Chapter 5
Object-Oriented Analysis and Design

Objectives
In this chapter, you will:
• Explore OOA and OOD
• Understand the Unified Modeling Language (UML)
• Use three-tier design in OO development
• Learn about the Bradshaw Marina case study

Exploring OOA and OOD
• System analysis:
  – Study
  – Understand
  – Define
  the system requirements
• System design: show how various system components will be implemented using specific technology

Exploring OOA and OOD
• System requirements
  – Define what the system needs to accomplish for users in business terms
  – Usually described using
    • Diagrams
    • Models

Exploring OOA and OOD
• A model depicts some aspect of the required system
  – Logical models: created during system analysis
  – Physical models: created during system design
• Model-driven development: creating logical and physical models during analysis and design

Exploring OOA and OOD
• OO development models
  – Define classes of objects
  – Depict object interactions
  – Are based on UML
  – Include
    • Use case diagrams
    • Class diagrams
    • Sequence diagrams
Exploring OOA and OOD

- OO development works well with an iterative approach to development
- Iterative development
  - Analysis, design, and programming are performed in parallel, with the process repeated several times until the project is done
  - Contrasts with the waterfall method
    - An earlier approach
    - All of analysis was completed before design could start, and all of design was completed before programming could start

Exploring OOA and OOD

- OO development also uses incremental development
  - Some of the system is completed and put into operation before the entire system is finished

Understanding the Unified Modeling Language

- Object-oriented development requires a collection of models that depict system requirements and designs
- UML defines a standard set of constructs and diagrams to model OO systems
Creating and Interpreting the Use Case Diagram

- First step in system modeling: define the main system functions
- A use case diagram shows two key concepts:
  - Use case: a system function
  - Actor: person or entity using the system

Creating and Interpreting the Use Case Diagram

- One approach to identifying use cases is to identify events the system must respond to
- Three types of events can affect a system:
  - External events: occur outside the system and require the system to respond
  - Temporal events: occur at a specific point in time, such as at the end of each day or at the end of the month
  - State event: occur when the properties of an object change. Require system processing

Creating and Interpreting the Use Case Diagram

- Each use case can be documented as a series of steps users follow when they interact with the system
- Scenarios: variations in the main steps
- As the development team identifies use cases, it creates use case diagrams

Creating and Interpreting the Class Diagram

- Class diagram
  - Shows the classes involved in the system
  - Is a rectangle with three sections
    - Top section contains name of the class
    - Middle section contains attributes of the class
    - Bottom section contains methods of the class

Creating and Interpreting the Use Case Diagram

- An activity diagram
  - An additional diagram defined by UML that can be used to document use cases
  - Sometimes created for each scenario for a use case

Figure 5-1 Example of a UML use case diagram
Creating and Interpreting the Class Diagram

- **Class Diagram**
  - Association relationship between two classes is shown with a line connecting the two classes
  - Number of associations between classes is written on each end of the line
  - UML refers to the number of associations as multiplicity

Creating and Interpreting the Class Diagram

- **Class Diagram**
  - Shows generalization/specialization hierarchies (inheritance)
  - Abstract classes shown in italics
  - Objects cannot be created for an abstract class
  - Only serves to allow subclasses to inherit from it
Creating and Interpreting a Sequence Diagram

- In a sequence diagram
  - Actor can be shown as a stick figure or a rectangle
  - Objects are shown as rectangles
  - Lifelines
    - Represent a sequence of time
    - Shown as either a dashed line or a narrow box
  - Horizontal arrows represent messages sent or received in sequence
  - Data returned in response to a message is shown as a dashed line

Using Three-Tier Design in OO Development

- Three-tier design requires that OO system developers separate three categories of classes when designing and building a system
  - Three tiers
    - Graphical user interface (GUI) classes
    - Problem domain classes
    - Data access classes

Using Three-Tier Design in OO Development

- Separating GUI classes, problem domain classes, and data access classes leads to loosely coupled system components
- With loosely coupled components
  - A component can be modified with minimal effects on other components
  - Makes it easier to maintain and enhance the system
  - Components are easier to reuse

- Three-tier design
  - Provides a framework for defining OOA and OOD
    - OOA involves identifying and modeling the problem domain classes
    - In OOD, decisions about the user interface and about database management are made
  - Works well with
    - Iterative development
    - Incremental development
  - The book follows the three-tier design approach
Introducing the Bradshaw Marina Case Study

- Bradshaw Marina case study demonstrates OO development principles and practices
- When a business determines it needs a computer system, it works with a team of developers to design and develop the system

Exploring the Background of Bradshaw Marina

- Bradshaw Marina
  - A privately owned corporation that rents boat slips and provides boat services on Clinton Lake
  - Wants an automated system to track customers, slips they lease, and boats in the slips

Identifying Bradshaw Use Cases and Scenarios

- First step in OOA process: identify use cases that fall within system scope
- Since main events of interest involve customers, boats, and slips, use cases also focus on customers, boats, and slips
- Bradshaw Marina use case diagram indicates the use cases
- Several scenarios could be associated with each use case

Tasks of development team

- Analyze business and identify system functions
- Begin object-oriented analysis to identify the required use cases and scenarios, creating use case diagrams
- Identify required problem domain classes and create class diagram
- Develop sequence diagrams to model object interactions

System

- Initially: system will maintain basic information for customers, slips, and boats, and perform day-to-day business tasks
- Later: Bradshaw wants to enhance the system
  - Add boat service records
  - Add billing features

Identifying Bradshaw Use Cases and Scenarios

- System
Identifying Bradshaw Problem Domain Classes

• To explore problem domain classes, the development team would
  – Meet with Bradshaw Marina to ask about things that are involved in the work of the marina
    • For example – customers, boats, leases, slips, and docks
  – Begin an initial class diagram that includes these potential classes

Identifying Bradshaw Problem Domain Classes

• Development team would further develop the class diagram by
  – Showing generalization/specialization hierarchies
  – Adding specific information about each class
  – Identifying and modeling the association relationship among classes

Creating a Bradshaw Sequence Diagram

• Methods can be added to the class diagram by exploring scenarios and documenting them with sequence diagrams
• A sequence diagram should be created for each scenario of each use case
• As you move from OOA to OOD, you will expand the diagram to show
  – GUI objects the actor interacts with
  – Data access classes that handle interaction with database

Summary

• System analysis: study, understand, and define requirements for the system
• System requirements define what a system needs to accomplish for users in business terms
• Model-driven development: creating logical and physical models during analysis and design
• Iterative development: analysis, design, and programming are performed in parallel, with the process repeated several times until the project is done

Summary

• Incremental development: part of the system is put to use before the rest is finished
• Use case diagram shows system functions, called use cases
• Class diagram shows classes of objects that interact in the system
• Sequence diagram shows messages that the actor sends to objects and that objects send to each other
• Three-tier design divides classes into GUI classes, problem domain classes, and data access classes