CHAPTER 3

Designing OOP Solutions: Modeling the Object Interaction

The previous chapter focused on modeling the static (organizational) aspects of an OOP solution. It introduced and discussed the methodologies of the UML. It also looked at the purpose and structure of use case diagrams and class diagrams. This chapter continues the discussion of UML modeling techniques and focuses on modeling the dynamic (behavioral) aspects of an OOP solution. The focus in this chapter is on how the objects in the system must interact with each other and what activities must occur to implement the solution. This is an important aspect of the modeling processes. These models will serve as the basis for coding the various methods of the classes (identified in Chapter 2) that will make up the software application.

After reading this chapter, you should be familiar with the following:

- the purpose of scenarios and how they extend the use case models
- how sequence diagrams model the time-dependent interaction of the objects in the system
- how activity diagrams map the flow of activities during application processing
- the importance of graphical user interface design and how it fits into the object-oriented design process

Understanding Scenarios

Scenarios help determine the dynamic interactions that will take place between the objects (class instances) of the system. A scenario is a textual description of the internal processing needed to implement the functionality documented by a use case. Remember that a use case describes the functionality of the system from the viewpoint of the system’s external users. A scenario details the execution of the use case. In other words, its purpose is to describe the steps that must be carried out internally by the objects making up the system.

Figure 3-1 shows a Process Movie Rental use case for a video rental application. The following text describes the use case:

- Preconditions: The customer makes a request to rent a movie from the rental clerk. The customer has a membership in the video club and supplies the rental clerk with her membership card and personal identification number (PIN). The customer’s membership is verified. The customer information is displayed, and the customer’s account is verified to be in good standing.
- Description: The movie is confirmed to be in stock. Rental information is recorded, and the customer is informed of the due date.
- Post conditions: None.
The following scenario describes the internal processing of the Process Movie Rental use case:

- The movie is verified to be in stock.
- The number of available copies in stock is decremented.
- The due date is determined.
- The rental information is recorded. This information includes the movie title, copy number, current date, and due date.
- The customer is informed of the rental information.

This scenario describes the best possible execution of the use case. Because exceptions can occur, a single use case can spawn multiple scenarios. For example, another scenario created for the Process Movie Rental use case could describe what happens when a movie is not in stock.

After you map out the various scenarios for a use case, you can create interaction diagrams to determine which classes of objects will be involved in carrying out the functionality of the scenarios. The interaction diagram also reveals what operations will be required of these classes of objects. Interaction diagrams come in two flavors: sequence diagrams and collaboration diagrams.

**Introducing Sequence Diagrams**

A sequence diagram models how the classes of objects interact with each other over time as the system runs. The sequence diagram is a visual, two-dimensional model of the interaction taking place and is based on a scenario. Figure 3-2 shows a generic sequence diagram.

![Generic sequence diagram](image-url)