shows a close relationship between a general and a specific class.

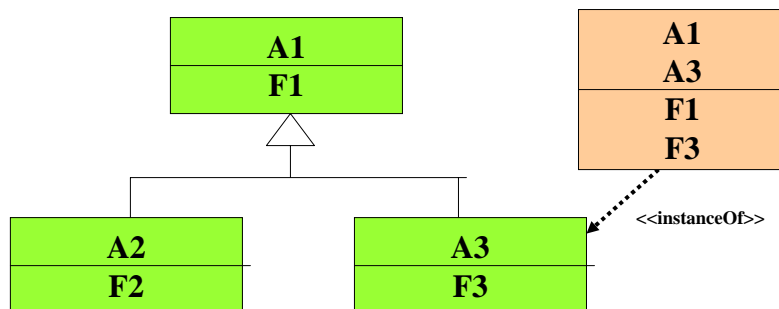
- Apple is-a fruit
- Orange is-a fruit



Ochimizu, Higashida, "Object Modeling", Addison-Wesley Publishers Japan

## Class inheritance

- We need not re-define the attributes and operations which are already defined in a super class.



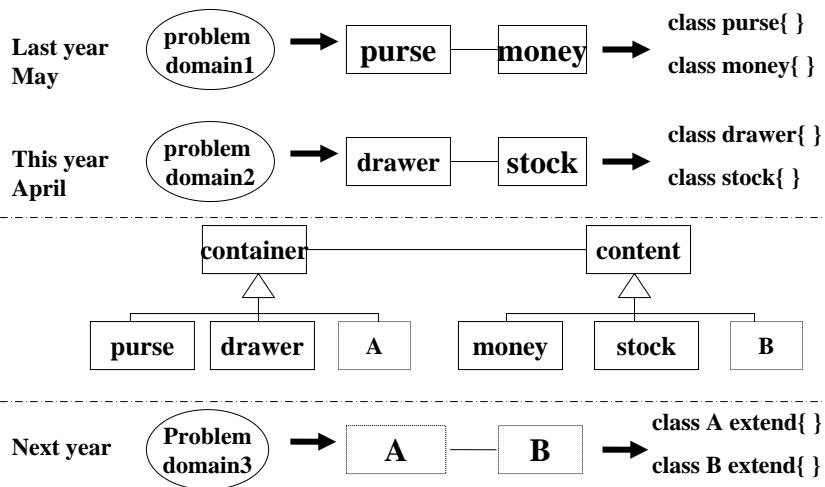
Ochimizu, Higashida, "Object Modeling", Addison-Wesley Publishers Japan

## Class Library

- A class library is a group of classes organized as a tree using “is-a” relationship.

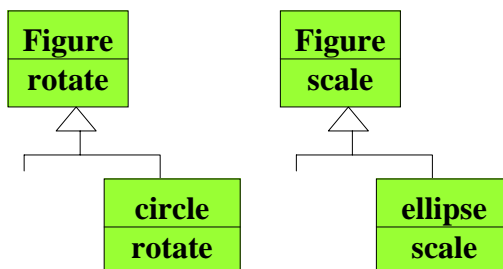
Ochimizu, Higashida, "Object Modeling", Addison-Wesley Publishers Japan

## Knowledge Accumulation by Inheritance

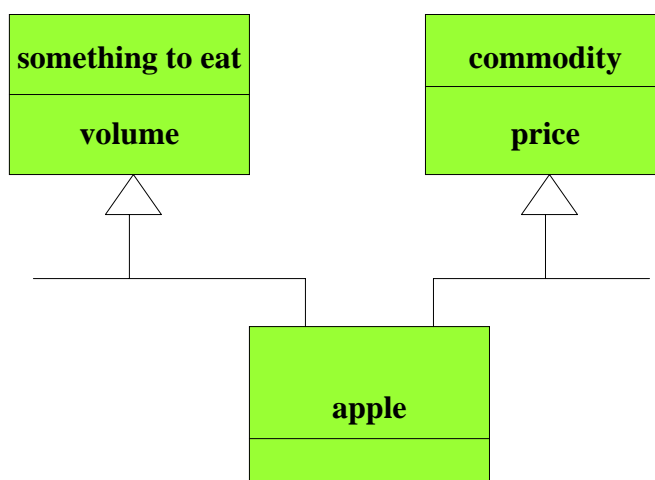


## Overriding

- Extension
- Restriction
- Speed up



## Multiple Inheritance



Ochimizu, Higashida, "Object Modeling", Addison-Wesley Publishers Japan

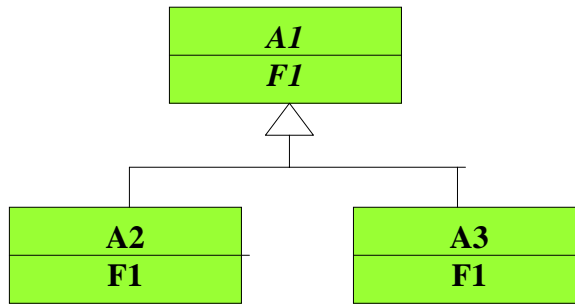
## **Class inheritance and Interface Inheritance**

- **Class inheritance:** copy attributes and operations defined in a super class into its subclass. We only add new attributes and operations specific to the sub class. A sub class may override a super class features (attributes and operations) by defining a feature with the same name.
- **Interface inheritance:** inherit only the signature defined in an abstract operation. We prepare the different implementation of method in each concrete sub class. And we invoke them with the same signature.

## **Abstract class and Interface inheritance**

- An abstract class is a class that has an abstract operation
- An abstract operation only defines a signature but does not define a method.
- A method is defined in a subclass with the same signature.
- We can invoke different methods with the same interface.

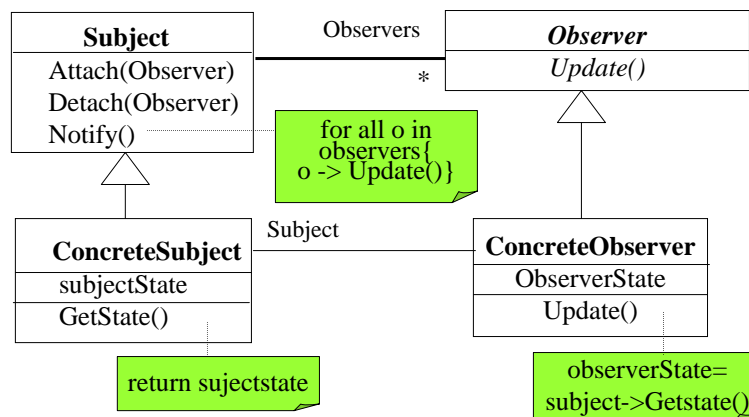
## Interface Inheritance



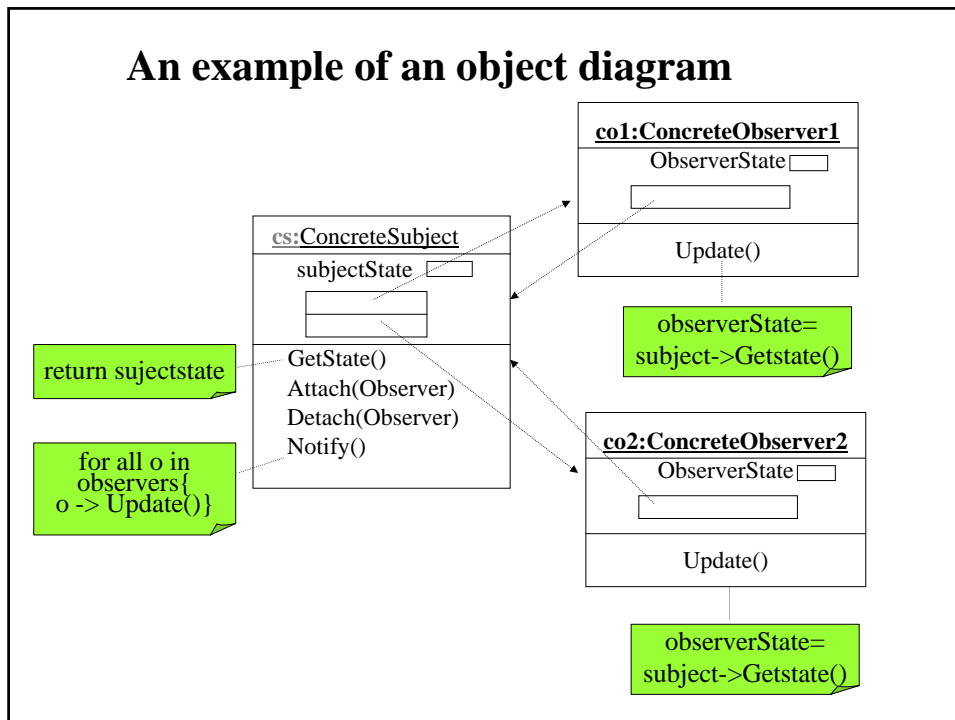
Ochimizu, Higashida, "Object Modeling", Addison-Wesley Publishers Japan

## Observer

"ConcreteSubject notifies its observers whenever a change occurs that could make its observers' state inconsistent with its own. After being Informed of a change in the concrete subject, ConcreteObserver uses this information to reconcile its state with that of the subject." [39]



Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Design Patterns", Addison-Wesley Publishing

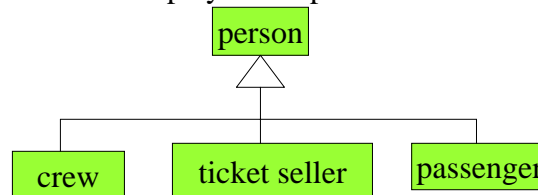


## Class Inheritance and Interface Inheritance

- Class inheritance
  - Define attributes and operations only for the difference between superclass and subclass in Modeling
  - Programming to the difference in Programming
- Interface Inheritance
  - Open Closed Principles (B. Meyer)
  - open to extension
  - closed to modification

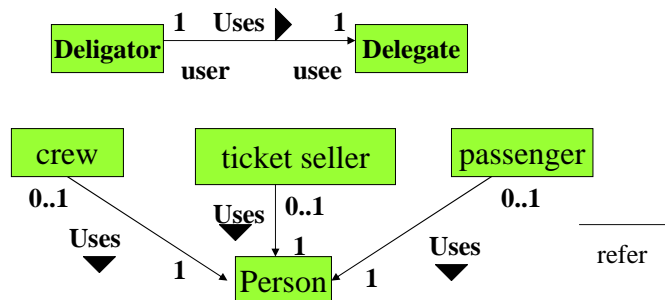
## Delegation

- Inheritance is not almighty
  - **is-a-role-played-by**
- The same person can play the multiple roles.
  - A crew is sometimes a passenger
  - A crew sometimes sells a ticket
- It is ridiculous to define sub classes for all combination
- A person sometimes plays multiple roles



## Object Composition

- We can extend the behavior of class *crew*, *ticket seller* and *passenger* by using delegation and object composition
- Delegation is more general than inheritance.



## Achievements and Issues

### Achievements

**Easy-to-change of Data Structure (Information Hiding)**  
**Programming-to-difference (Class Inheritance)**  
**Easy-to-evolve (Interface inheritance)**

### Topics

**Coarse-grained Reuse**  
**( Design Patterns, Frameworks and Components)**

## 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 super class?**
  2. **What is a generalization?**
  3. **What is a class inheritance?**
  4. **What is an overriding?**
  5. **What is an abstract operation?**
  6. **What is an abstract class?**
  7. **What is an interface inheritance?**
  8. **What is a delegation?**