CHAPTER 7

Use-Case Modeling

MAJOR TOPICS

Objectives ................................................................. 84
Pre-Test Questions .................................................... 84
Introduction ............................................................. 85
Developing a Use-Case Model ............................... 90
Summary ................................................................. 100
Post-Test Questions .................................................. 101
OBJECTIVES

At the completion of this chapter, you will be able to:

- Identify the actors and use cases of a system.
- Describe the four types of use-case relationship.
- Develop a use-case model to capture project requirements.

PRE-TEST QUESTIONS

The answers to these questions are in Appendix A at the end of this manual.

1. What is the primary purpose of a use-case model?

....................................................................................................................................
....................................................................................................................................

2. What is a use-case?

....................................................................................................................................
....................................................................................................................................
The principal medium for capturing requirements in the Unified Process is the use-case diagram. Use-case diagrams illustrate the relationship between a software system and its users. They also describe how each type of user will interface with the system. For example, consider a library system. The librarian will check books in and out, and the library patron will search the database. These relationships are illustrated with use-case diagrams. The set of all use-case diagrams and their associated use-case descriptions constitute the use-case model. The use-case model is one of the four principal deliverables of the requirements workflow.

Actors

Users of a software system are represented as actors in a use-case diagram. Actors represent specific users, groups of users, organizational users and even external software systems. For example, a library may employ several librarians. Because all of these librarians will access the system in the same manner, a single actor (Librarian) represents them. An actor is represented by a labeled stick figure. Figure 7-1 is an example of a Librarian actor.

![Librarian](image)

Figure 7-1: Actor example

Multiple users may assume the role of a single actor. For example, all of the librarians will play the role of the Librarian actor. In addition, a single user may assume the role of multiple actors. For example, an actor may represent a Head Librarian who has the ability to purchase new books and run audits on the system. A user acts as a Head Librarian only while executing the special tasks of this position; in other instances, the same user will assume the role of the Librarian actor.
Organizational users of the library system include book publishers and other libraries. The Web site interface to the library system is an external software system that can be represented as an actor.

Use cases

Actors maintain relationships with use cases. A use case is a description of some software activity that an actor may initiate. For example, a librarian can check out an asset, such as a book. This activity is represented as a use case. Use cases are represented by an oval with a label inside or beneath that identifies the use case. Figure 7-2 is an example of a Check Out Asset use case.

![Figure 7-2: Use case example](check out asset)

Use-case relationships

Use cases can participate in four types of relationships: association, generalization, include and extend. Each relationship has its own purpose and notation. Table 7-1 lists the four types of use-case relationships, their purposes and their notations.

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Purpose</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association</td>
<td>Denotes a relationship between an actor and a use case</td>
<td></td>
</tr>
<tr>
<td>Generalization</td>
<td>Denotes inheritance between use cases</td>
<td></td>
</tr>
</tbody>
</table>
The association relationship is the interface between an actor and a use case. It is represented by a line between an actor and a use case. For example, a librarian will use the library system to check assets out. Figure 7-3 illustrates this interface using an association relationship. The box around the Check Out Asset use case represents the system boundary.

![Figure 7-3: Association relationship](image)

### Generalization

The generalization relationship is a link between use cases. Use cases often share common characteristics. Suppose you identified a second use case: Check In Asset. This second use case shares certain features with the Check Out Asset use case. Both use cases perform a transaction that affects the library’s inventory. A generalization allows you to represent this shared functionality in a third use case (the Perform Transaction use case) and inherit its functionality in both the Check In Asset and the Check Out Asset use cases.

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Purpose</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include</td>
<td>Includes the functionality of one use case in another</td>
<td>&lt;&lt;include&gt;&gt;</td>
</tr>
<tr>
<td>Extend</td>
<td>Extends the functionality of one use case to another under certain conditions</td>
<td>&lt;&lt;extend&gt;&gt;</td>
</tr>
</tbody>
</table>
The generalization relationship is represented by a solid line ending with an open arrowhead. The arrow points from the child use case to the parent use case. Figure 7-4 illustrates the generalization between the Perform Transaction use case and the Check Out Asset use case.

![Figure 7-4: Generalization](image)

**Include**

The include relationship allows one use case to include the functionality of another. Before an asset can be checked out, the system must verify that the patron does not owe any overdue fines. An include relationship allows the Perform Transaction use case to include the functionality of a Check Account Balance use case.
The include relationship is represented by a dashed line with an arrowhead. The word <<include>> is superimposed above the line to distinguish an include relationship from an extend relationship. The arrow points from the use case that includes the additional functionality to the use case being included. Figure 7-5 illustrates the include relationship between the Check Account Balance use case and the Perform Transaction use case.

![Include relationship diagram](image)

**Figure 7-5: Include relationship**

**Extend**

The extend relationship combines the functionality of one use case with the functionality of another, if certain conditions exist. For example, if a library patron owes an overdue fine, the system requires that the fine be paid before another asset can be checked out. The Pay Overdue Fine use case extends the functionality of the Perform Transaction use case only if an overdue fine is owed.

---

...
The extend relationship is represented by a dashed line with an arrowhead. The word "<<extend>>" is superimposed above the line. The arrow points from the use case that provides the additional functionality to the use case that accepts the functionality. Figure 7-6 illustrates the extend relationship between the Pay Overdue Fine use case and the Perform Transaction use case.

**DEVELOPING A USE-CASE MODEL**

The use-case model is one of the four principal deliverables of the requirements workflow. The use-case model usually takes the form of a document that consists of the following:

- Top-level use-case diagrams.
- Top-level use-case descriptions.
- Sub-level use-case diagrams.

Figure 7-7 is a use-case model document for the library system example. Note that the use-case model document contains several pages.

The first step in developing a use-case model is identifying top-level use cases. Top-level use cases are those use cases that interface directly with one or more actors. Candidate requirements that are being formalized as use cases usually translate directly into top-level use cases. Pages 2 and 3 of Figures 7-7 are top-level use-case diagrams.
At a minimum, use-case descriptions for the top-level use cases should be included. They are often included for sub-level use cases as well. Use-case descriptions provide a brief narrative of the steps in executing a use case. Any special considerations that are not adequately described in the use-case diagrams should also be included. Pages 4 and 5 of Figures 7-10 and 7-11 are the top-level use-case descriptions for the library system.

Sub-level use-case diagrams include all use cases and any relationships between use cases. Pages 6 and 9 of Figures 7-12 and 7-15 are the sub-level use-case diagrams.
Add asset to database
The system shall permit head librarians to add library assets such as books, magazines, newspapers, microfilm, and audio/visual materials to the asset database. A bar code sticker affixed to the asset by a librarian will be scanned and used to refer to the asset within the system’s asset database.

- Books and audio/visual materials are due within 31 days.
- Magazines are due within two days.
- Newspapers can only be checked out.
- Microfilm is due within three hours.

Delete asset from database
The system shall permit head librarians to remove assets from the asset database. An asset that is checked out will automatically be checked in before being removed from the asset database.

Generate circulation report
The system shall permit head librarians to generate a circulation report that provides information about the number of active patrons, the volume of checked-out assets, and the most frequently checked-out assets.

Generate accounts report
The system shall permit head librarians to generate an accounts report that provides information about the number of patrons with a negative balance, the total of overdue fine amount, the total number of overdue accounts, and a list of accounts overdue by more than 90 days.

Check out asset
The system shall permit librarians to check assets out on behalf of patrons. Patrons must supply a library card containing a bar code to be scanned by the librarian. The bar-code number shall be used to refer to the patron’s account information and determine whether an overdue fine is owed before allowing another asset to be checked out. The librarian will scan the bar code affixed to the asset being checked out, and the asset will be added to the patron’s account information and marked as checked out in the asset database.

© 2000 Software Corporation. All Rights Reserved.

Figure 7-10: Use-case model-page 4
Check in asset
The system shall permit librarians to check assets in. The librarian will scan the bar code affixed to an asset being checked in, and the asset will be removed from the patron's account information and marked as available in the asset database. If the patron owes an overdue fine, the system shall notify the librarian and permit receipt of payment.

Issue library card
The system shall permit librarians to issue library cards to patrons. The librarian shall scan a library card labeled with a unique bar-coded number. The system shall record information about the patron's name, address, home telephone number, and date of birth. A new account record shall be created containing the information entered.

Access account information
The system shall permit librarians to access patron account information. The librarian shall scan the bar-coded number from the patron's library card. The system shall provide information about the patron's name, address, home telephone number, date of birth, assets currently checked out, and overdue fines.

Generate overdue notices
The system shall permit librarians to generate a mailing of overdue notices.

Pay overdue fine
The system shall permit librarians to indicate receipt of an overdue fine. The patron's account record shall be updated to reflect payment.

Search asset database
The system shall permit patrons, the library Web site system, and other library systems to search the asset database by title, author, ISBN, date of publication, publisher, or subject.

Determine availability of asset
The system shall permit patrons, the library Web site system, and other library systems to determine the availability of an asset.
Figure 7-13: Use-case model-page 7
Figure 7-14: Use-case model-page 8
Figure 7-15: Use-case model-page 9
Exercise 7-1: Developing a use-case model

1. Create a list of actors and top-level use cases for the grocery store inventory system. Use the candidate requirements document you developed in the previous chapter exercise as a guide.

2. Using Argo/UM L, create a top-level use-case diagram for the use cases you identified in Step 1 of this exercise. Refer to Appendix A for assistance using Argo/UM L.

3. With a word processor, create a use-case description for each of the top-level use cases you identified in Step 1.

4. Using Argo/UM L, create sub-level use-case diagrams for each of the use cases you identified in Step 1.

SUMMARY

Use-case diagrams illustrate the relationships between a software system and its users. Actors are used to represent specific users, groups of users, organizational users and external software systems. Actors maintain relationships with use cases. Use cases exist within the system and represent specific ways the system can be used. Use cases can participate in four types of relationships: association, generalization, include and extend. An association is a relationship between an actor and a use case. A generalization is a relationship between use cases that denotes inheritance. An include relationship allows one use case to include the functionality of another use case. An extend relationship extends the functionality of one use case to another if certain conditions are met.
POST-TEST QUESTIONS

The answers to these questions are in Appendix A at the end of this manual.

1. What can Actors represent in use-case modeling?

2. What are the four types of use-case relationship?

3. What three things are typically included in a use-case model?