Systems Analysis and Design 11th Edition

Chapter 6
Object Modeling

Chapter Objectives

- Explain how object-oriented analysis can be used to describe an information system
- Define object modeling terms and concepts, including objects, attributes, methods, messages, classes, and instances
- Explain relationships among objects and the concept of inheritance
- Draw an object relationship diagram

Chapter Objectives (Cont.)

- Describe Unified Modeling Language (UML) tools and techniques including use cases, use case diagrams, class diagrams, sequence diagrams, state transition diagrams, and activity diagrams
- Explain the advantages of using CASE tools in developing the object model
- Explain how to organize an object model

Overview of Object-Oriented Analysis

- O-O methodology is popular because it integrates easily with object-oriented programming languages such as Java, Smalltalk, VB.Net, Python, and Perl
- Programmers also like O-O code because it is modular, reusable, and easy to maintain
- The end product of O-O analysis is an object model
  - Object model: Represents the information system in terms of objects and O-O concepts

Overview of Object-Oriented Analysis (Cont.1)

- **Object-Oriented Terms and Concepts**
  - Unified modeling language (UML)
    - Method of visualizing and documenting an information system
  - Attributes: Characteristics that describe an object
  - Methods: Tasks or functions that the object performs
  - Message: Command to perform a specific function
  - A class is a group of similar objects
    - Instance: Specific member of a class

Overview of Object-Oriented Analysis (Cont.2)

- **Objects**
  - Represented as a rectangle
    - The object name is at the top, followed by the object's attributes and methods

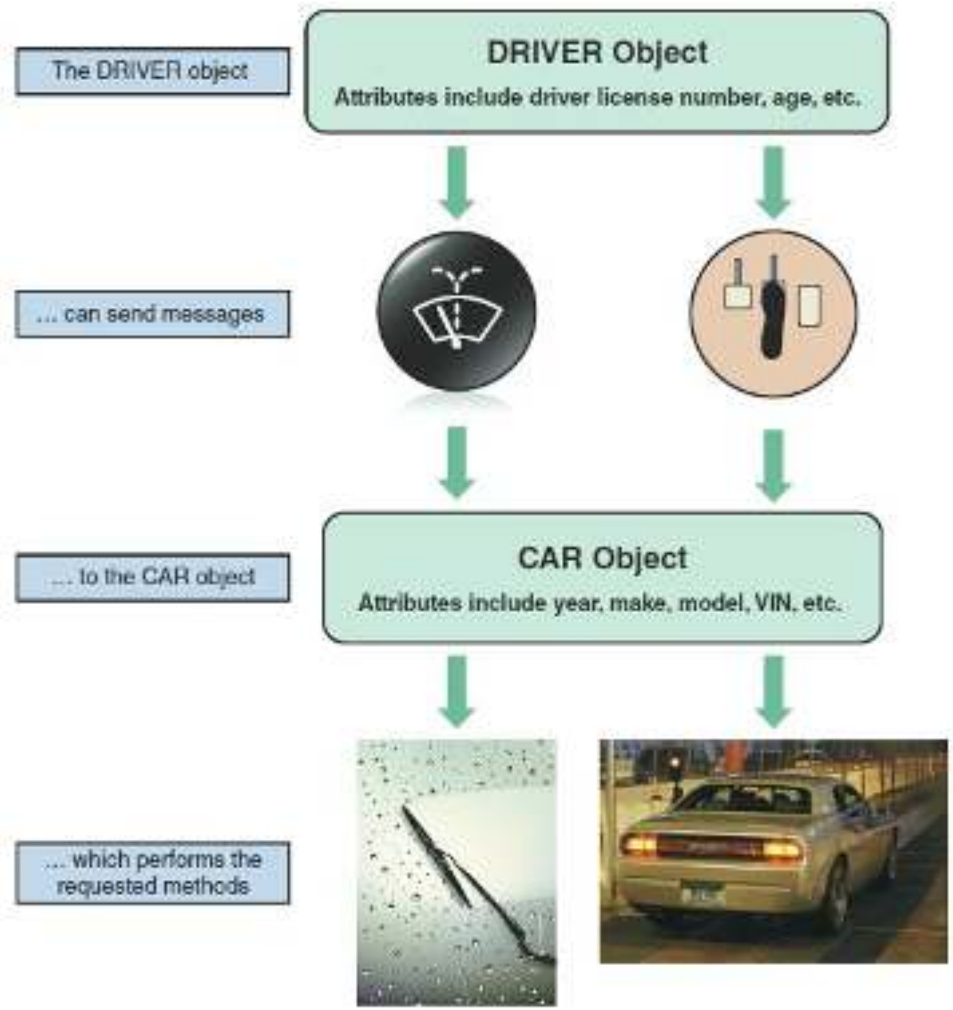


FIGURE 6-1 Objects have attributes, can send and receive messages, and perform actions called methods.

Overview of Object-Oriented Analysis (Cont.3)

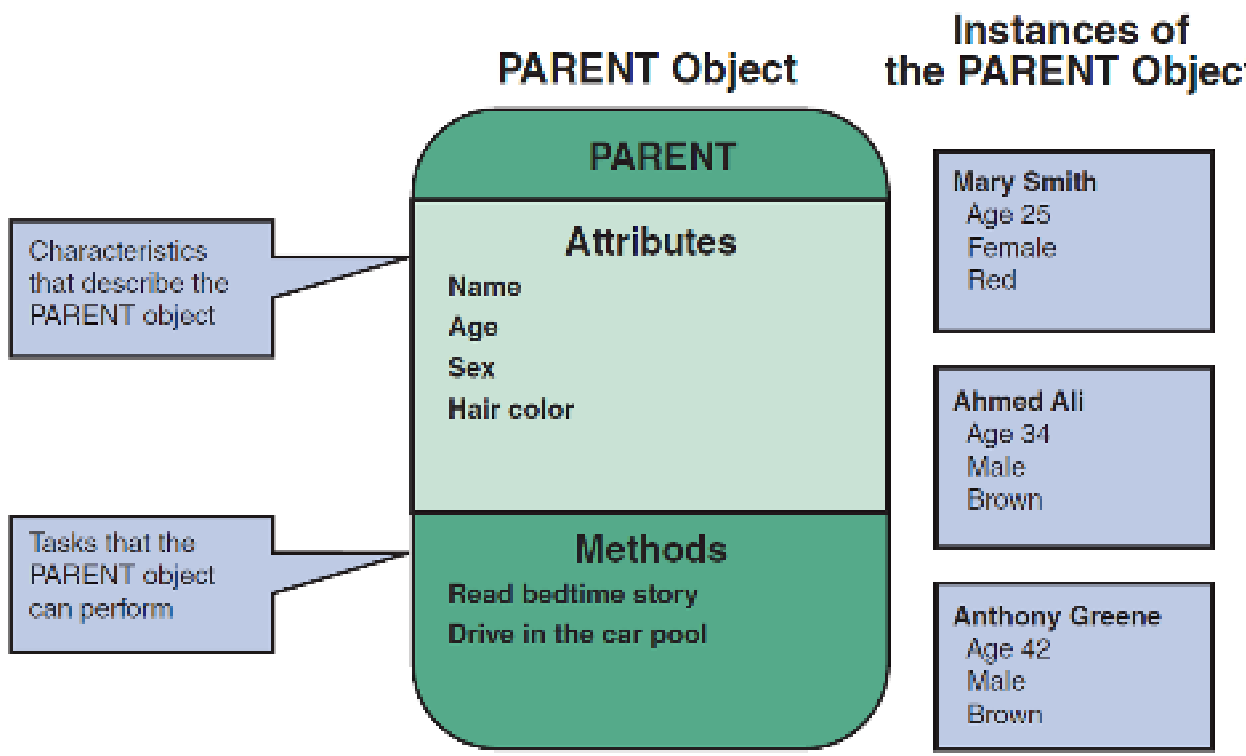


FIGURE 6-2 The PARENT object includes four attributes and two methods. Mary Smith, Ahmed Ali, and Anthony Greene are instances of the PARENT object.

Overview of Object-Oriented Analysis (Cont.4)

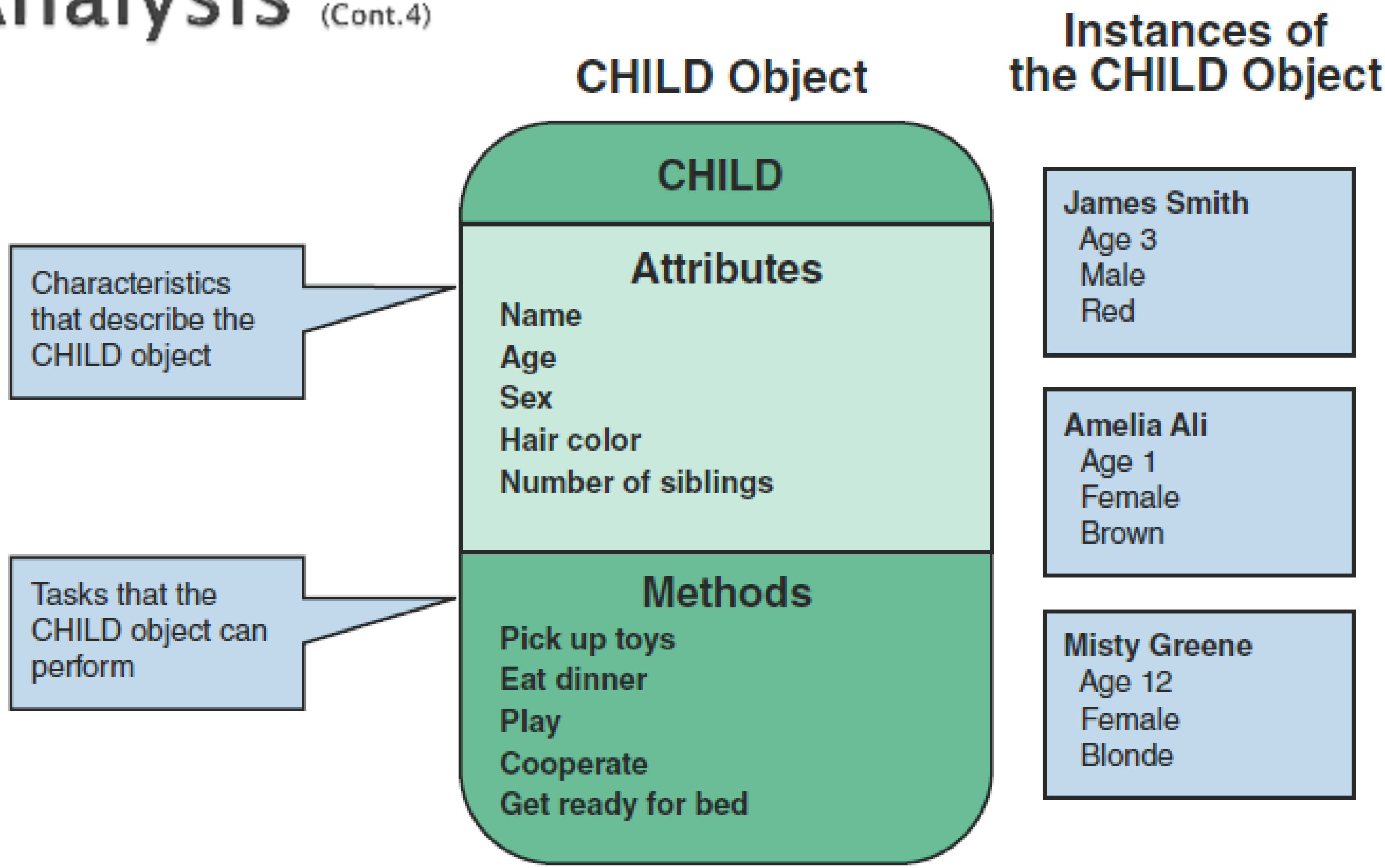


FIGURE 6-3 The CHILD object includes five attributes and five methods. James Smith, Amelia Ali, and Misty Greene are instances of the CHILD object.

Overview of Object-Oriented Analysis (Cont.5)

- **Attributes**
  - Describe the characteristics of an object
  - The number of attributes required depends on:
    - Business requirements of the information system
    - Requirements of users
  - Attributes of an object are defined during the system development process
  - Objects possess a state
    - **State**: Describes the object's current status

Overview of Object-Oriented Analysis (Cont.6)

- **Methods**
  - Specific tasks that an object can perform
  - Identify functions performed
  - Describe the functions performed

Method:	Steps:
MORE FRIES	1. Heat oil 2. Fill fry basket with frozen potato strips 3. Lower basket into hot oil 4. Check for readiness 5. When ready raise basket and let drain 6. Pour fries into warming tray 7. Add salt

FIGURE 6-4 The MORE FRIES method requires the server to perform seven specific steps.

Overview of Object-Oriented Analysis (Cont.7)

- **Message**
  - A command that tells an object to perform a certain method
  - **Polymorphism**: Concept that a message gives different meanings to different objects

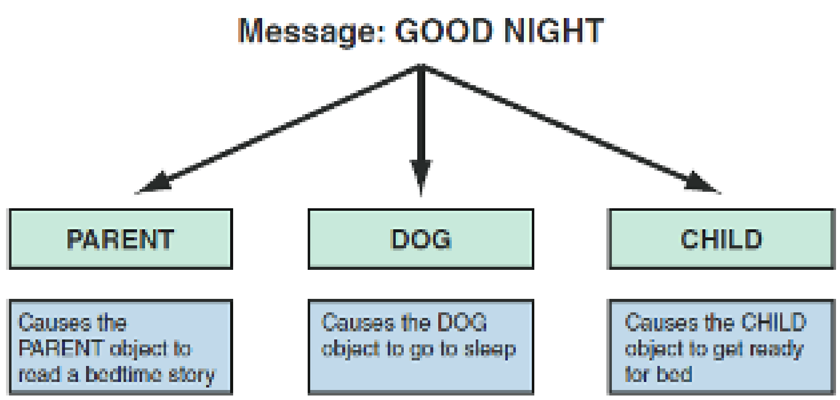


FIGURE 6-5 In an example of polymorphism, the message GOOD NIGHT produces different results, depending on which object receives it.

Overview of Object-Oriented Analysis (Cont.8)

- **Message** (cont.)
  - A message to the object triggers changes within the object without specifying how the changes must be carried out
    - An object can be viewed as black box
  - **Encapsulation**: Idea that all data and methods are self-contained, as in a black box

Figure 6-6 In a school information system, an INSTRUCTOR object sends an ENTER GRADE message to an instance of the STUDENT RECORD class.

Overview of Object-Oriented Analysis (Cont.9)

- Classes
  - An object belongs to a group or category called a class
    - All objects within a class share common attributes and methods
  - **Subclasses**: Categories within a class
  - **Super-class**: A class belonging to a general category

Overview of Object-Oriented Analysis (Cont.10)

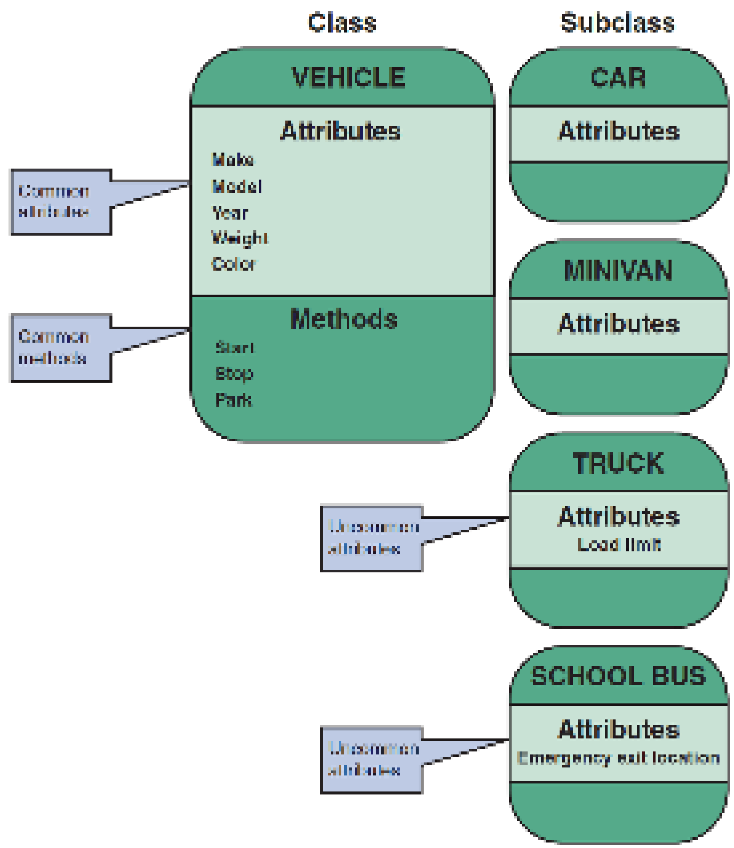


FIGURE 6-7 The VEHICLE class includes common attributes and methods. CAR, TRUCK, MINIVAN, and SCHOOL BUS are instances of the VEHICLE class.

Overview of Object-Oriented Analysis (Cont.11)

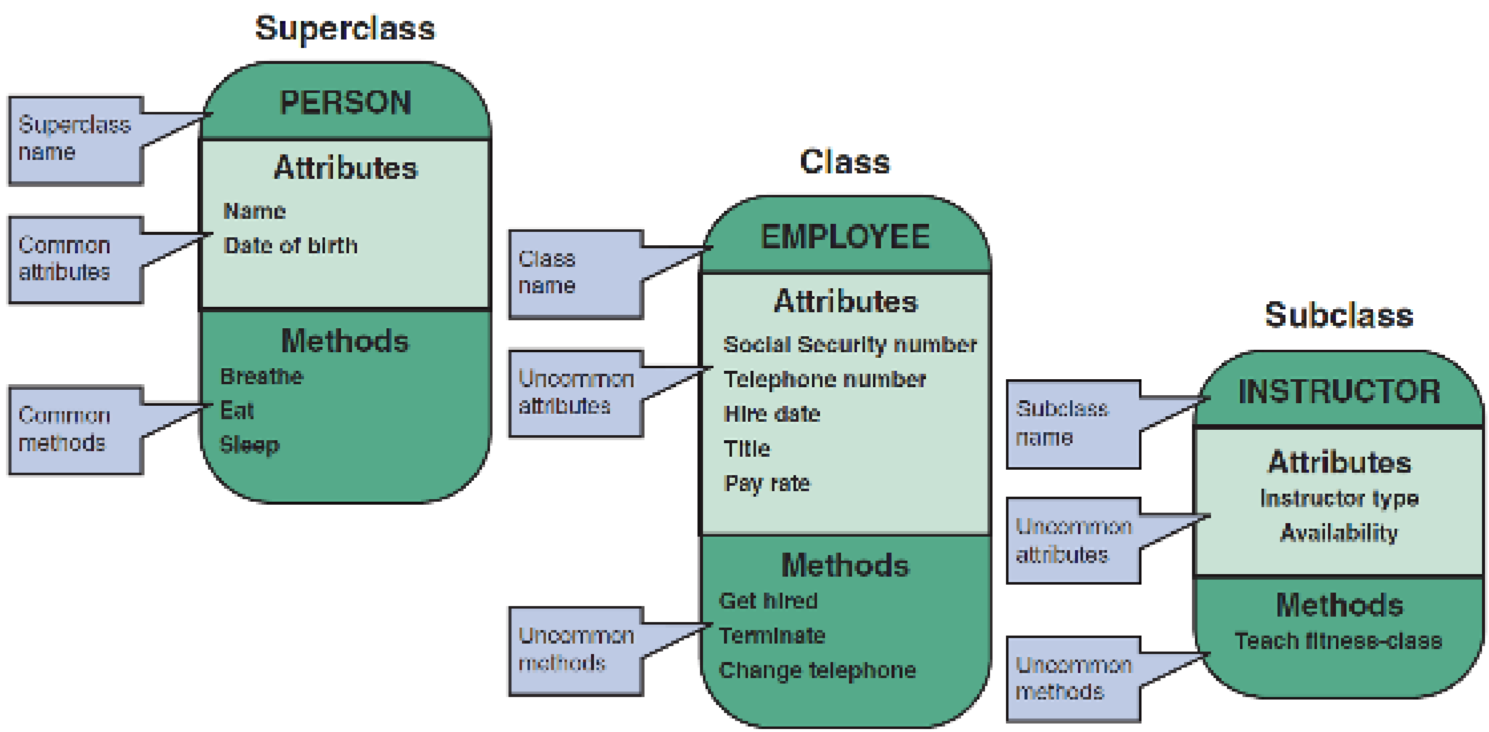


FIGURE 6-9 At the fitness center, the PERSON superclass includes common attributes and methods. EMPLOYEE is a class within the PERSON superclass. INSTRUCTOR is a subclass within the EMPLOYEE class.

Relationships Among Objects and Classes

- **Relationships**
  - Enable objects to communicate and interact as they perform business functions and transactions
  - Describe what objects need to know about each other
- **Inheritance**
  - The strongest relationship
  - Enables an object to derive one or more of its attributes from another object

Relationships Among Objects and Classes (Cont.1)

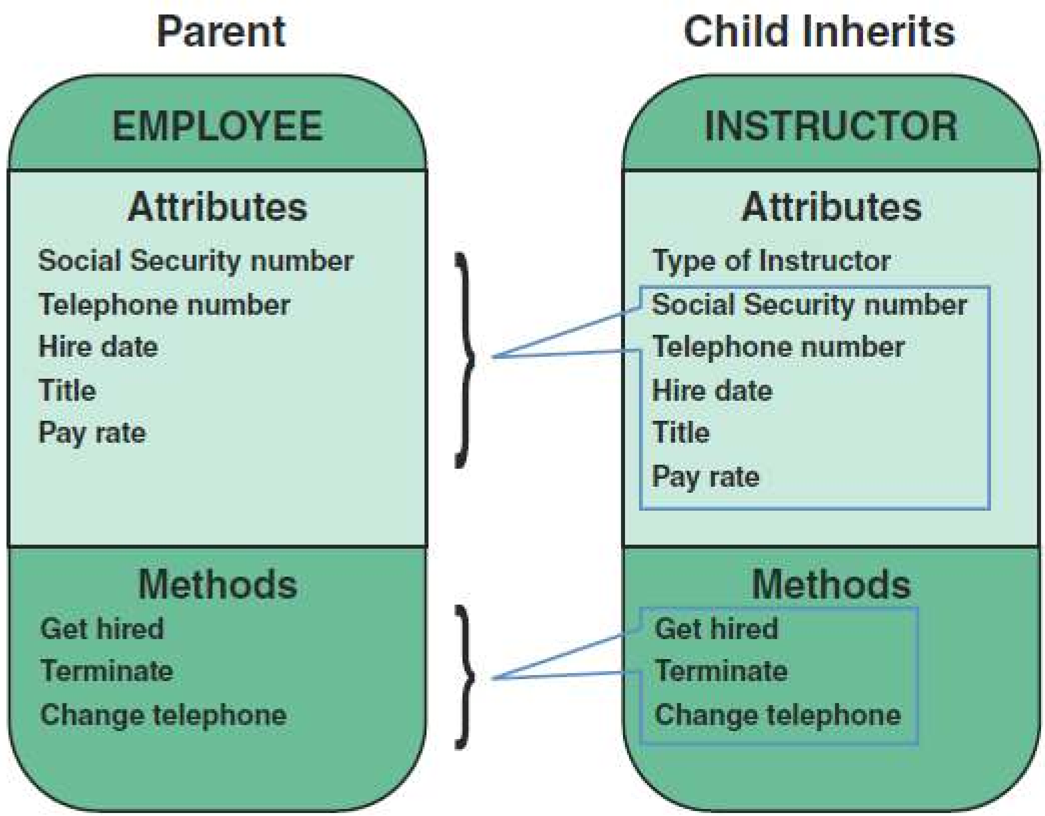


FIGURE 6-10 An inheritance relationship exists between the INSTRUCTOR and EMPLOYEE objects. The INSTRUCTOR (child) object inherits characteristics from the EMPLOYEE (parent) class and can have additional attributes of its own.

Relationships Among Objects and Classes (Cont.2)

- **Object Relationship Diagram**
  - Displays objects and how they interact to perform business functions and transactions

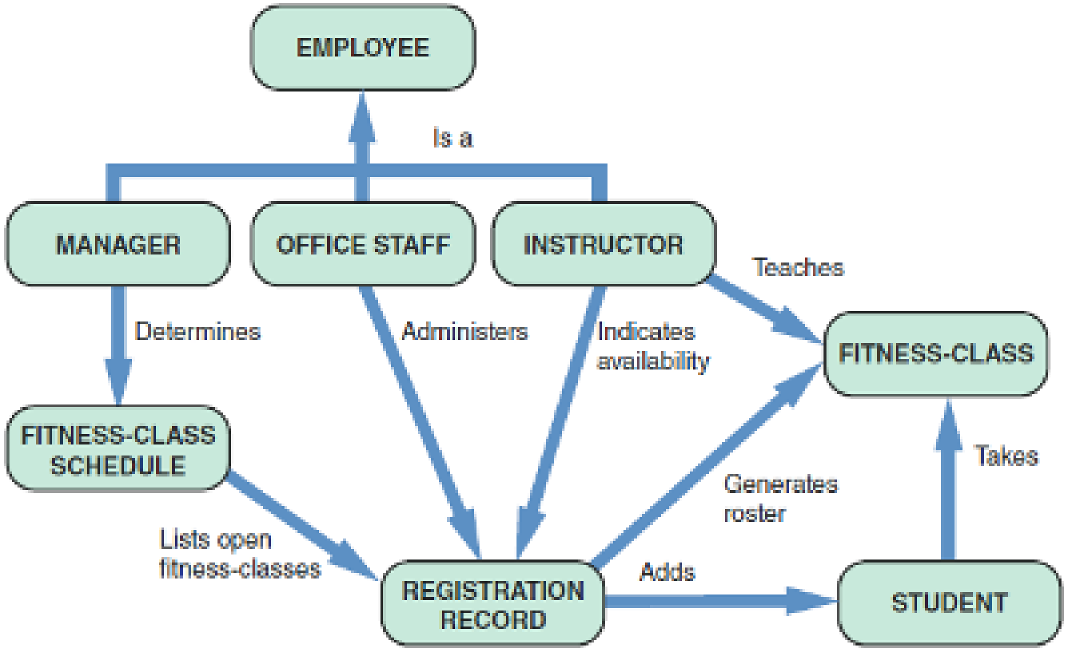


FIGURE 6-11 Object relationship diagram for the fitness center.

Object Modeling with the Unified Modeling Language

- UML uses a set of symbols to represent graphically the various components and relationships within a system
- **Use Case Modeling**
  - **Use case**: Represents the steps in a specific business function or process
  - An external entity, called an actor, initiates a use case by requesting the system to perform a function or process

Object Modeling with the Unified Modeling Language (Cont.1)

- **Use Case Modeling** (Cont.)
  - UML symbol for a use case is an oval with a label that describes the action or event
  - The actor is shown as a stick figure, with a label that identifies the actor's role
  - **Use case description**: Documents the name of the use case, the actor, a description of the use case
    - Provides a step-by-step list of the tasks and other key descriptions and assumptions

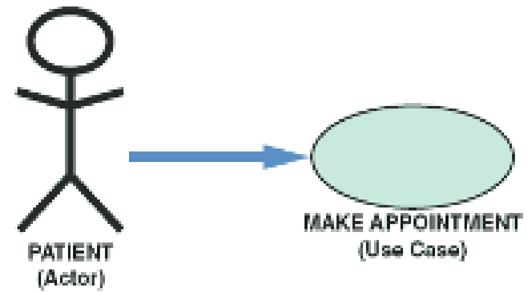


FIGURE 6-12 In a medical office system, a PATIENT (actor) can MAKE APPOINTMENT (use case).

Object Modeling with the Unified Modeling Language (Cont.2)

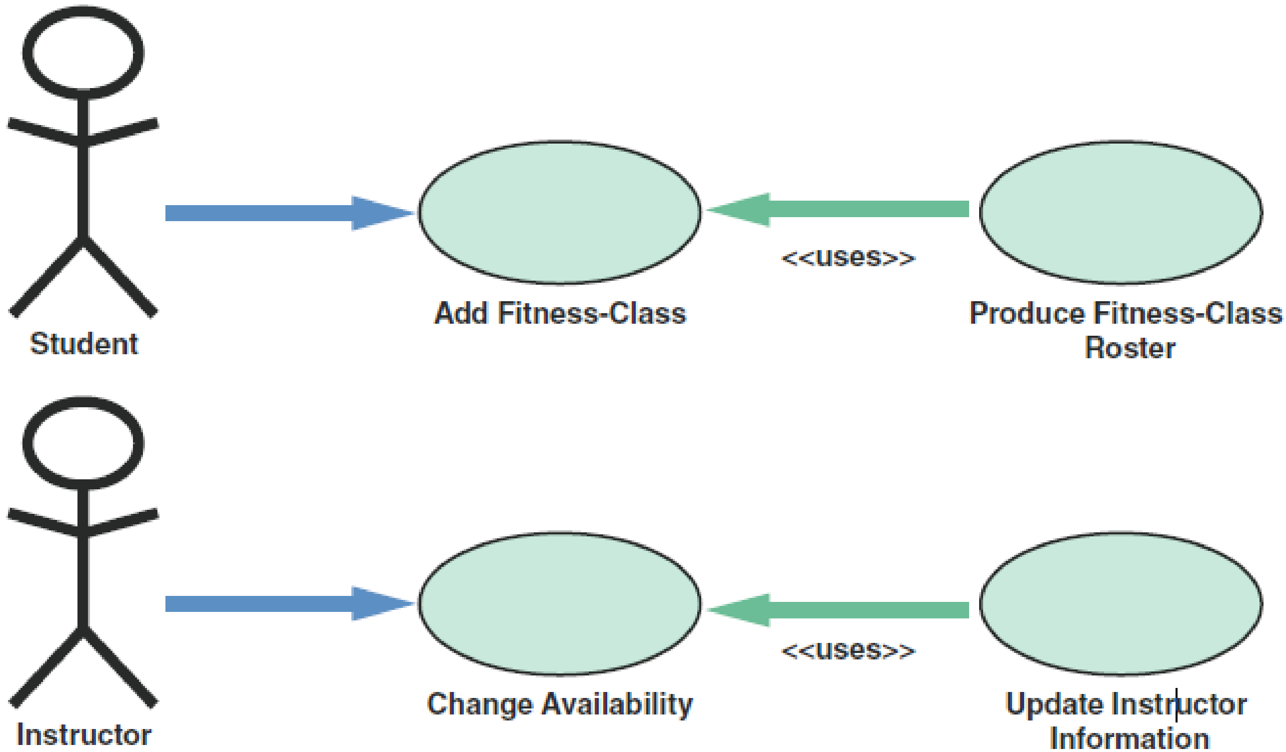
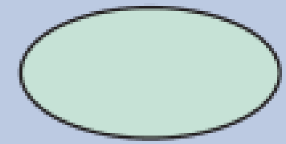


FIGURE 6-13 When a student adds a class, PRODUCE FITNESS-CLASS ROSTER uses the results of ADD CLASS to generate a new class roster. When an instructor changes his or her availability, UPDATE INSTRUCTOR INFORMATION uses the CHANGE AVAILABILITY use case to update the instructor's information.

Object Modeling with the Unified Modeling Language (Cont.3)

FIGURE 6-14 The ADD NEW STUDENT use case description documents the process used to add a current student into an existing class.



Add New Student

ADD NEW STUDENT Use Case

Name:	Add New Student
Actor:	Student/Manager
Description:	Describes the process used to add a student to a fitness-class
Successful completion:	1. Manager checks FITNESS-CLASS SCHEDULE object for availability 2. Manager notifies student 3. Fitness-class is open and student pays fee 4. Manager registers student
Alternative:	1. Manager checks FITNESS-CLASS SCHEDULE object for availability 2. Fitness-class is full 3. Manager notifies student
Precondition:	Student requests fitness-class
Postcondition:	Student is enrolled in fitness-class and fees have been paid
Assumptions:	None

Object Modeling with the Unified Modeling Language (Cont.4)

- **Use Case Diagrams**
  - A visual summary of several related use cases within a system or subsystem
  - The first step is to identify the system boundary which is represented by a rectangle
    - **System boundary**: Shows what is included in the system (inside the rectangle) and what is not included in the system (outside the rectangle)

Object Modeling with the Unified Modeling Language (Cont.5)

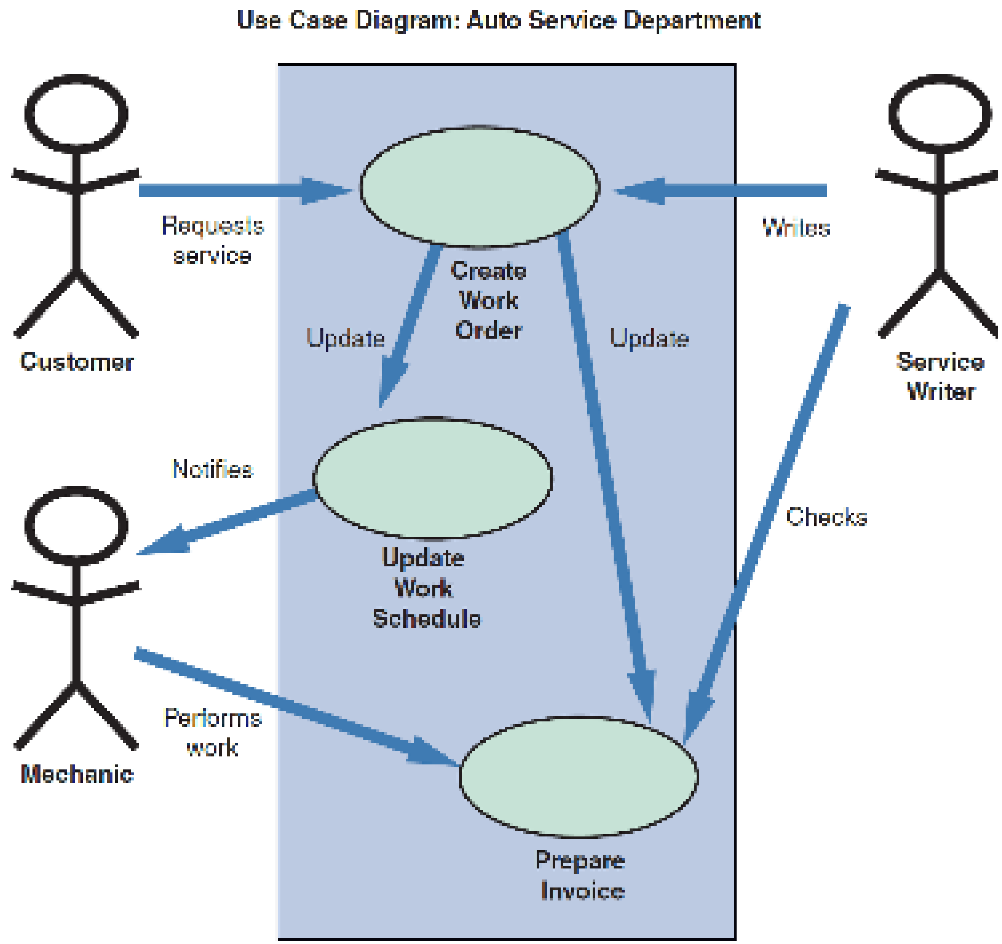


FIGURE 6-16 A use case diagram to handle work at an auto service department.

Object Modeling with the Unified Modeling Language (Cont.6)

- **Class Diagrams**
  - Show the object classes and relationships involved in a use case
  - Each class appears as a rectangle, with the class name at the top, followed by the class's attributes and methods
  - Lines show relationships between classes and have labels identifying the action that relates the two classes
  - Includes a concept called cardinality
    - **Cardinality**: Describes how instances of one class relate to instances of another class

Object Modeling with the Unified Modeling Language (Cont.7)

UML Notation	Nature of the Relationship	Example	Description
0..*	Zero or many	Employee 1 — 0..* Payroll Deduction	An employee can have no payroll deductions or many deductions.
0..1	Zero or one	Employee 1 — 0..1 Spouse	An employee can have no spouse or one spouse.
1	One and only one	Office Manager 1 — 1 Sales Office	An office manager manages one and only one office.
1..*	One or many	Order 1 — 1..* Item Ordered	One order can include one or many items ordered.

FIGURE 6-17 Examples of UML notations that indicate the nature of the relationship between instances of one class and instances of another class.

Object Modeling with the Unified Modeling Language (Cont.8)

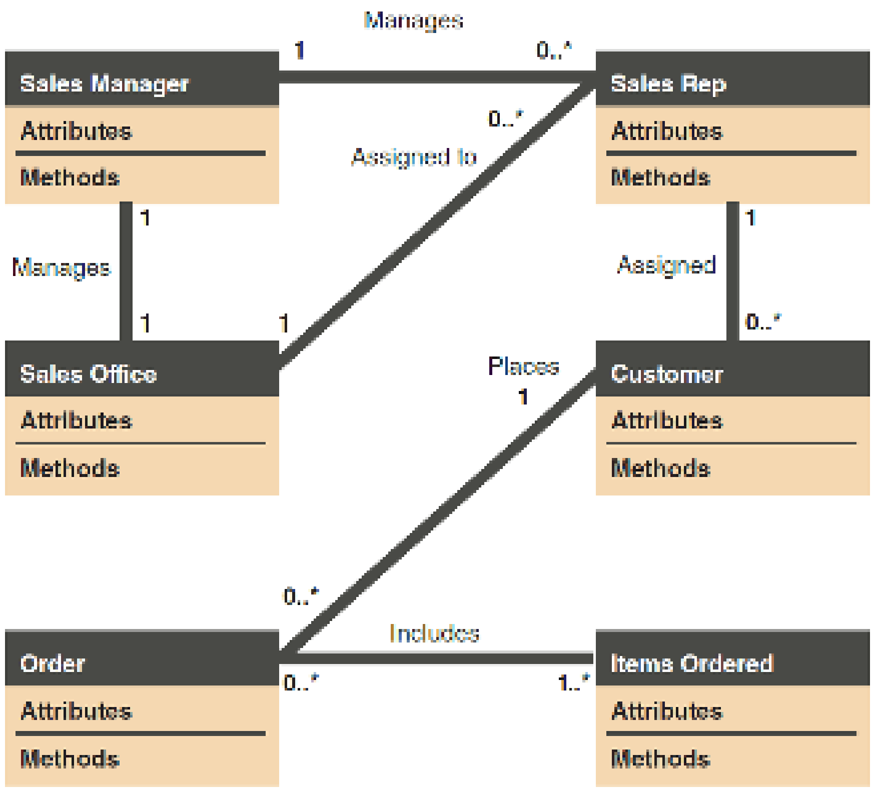


FIGURE 6-18 Class diagram for a sales order use case (attributes and methods omitted for clarity).

Object Modeling with the Unified Modeling Language (Cont.9)

- **Sequence Diagrams**
  - Dynamic model of a use case, showing the interaction among classes during a specified time period
  - Graphically document the use case by showing the classes, the messages, and the timing of the messages
  - Include symbols that represent classes, lifelines, messages, and focuses

Object Modeling with the Unified Modeling Language (Cont.10)

- **Sequence Diagrams** (Cont.)
  - Classes
    - Send or receive messages
      - Shown at the top of the sequence diagram
  - **Lifelines**
    - Represent the time during which the object above it is able to interact with the other objects in the use case
    - An X marks the end of the lifeline
  - Messages
    - Include additional information about the contents
  - Focuses
    - Indicate when an object sends or receives message

Object Modeling with the Unified Modeling Language (Cont.11)

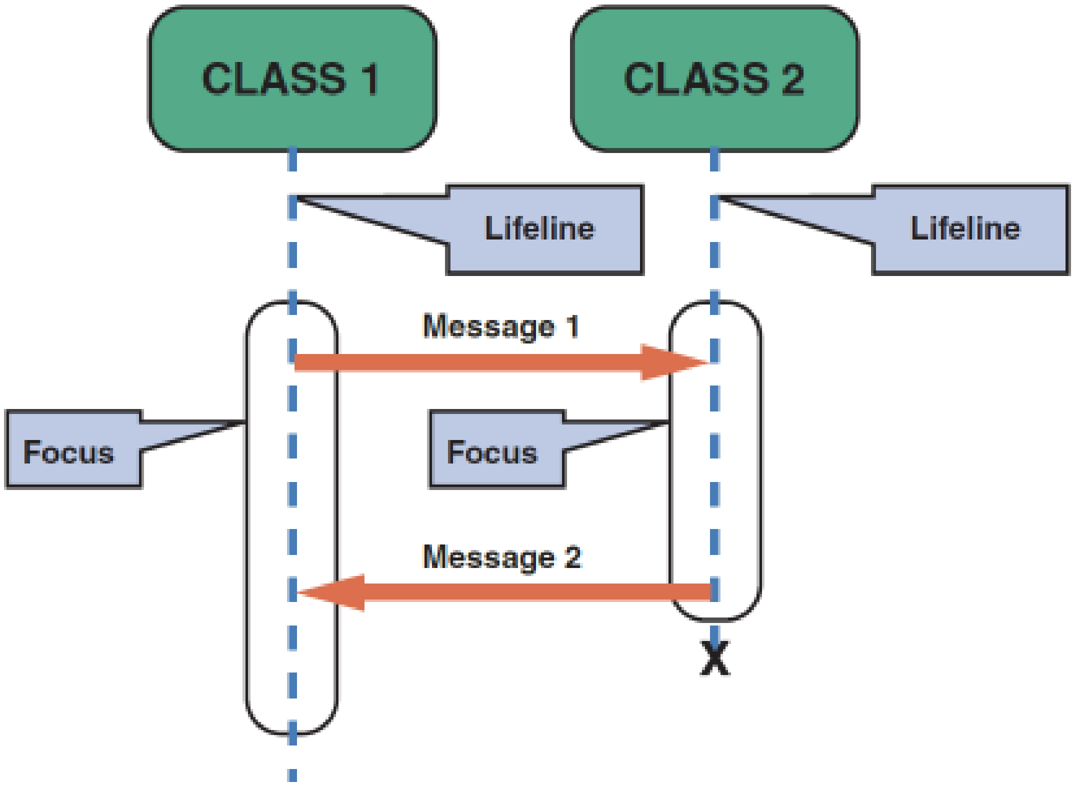


FIGURE 6-19 A sequence diagram with two classes. Notice the *X* that indicates the end of the CLASS 2 lifeline. Also notice that each message is represented by a line with a label that describes the message, and that each class has a focus that shows the period when messages are sent or received

Object Modeling with the Unified Modeling Language (Cont.12)

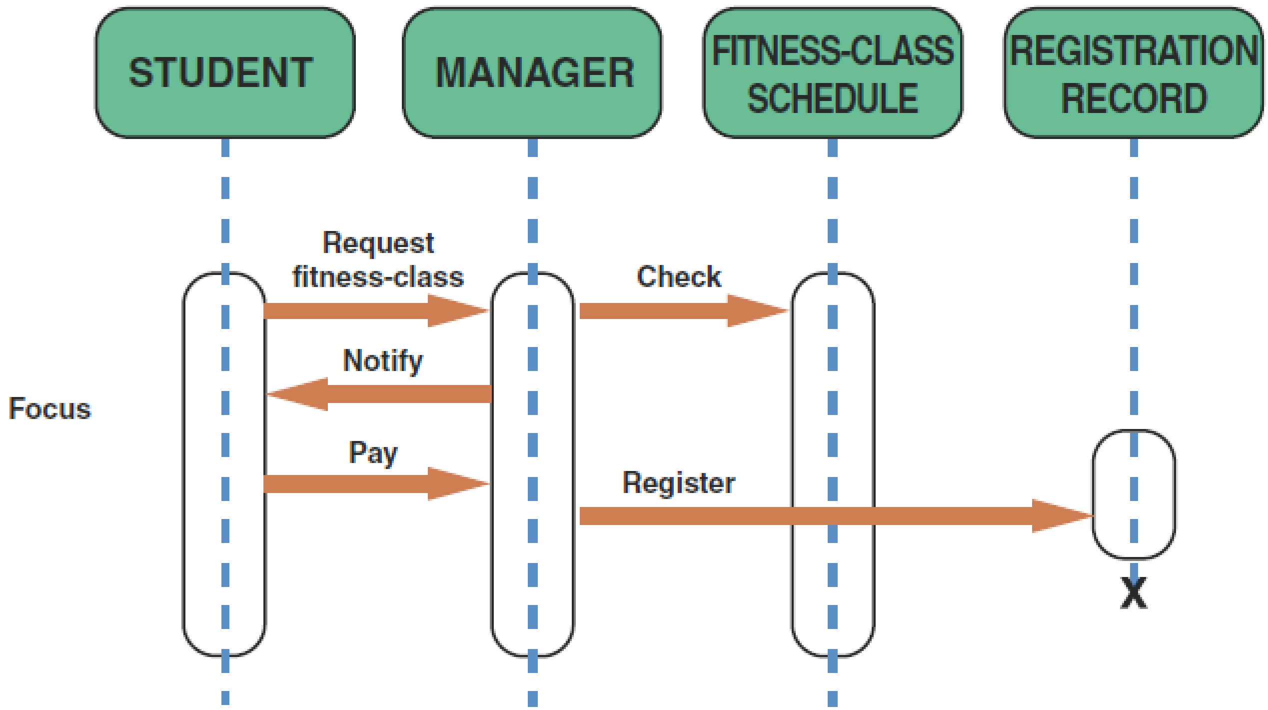


FIGURE 6-20 The sequence diagram for the ADD NEW STUDENT use case. The use case description for ADD NEW STUDENT is shown in Figure 6 14.

Object Modeling with the Unified Modeling Language (Cont.13)

- **State Transition Diagrams**
  - Show how an object changes from one state to another, depending on events that affect the object
  - All possible states must be documented in the state transition diagram
  - States appear as rounded rectangles with the state names inside

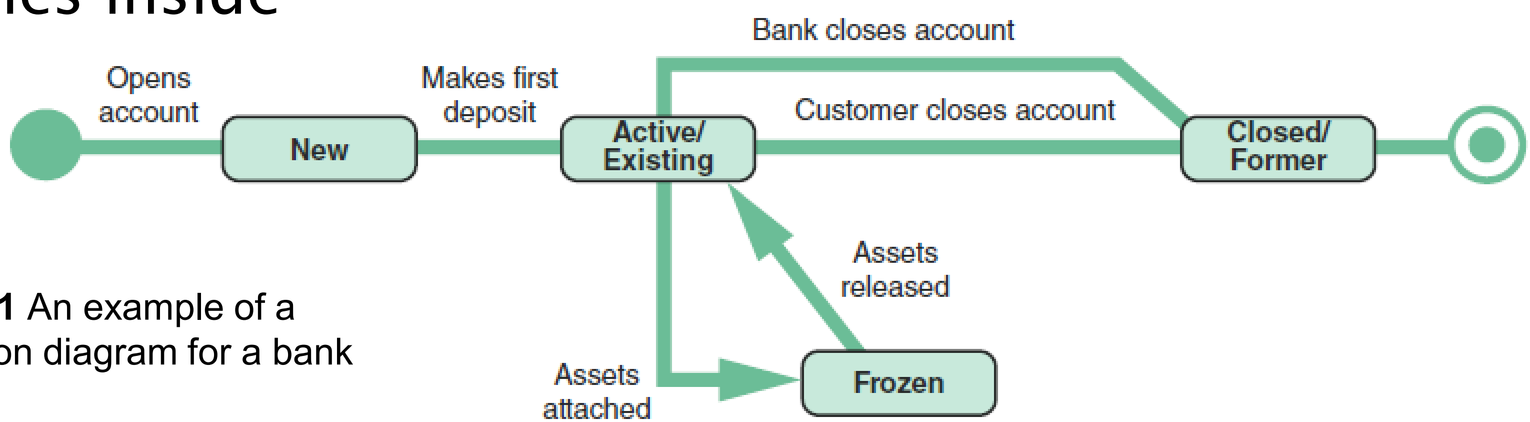


FIGURE 6-21 An example of a state transition diagram for a bank account.

Object Modeling with the Unified Modeling Language (Cont.14)

- **Activity Diagrams**
  - Show actions and events as they occur
  - Show the order in which the actions take place and identify the outcomes

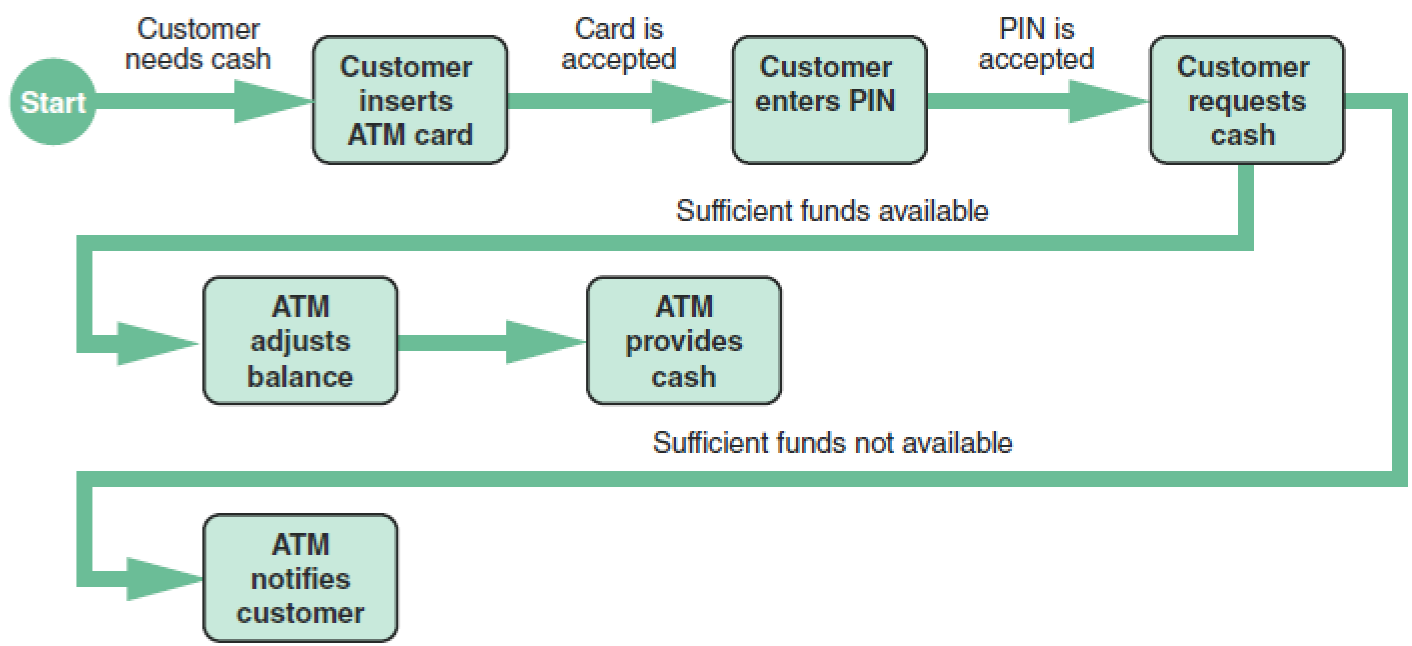


FIGURE 6-22 An activity diagram shows the actions and events involved in withdrawing cash from an ATM.

Object Modeling with the Unified Modeling Language (Cont.15)

- **Business Process Modeling (BPM)**
  - Represents the people, events, and interaction in a system
  - Can be used anytime during the systems development process
  - Compatible with object modeling

Object Modeling with the Unified Modeling Language (Cont.16)

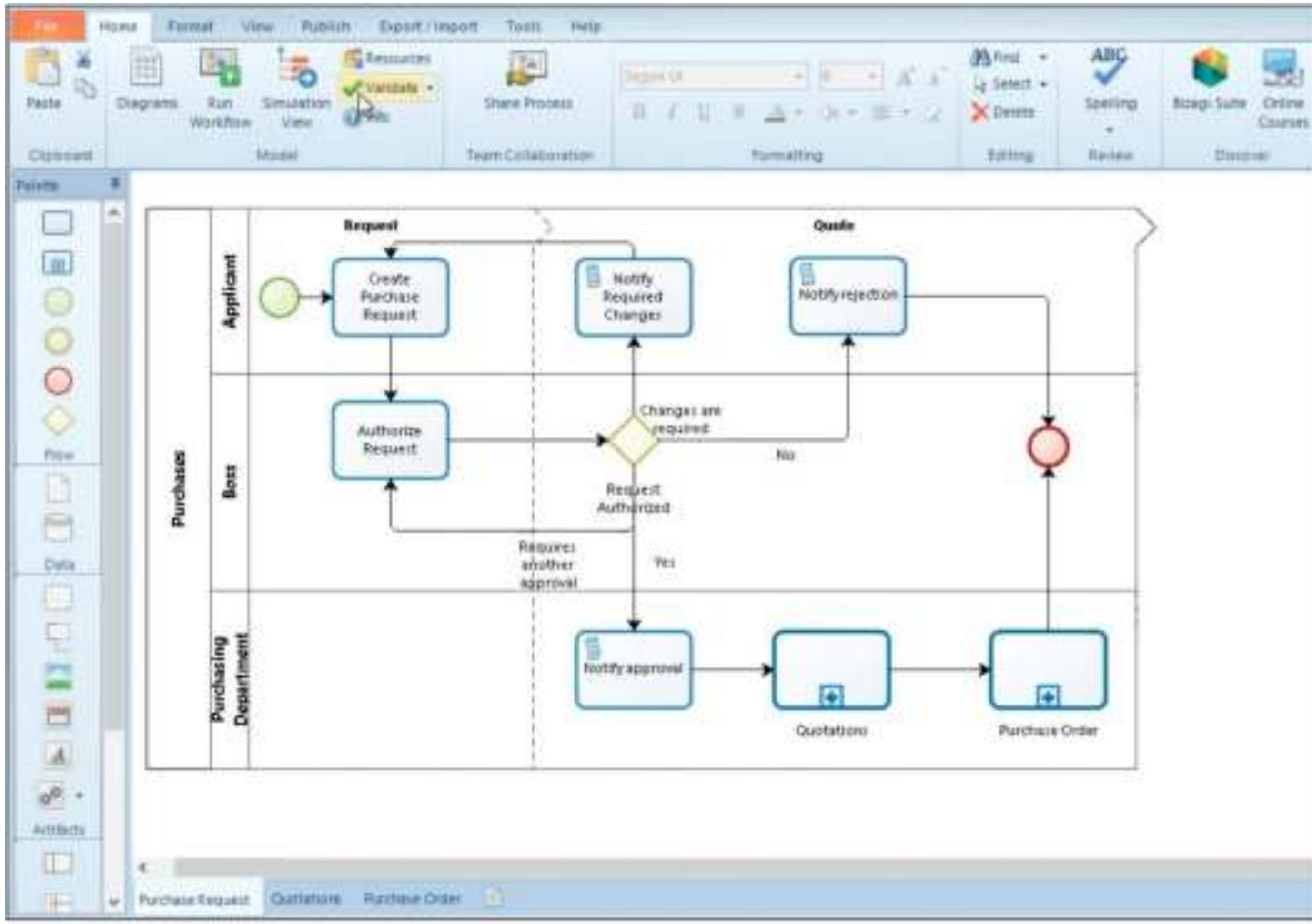


FIGURE 6-23 The Bizagi Modeler tool supports business modeling and simulation using the standard BPM notation.
Source: bizagi.com

Object Modeling with the Unified Modeling Language (Cont.)

- **CASE Tools**
  - Provide an overall framework for documenting the system components
    - Object modeling requires many types of diagrams to represent proposed systems
    - CASE tools speed up the process
  - Ensure consistency and provide common links
    - Once objects are described and used in one part of the design, they can be reused multiple times without further effort

Organizing the Object Model

- Develop an object relationship diagram that provides an overview of the system
- Support each diagram or object definition with clear and relevant documentation that can be accessed easily
    - Organize use cases and use case diagrams so they can be linked to the appropriate class, state transition, sequence, and activity diagrams
- Maintain accuracy

Chapter Summary

- Object modeling is a popular technique that describes a system in terms of objects
- Object-oriented terms include classes, attributes, instances, messages, and methods
- Objects can send messages, or commands, that require other objects to perform certain methods, or tasks

Chapter Summary (Cont.)

- The Unified Modeling Language (UML) is a widely used method of visualizing and documenting an information system
- Use case describes a business situation initiated by an actor, who interacts with the information system
- At the end of the object modeling process, the use cases and use case diagrams are organized and class, sequence, state transition, and activity diagrams are created