Introduction

- Object-oriented design.
- Unified Modeling Language (UML).

System modeling

- Need languages to describe systems:
  - useful across several levels of abstraction;
  - understandable within and between organizations.
- Block diagrams are a start, but don't cover everything.

Object-oriented design

- Object-oriented (OO) design: Emphasizes two concepts of importance-
  - encourages the design to be described as a number of interacting objects
  - objects will correspond to real pieces of software or hardware in the system.
- Object = state + methods.
  - State provides each object with its own identity.
  - Methods provide an abstract interface to the object.

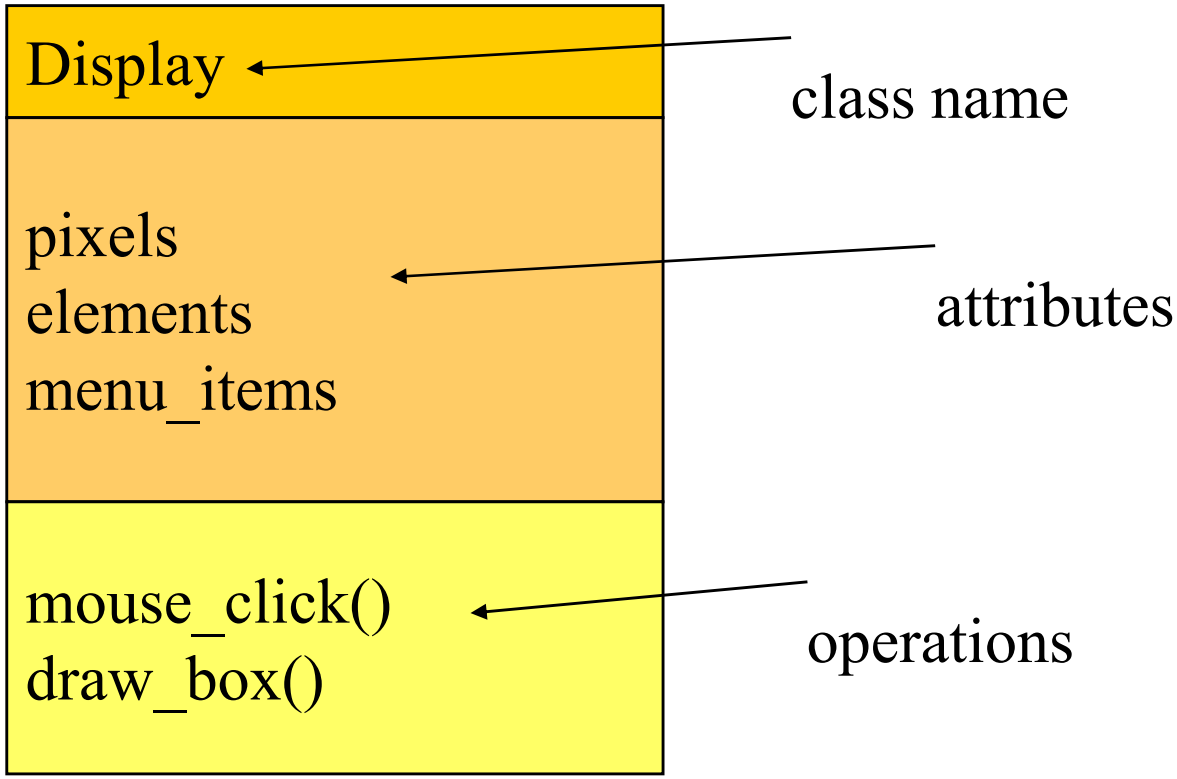
Objects and classes

- Class: object type.
- Class defines the object's state elements but state values may change over time.
- Class defines the methods used to interact with all objects of that type.
  - Each object has its own state.

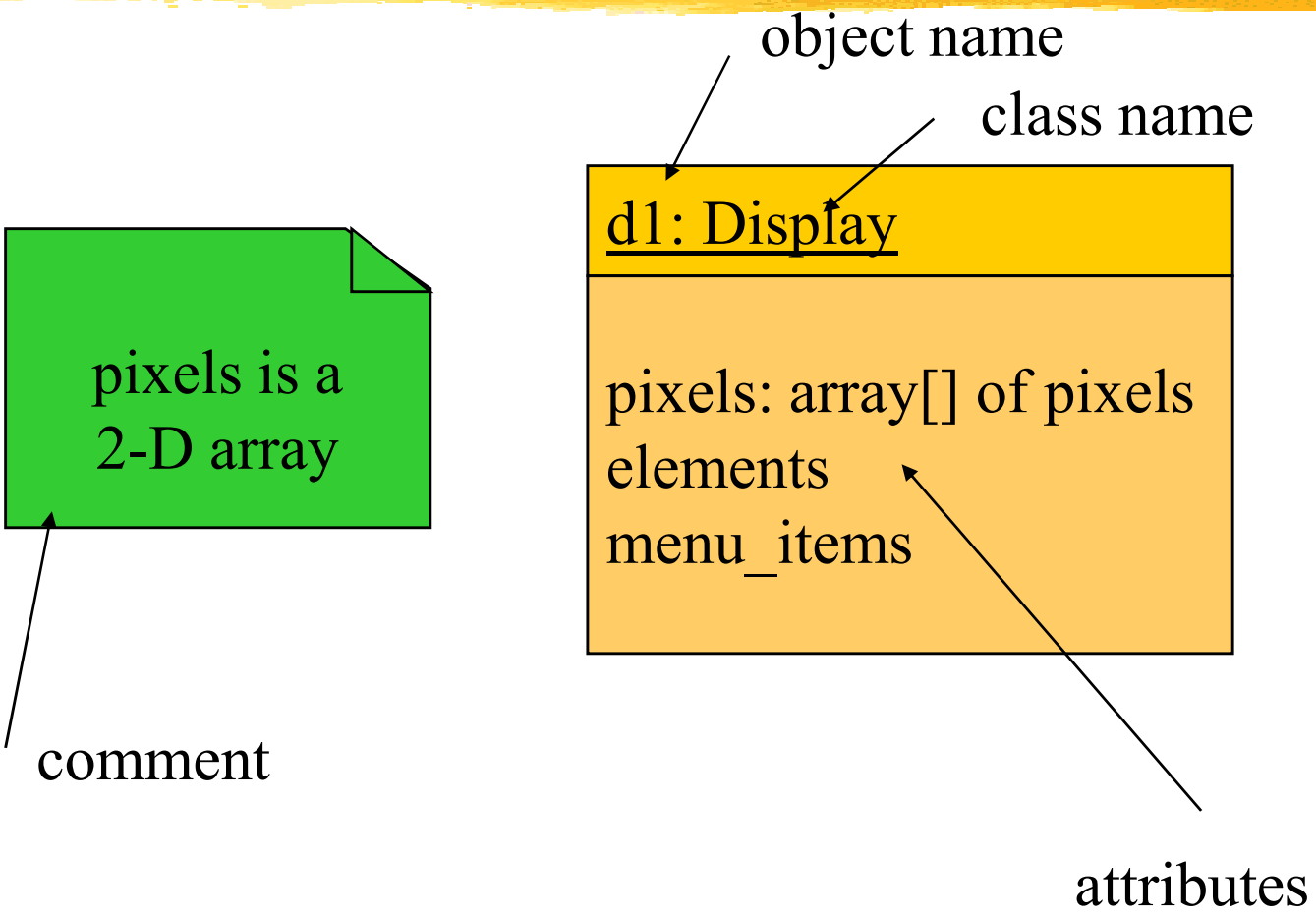
UML

- ***Unified Modeling Language***
- Developed by Booch et al.
- UML is an ***object-oriented modeling language***
- Goals:
  - object-oriented;
  - visual;
  - useful at many levels of abstraction;

UML class notation



UML object notation



The class interface

- A class defines
  - ***interface*** *for a particular type of object and*
  - object's ***implementation.***
- When we use an object, we do not directly manipulate its attributes—we can only read or modify the object's state through the operations that define the interface to the object.

The class interface [contd..]

- Operations may have arguments, return values.
- An operation can examine and/or modify the object's state.

Relationships between objects and classes

- Association:
  - objects communicate but one does not own the other.
  - one-to-one, one-to-many, many-to-one, many-to-many defines an association between objects
  - Symbol $\longrightarrow$
  - e.g A Student and a Faculty are having an association.

Relationships between objects and classes

- Aggregation:
  - a special case of association.
  - an object 'has-a' another object.
  - Symbol 
  - E.g. Relationship between library and student is aggregation.

Relationships between objects and classes

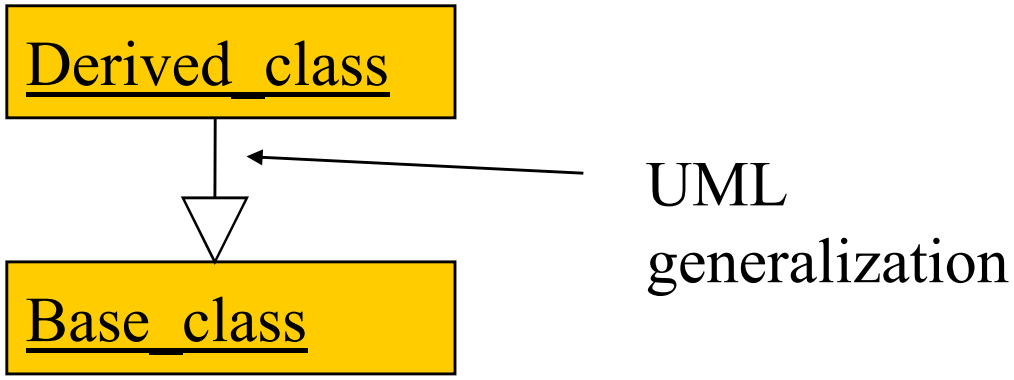
- Composition:
  - a special case of aggregation
  - composed object cannot exist without the other object
  - Symbol 
  - E.g. Relationship between library and book is composition.

Relationships between objects and classes

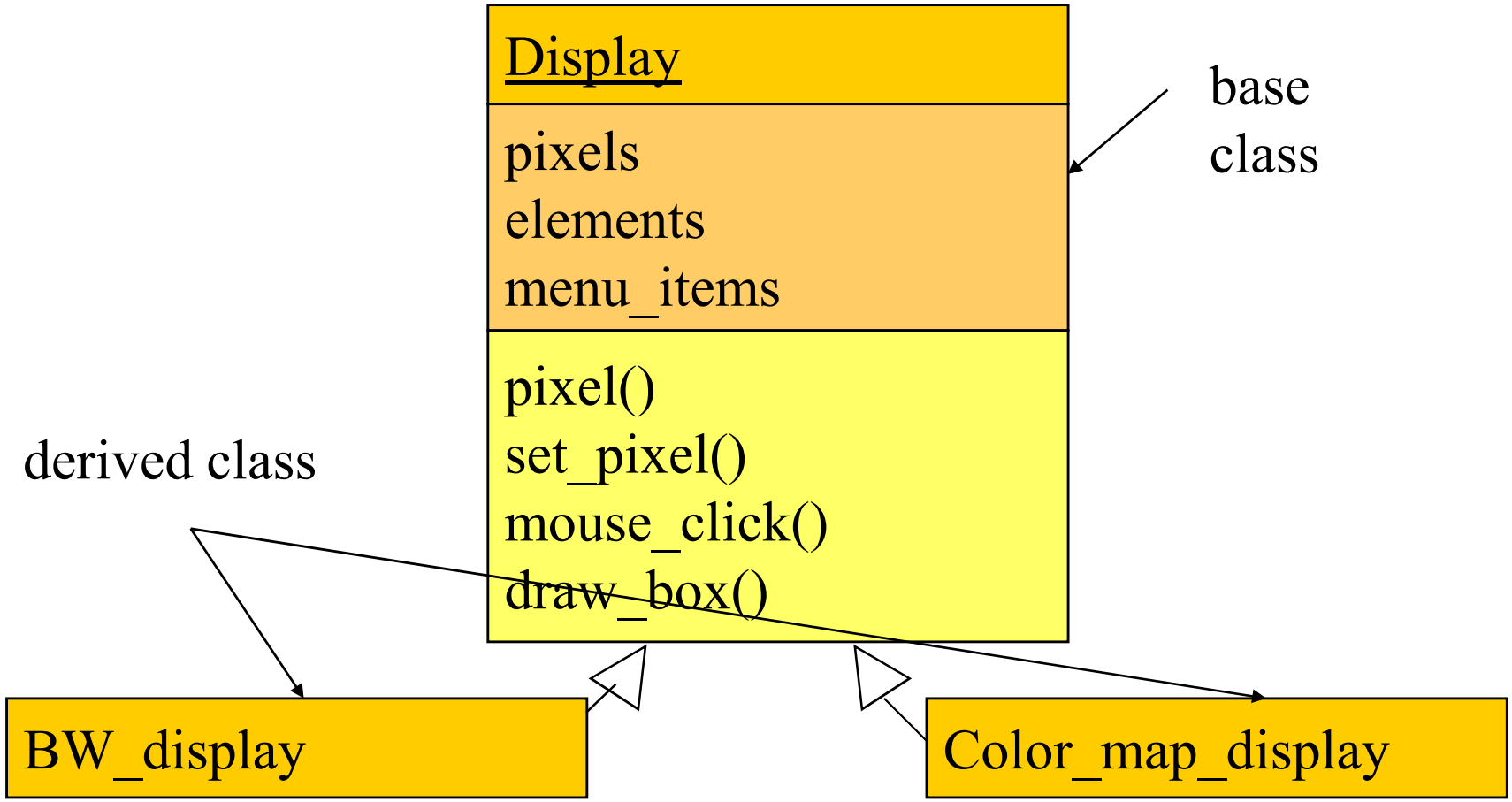
- Generalization:
  - define one class in terms of another.
  - uses a "is-a" relationship.
  - Symbol 
  - A student is a person. A faculty is a person.

Class derivation

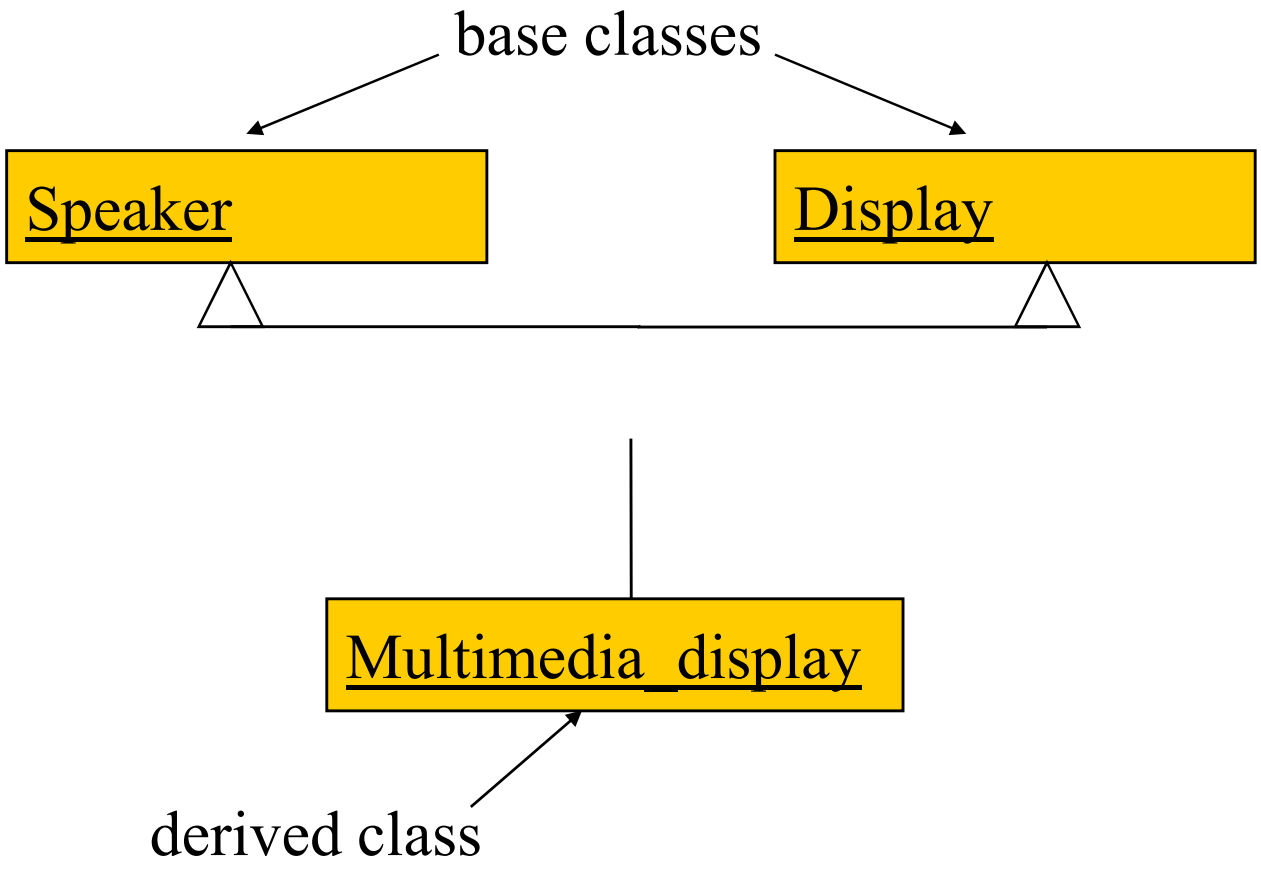
- May want to define one class in terms of another.
  - Derived class inherits attributes, operations of base class.



Class derivation example



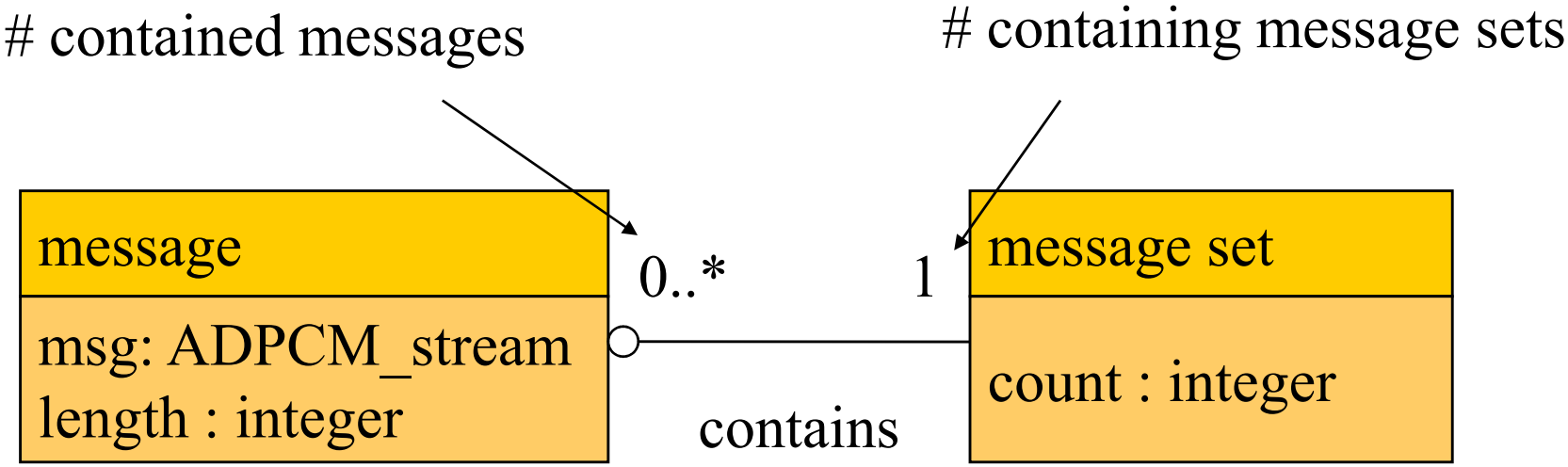
Multiple inheritance



Links and associations

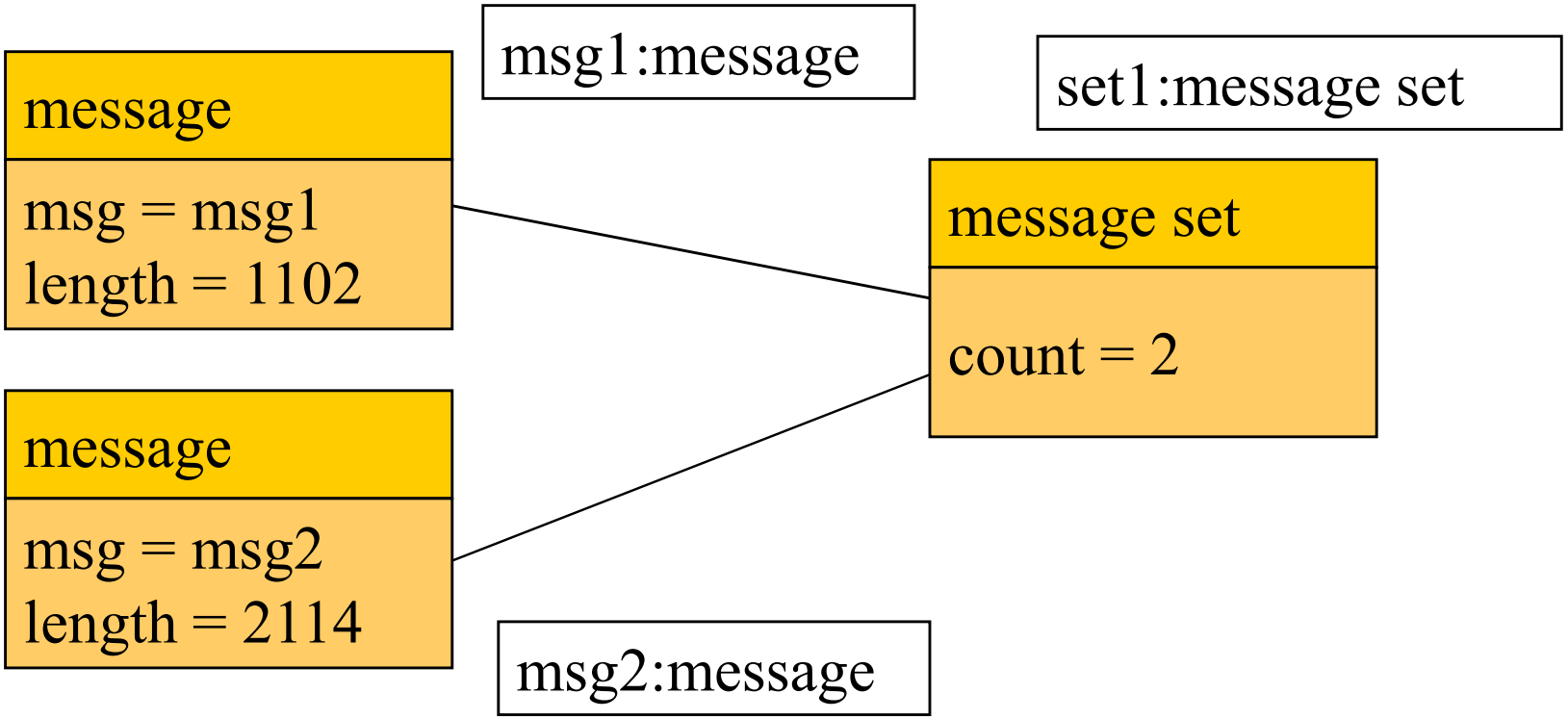
- Link: describes relationships between objects.
- Association: describes relationship between classes.

Association example



Link example

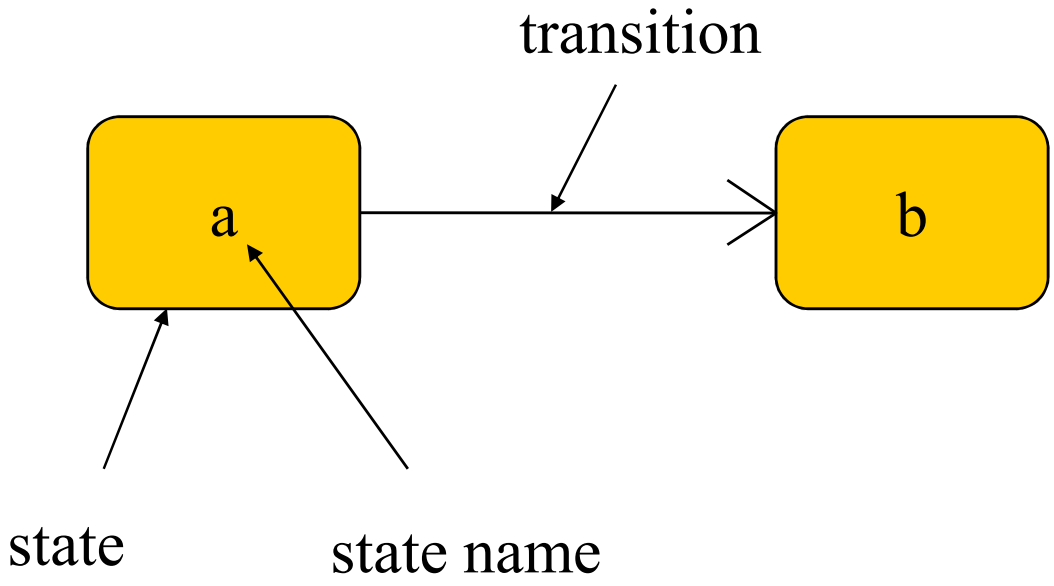
- Link defines the contains relationship:



Behavioral description

- Behaviour of a system can be specified through **State machine.**
- Change from one state to another are triggered by the occurrence of ***events.***
- Behavior can be defined-
  - Internally
  - Externally

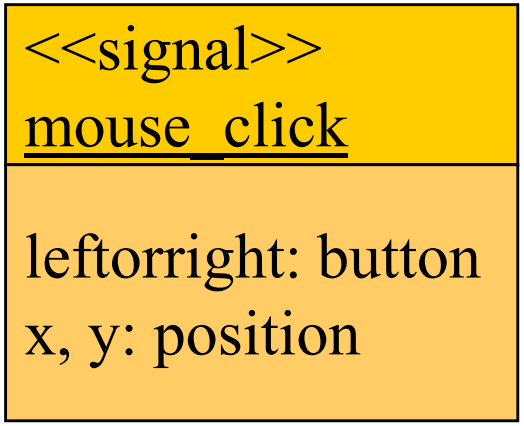
State machines



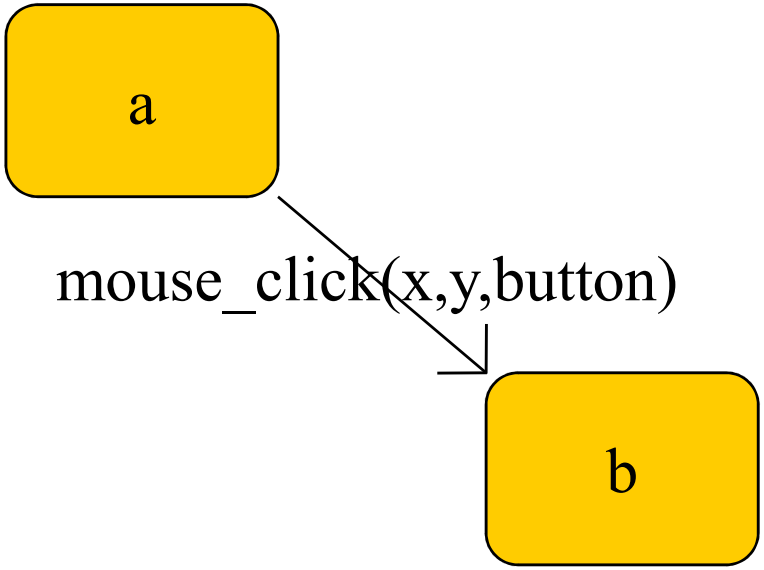
Types of events

- Signal: asynchronous event.
- Call: synchronized communication.
- Timer: activated by time.

Signal event

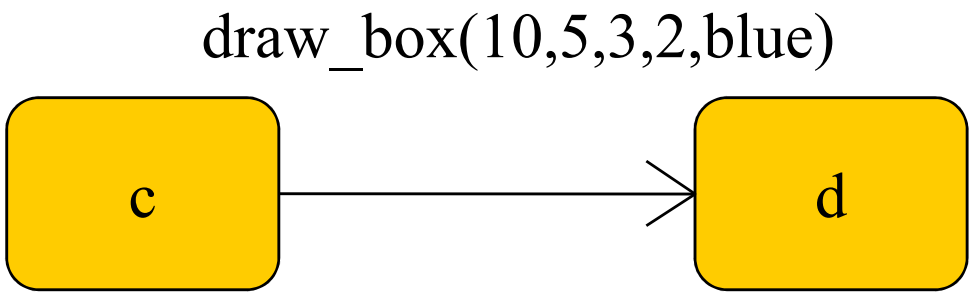


declaration

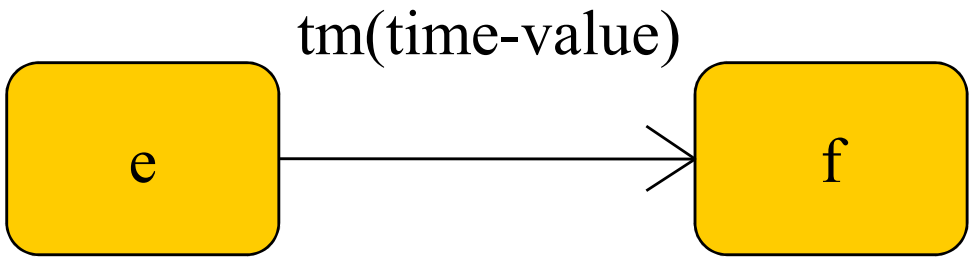


event description

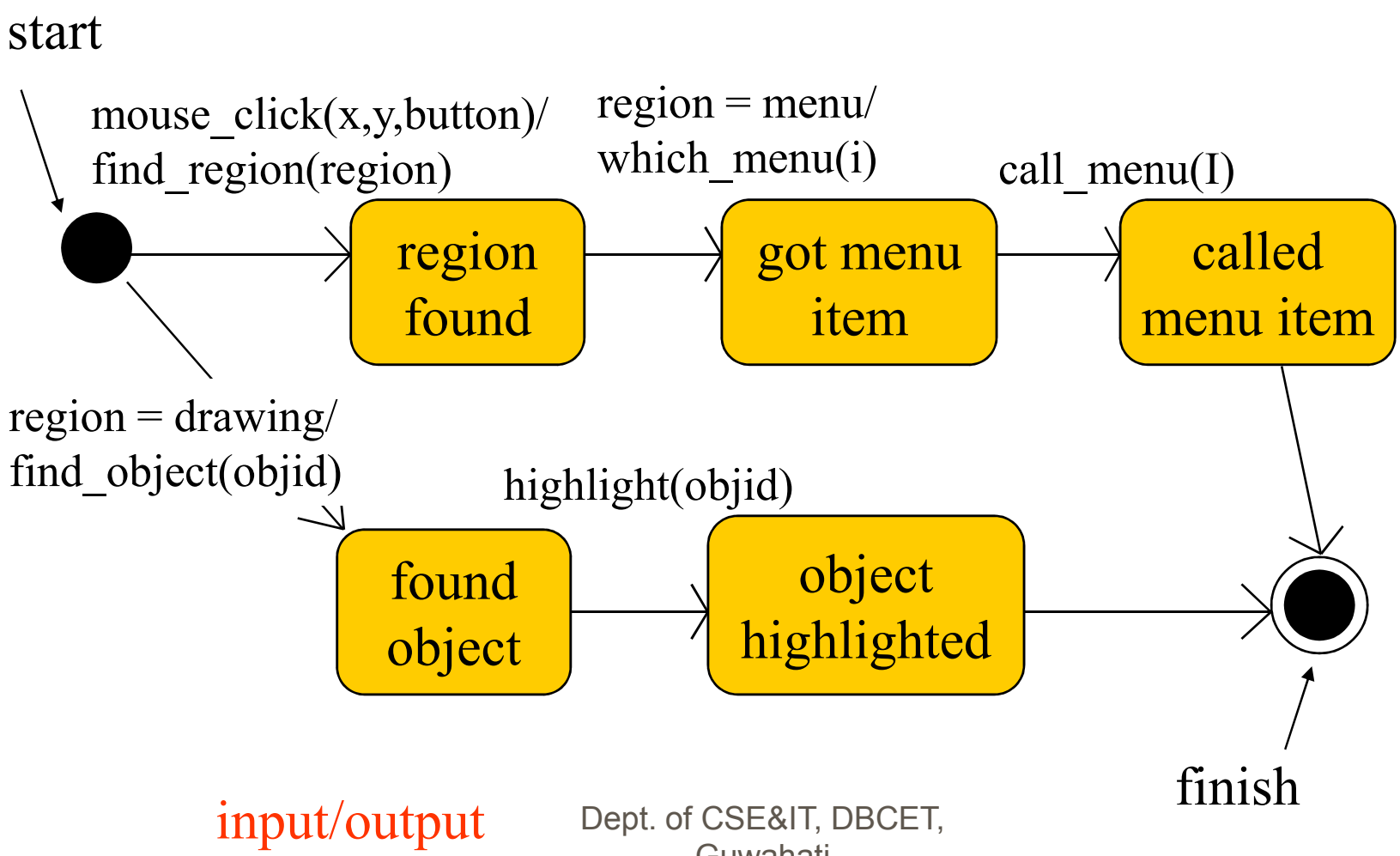
Call event



Timer event



Example state machine

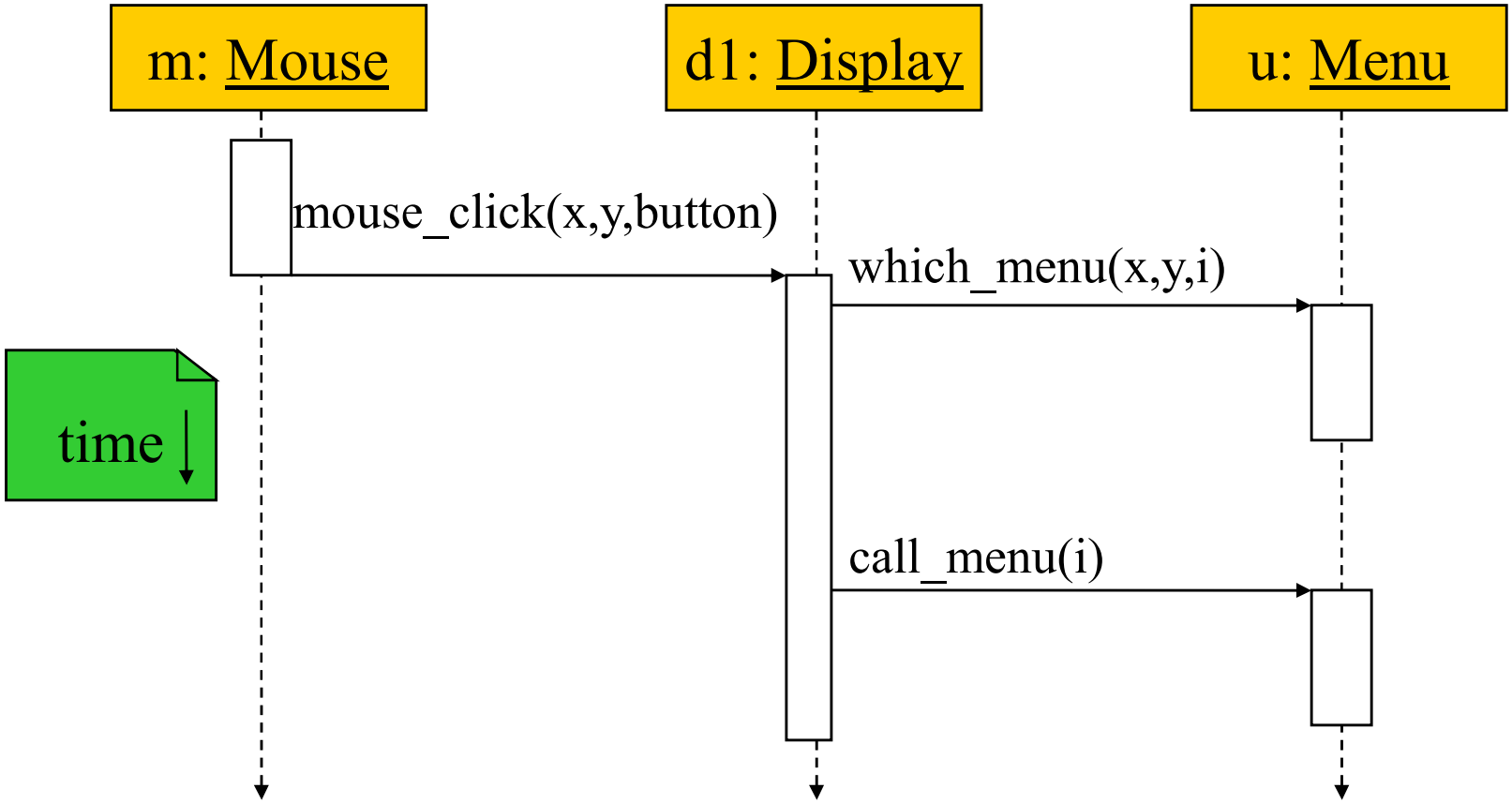


input/output

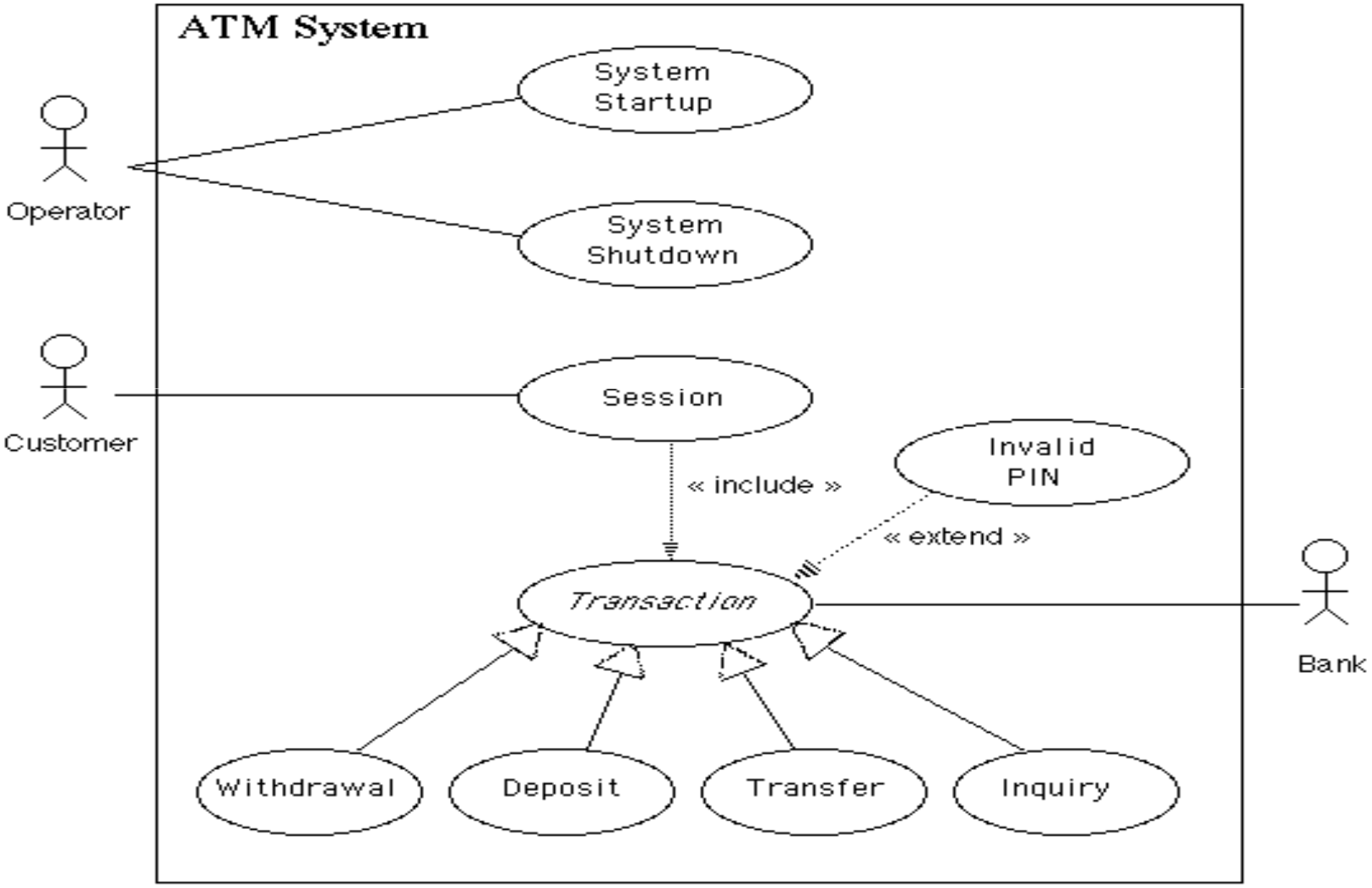
Sequence diagram

- Shows sequence of operations over time.
- Relates behaviors of multiple objects.

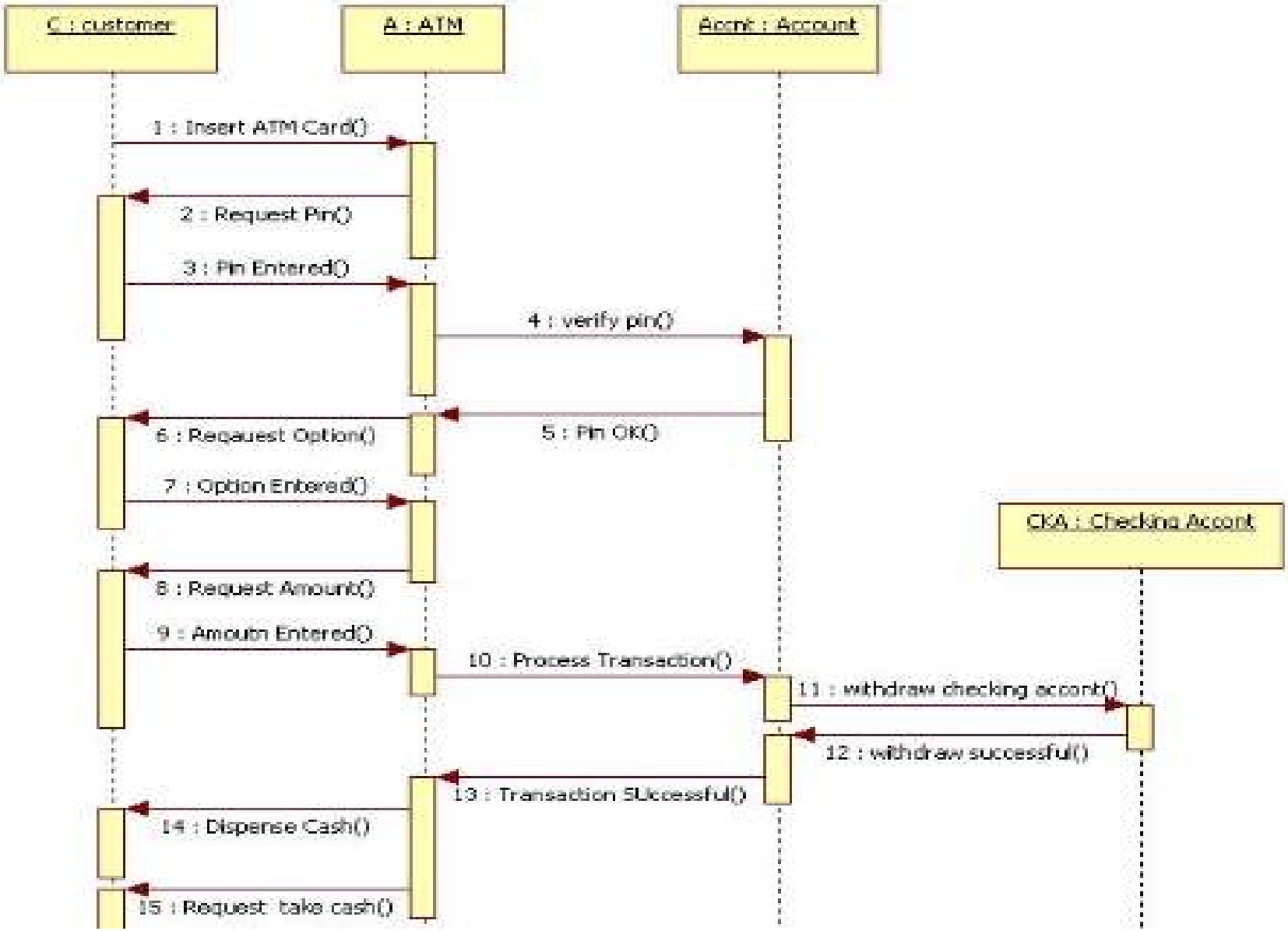
Sequence diagram example



ATM – System Use Case Diagram



Sequence diagram - ATM System



Summary

- Object-oriented design helps us organize a design.
- UML is a transportable system design language.
  - Provides structural and behavioral description primitives.