

CSC444F: Software Engineering I

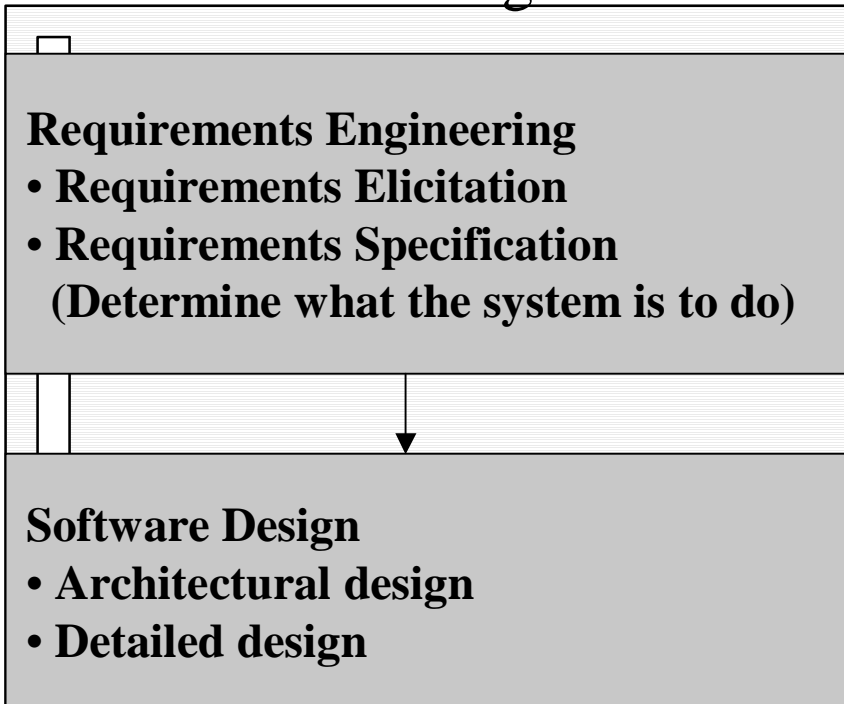
Mou Hu
mou.hu@utoronto.ca

Lecture 10: Object-Oriented Analysis and Design (I)

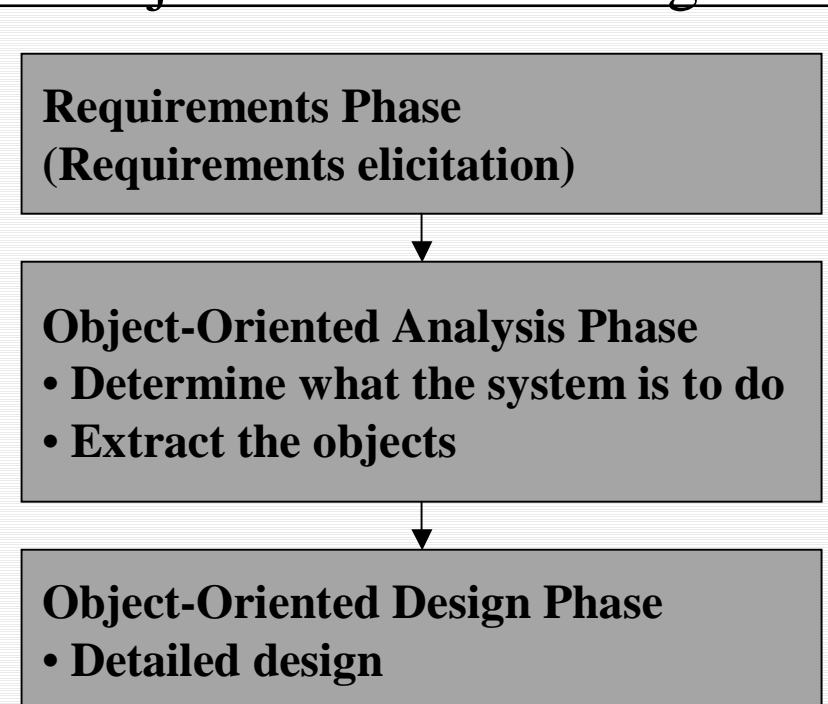
- Object-Oriented Life Cycle Model
- Concepts of Object-Orientation
- Object-Oriented Analysis and Design Notations
- Reading: Chapter 12

Object-Oriented Life Cycle Model Compared with Structured Paradigm

Structured Paradigm



Object-Oriented Paradigm



● The object-oriented paradigm is an integrated approach: the transition from phase to phase is far smoother than with structured paradigm.

Object-Oriented Concepts

■ Objects

- an entity that has state, attributes and services

■ Classes

- Provide a way of grouping objects with similar attributes or services
- Classes form an abstraction hierarchy through 'is_a' relationships

■ Attributes

- Together represent an object's state

■ Relationships

- 'is_a' classification relations
- 'part_of' assembly relationships
- 'associations' between classes

■ Methods (services, functions)

- These are the operations that all objects in a class can do...
- ...when called on to do so by other objects

■ Message Passing

- How objects invoke services of other objects

Source: Ref. [3]

Key Principles

■ Classification (using inheritance)

- Classes capture commonalities of a number of objects

- Each subclass inherits attributes and methods from its parent

- Forms an 'is_a' hierarchy

- Child class may 'specialize' the parent class

- by adding additional attributes & methods

- by replacing an inherited attribute or method with another

- Multiple inheritance is possible where a class is subclass of several different superclasses.

■ Information Hiding

- internal state of an object needs not be visible to external viewers

- Objects can encapsulate other objects, and keep their services internal

- useful for forming abstractions

■ Aggregation

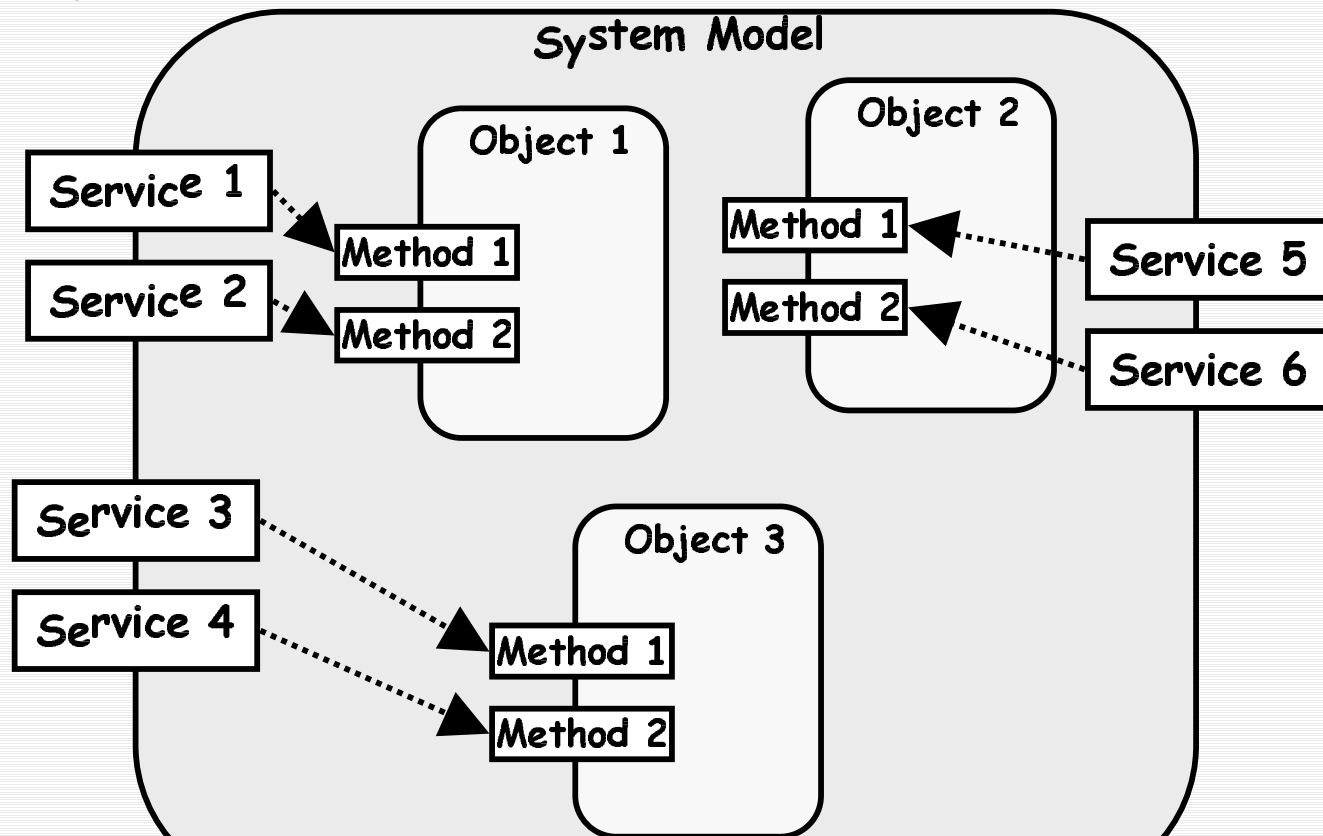
- Can describe relationships between parts and the whole

Source: Ref. [3]

Information Hiding

- Objects can contain other objects

- (compare with hierarchies of dataflow diagram in Structured Analysis)



Source: Ref. [3]

Unified Modeling Language (UML)

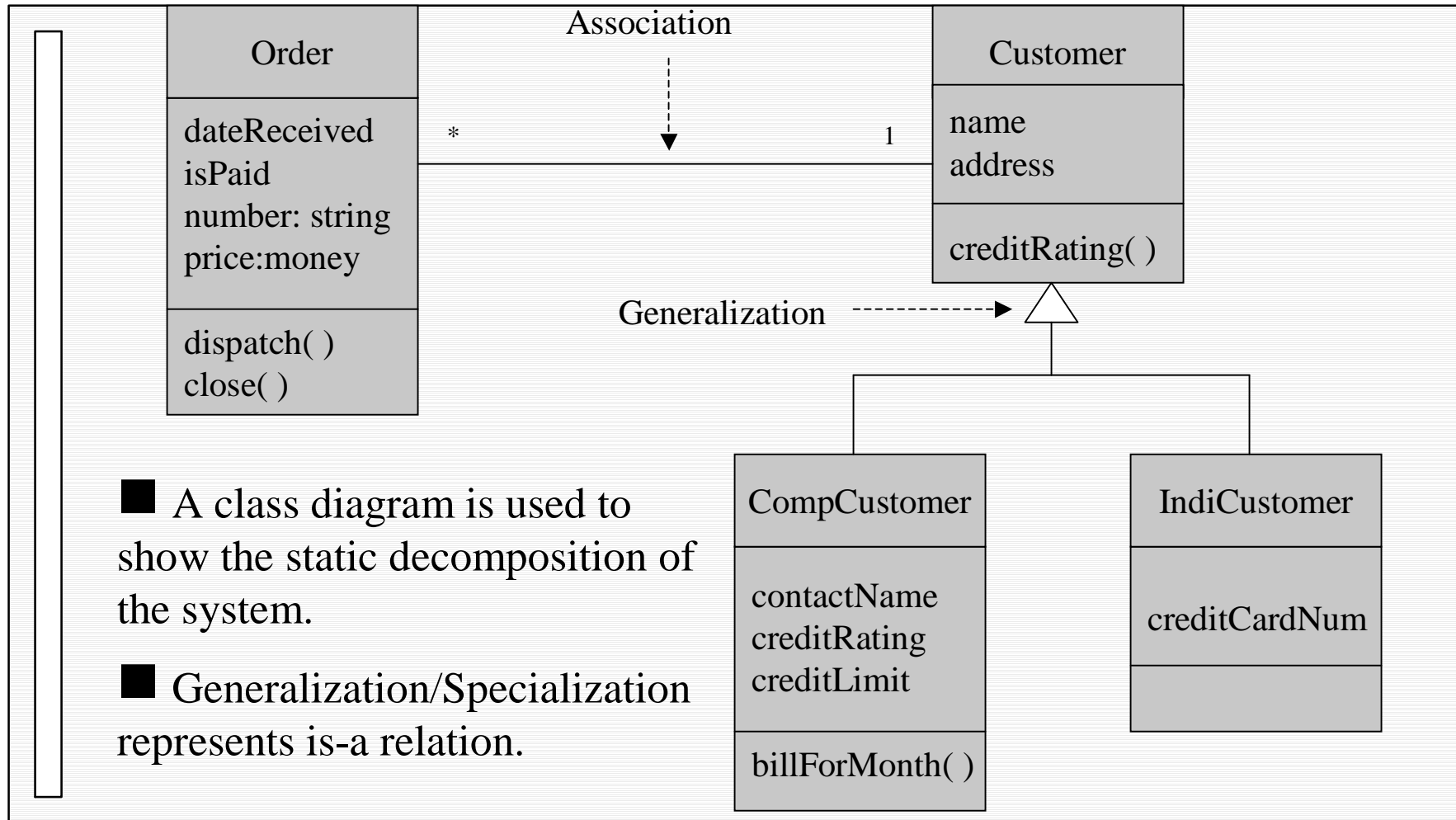
■ Third generation OO method

- Is purely a notation
 - No modeling method associated with it!
- But has been accepted as a standard for OO modeling
 - But is primarily owned by Rational Corp. (who sell lots of UML tools and services)

■ Notations

- Use case diagrams and CRC cards
- Class diagrams
- State diagrams
- Sequence diagrams
- Collaboration diagrams

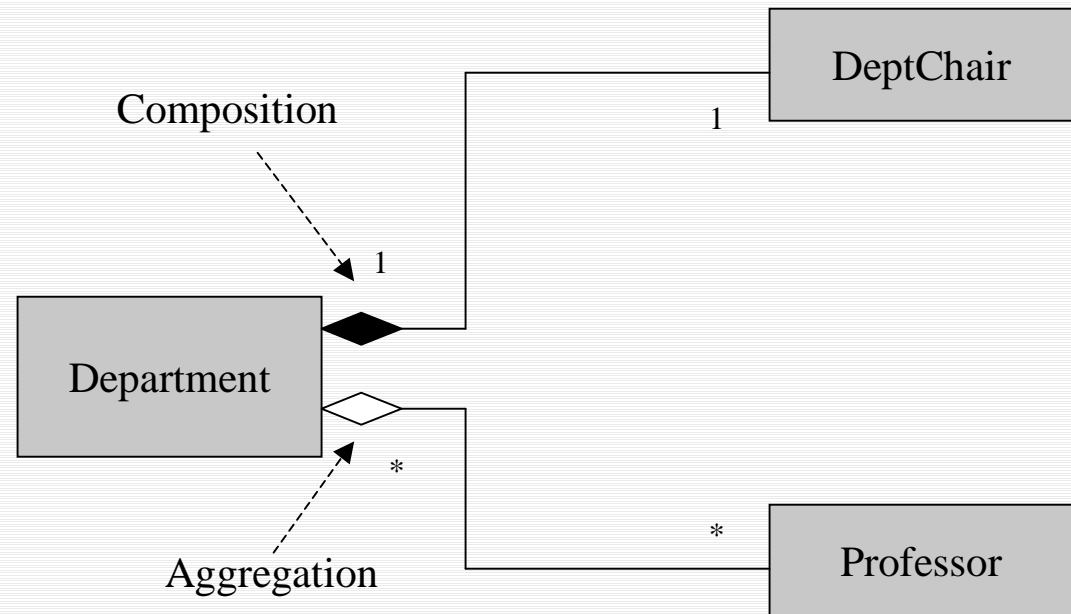
Class Diagram: Association and Generalization/Specialization



- A class diagram is used to show the static decomposition of the system.
- Generalization/Specialization represents is-a relation.

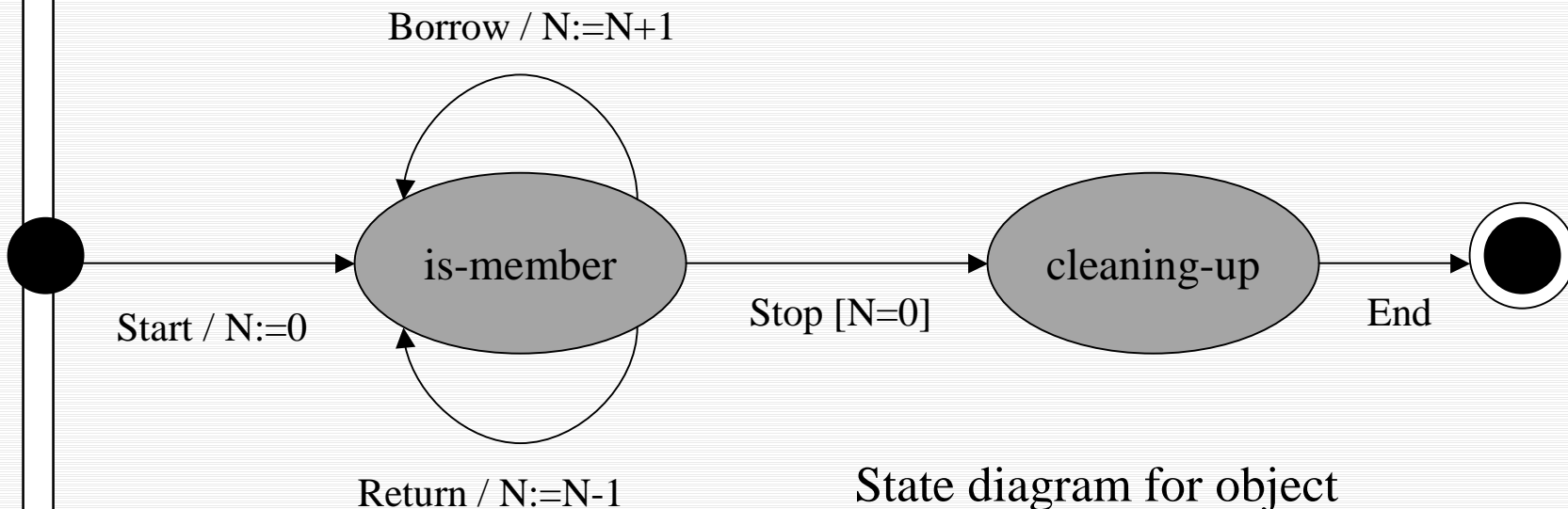
Class Diagram: Composition and Aggregation

- Aggregation is a part-of relation.
- Composition is a strong notion of aggregation, in which the part object may belong to only one whole object.
- With composition, the parts are expected to live and die with the whole.



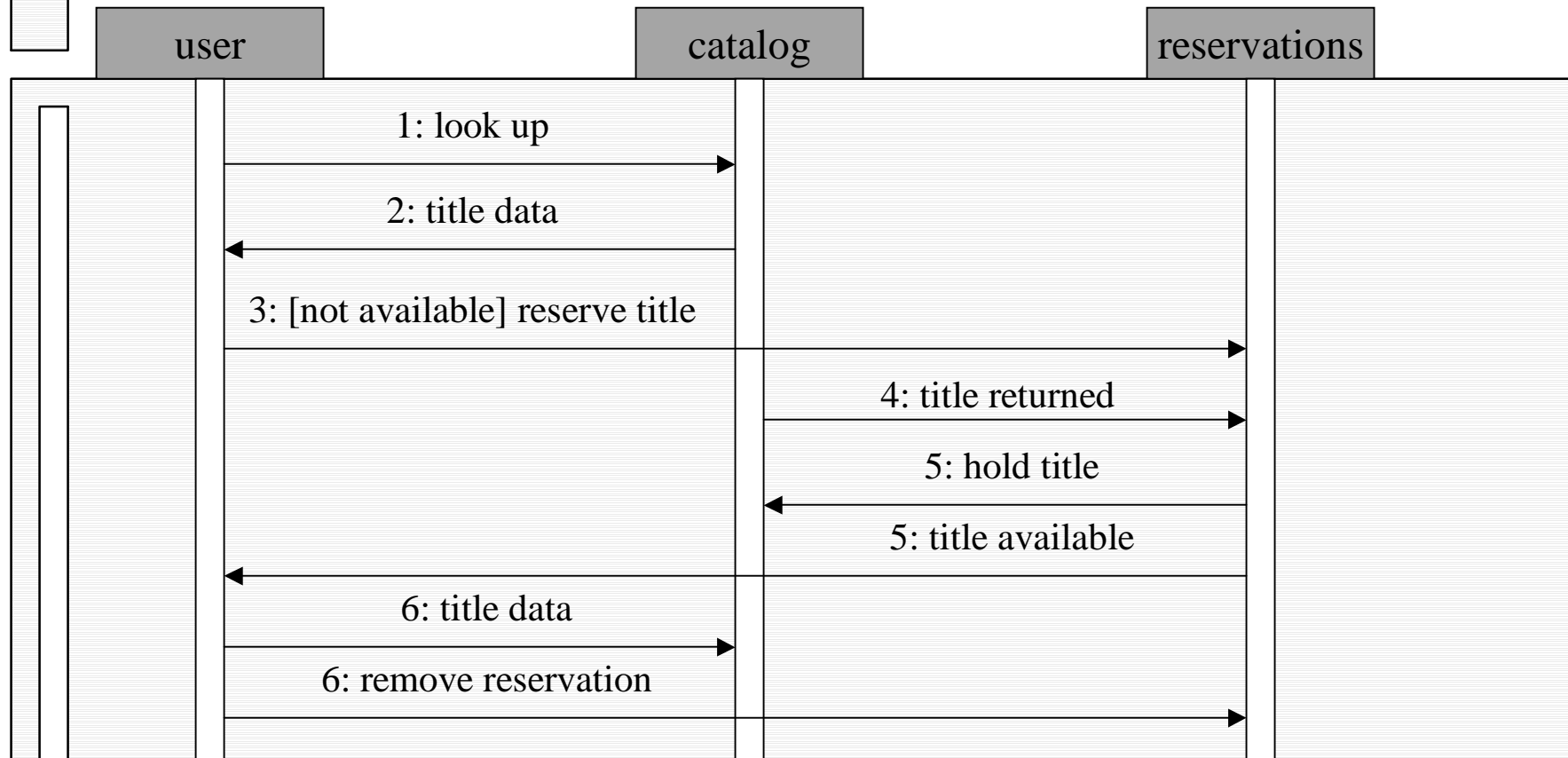
State Diagram

- A state diagram is used to depict the dynamic behavior of the objects.
- A state diagram is an extended finite state diagram (i.e., it has local variables).



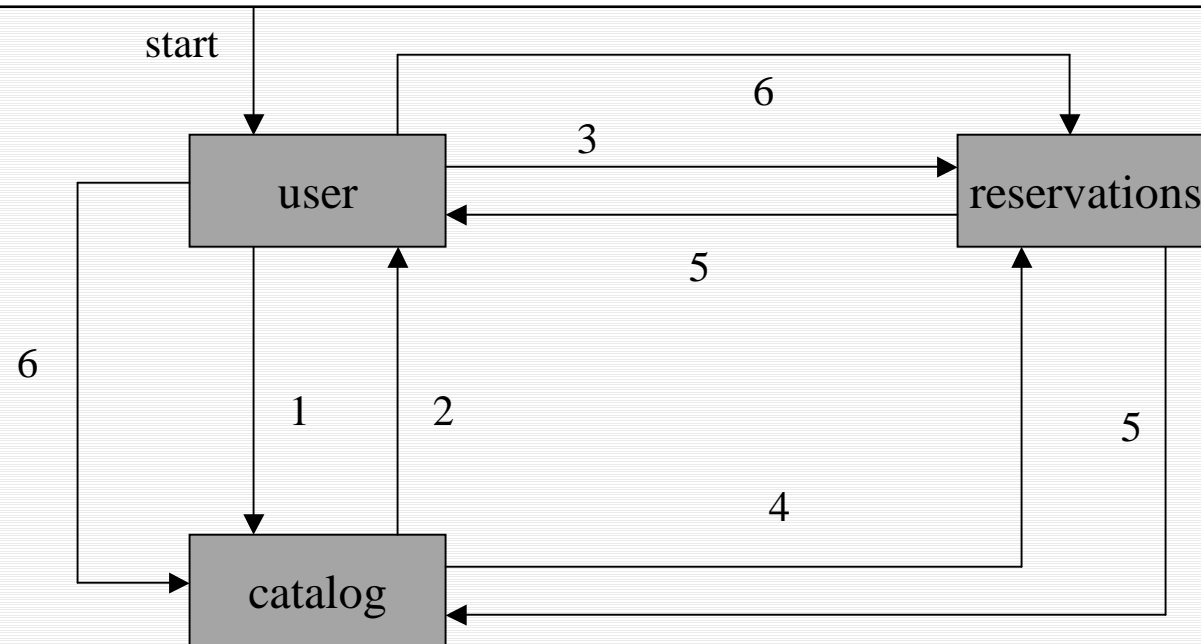
State diagram for object
Member of a library system

Sequence Diagram



■ A sequence diagram shows typical interactions consisting an ordered sequence of messages.

Collaboration Diagram



- A collaboration diagram is another way of showing the interaction between a number of related objects.
- A collaboration diagram is just a rearrangement of the sequence diagram.

References

- [1] Hans van Vliet, “Software Engineering: Principles and Practice”, John Wiley and Sons, Ltd., 2000.
- [2] Stephen R. Schach, “Object-Oriented and Classical Software Engineering”, McGraw-Hill Companies, Inc., 2002.
- [3] Steve Easterbrook, “Lecture Notes”, University of Toronto, 2001.
- [4] Martin Fowler and Kendall Scott, “UML Distilled: A Brief Guide to the Standard Object Modeling Language”, Addison Wesley Longman, Inc., 2000.