Use cases are used to explain and document the interaction that is required between the user and the system to accomplish the user’s task. Use cases are created to help the development team understand more fully the steps that are involved in accomplishing the user’s goals. Once created, use cases often can be used to derive more detailed functional requirements for the new system.

OBJECTIVES

■ Explain the purpose of use cases in the analysis phase of the SDLC.
■ Describe the various parts of a use case and the purpose of each part.
■ Explain the process used to create a use case.
■ Describe how use cases contribute to the functional requirements.
■ Describe how use cases inform the development of test plans.

CHAPTER OUTLINE

Introduction
Use Cases
   Elements of a Use Case
   Alternative Use Case Formats
   Use Cases and the Functional Requirements
Use Cases and Testing
   Building Use Cases
   Applying the Concepts at Tune Source
   Identifying the Major Use Cases
   Elaborating on the Use Cases
   Summary
Chapter 3 discussed the overall process of the analysis phase of the SDLC, resulting in the system proposal deliverable. Within the system proposal is the requirements definition, defining exactly what the new system should do. As we previously discussed, a key aspect of determining the requirements for the new system is understanding the user requirements: the things the users need to accomplish with the new system. In this chapter, we discuss use cases as a means of expressing user requirements. Since one of our goals in the systems development project is to create usable software, it is imperative to know what the users intend to do with it. Use cases help us understand and clarify the users’ required interactions with the system and can reveal most, if not all, functional requirements of the new system. Consequently, use cases are used extensively in the analysis phase when working with the users in interviews or workshop settings as a means of discovering user and functional requirements.

For many years, traditional requirements elicitation techniques involved asking the users what they wanted the system to do. The systems analysts would sit down with users and try to express what the system should do by drawing process models and data models. This was a challenge for the users for several reasons. First, the users may not know what is and is not possible for the system to do. Users are not likely to truly understand the capabilities and limitations of information systems technologies, especially new advances in technology. Second, users may have difficulty envisioning new ways to redesign business processes. Most of us find creating new ways of doing things to be a challenge because we are so accustomed to things being done the “old way.” Third, it is common for users to describe things they think they want from the new system, but our focus should be on the real needs for the new system. Finally, users often found it difficult to learn the process and data modeling languages used by the analysts.

Consequently, the concept of the use case has evolved as an important component of determining requirements for the new system. Use cases originated as a part of the object-oriented development world (see Chapter 14), but have been accepted as a useful tool regardless of the development methodology in use. This is not surprising since in any development approach (waterfall, RAD, or agile) we need to hear and understand what the user needs to accomplish with the system. Use cases are especially valuable for business system applications and Web sites. Both of these types of systems commonly involve extensive user interactions, so the use case is particularly helpful. Use cases are not as useful in other settings, such as batch processes, computationally intensive applications, or data warehousing. These settings have extensive “internal” complexity but limited user interactions. Therefore, the use case is not necessarily the best tool to use. As always, the analyst needs to be skilled in using a number of tools and must be able to select and apply the appropriate ones for the situation.

A use case represents how a system interacts with its environment by illustrating the activities that are performed by the users of the system and the system’s responses. The goal is to create a set of use cases that describe all the tasks that users need to perform with the system. Use cases are often thought of as an external or functional view of a business process, showing how the users view the process rather than the internal mechanisms by which the process operates. Since use cases describe the system’s activities from the user’s perspective in words, user involvement is essential in their development. Therefore, creating use cases helps ensure that users’ insights are explicitly incorporated into the new system.
Once the team has created a set of use cases that describe the things the users need to accomplish with the new system, there will be a number of important contributions to the analysis phase. First, the use cases will reveal considerable detail about the functional requirements of the new system. System developers commonly find that a well-constructed set of use cases specifies the majority of functional requirements. Second, use cases are very helpful in understanding exceptions, special cases, and error handling requirements in the new system. These requirements are easy to overlook, but use cases help to discover them. Finally, the text-based use case is easy for the users to understand, but it also flows easily into the creation of process models (Chapter 5) and the data model (Chapter 6), which are used by the analysts to more fully define the software that will be developed in the new system.

At one time, organizations applying traditional system development techniques used what was called business scenarios to describe user interactions with the system, while organizations applying object-oriented techniques (see Chapter 14) used what they called use cases. At present, these distinctions have largely disappeared and the term use case is widely accepted. The use case approach is the same whether the project team is focusing on understanding the as-is system or defining the to-be system, but obviously, the focus is different; the as-is model focuses on current business processes, whereas the to-be model focuses on desired business processes.

In this chapter, we first explain how to read use cases and describe their basic elements. We will depict several different styles of use cases. Then we describe the process applied to build use cases.

**USE CASES**

A use case depicts a set of activities performed to produce some output result. Each use case describes how an external user triggers an event to which the system must respond. For example, in a video store system, a customer might rent a DVD or return a DVD, or a DVD might become overdue. The acts of renting or returning DVDs and the passage of time are all events triggering a set of activities the system must perform. With this type of event-driven modeling, everything in the system can be thought of as a response to some trigger event. When there are no events, the system is at rest, patiently waiting for the next event to trigger it. When a trigger event occurs, the system (and the people using it) responds, performs the actions defined in the use case, and then returns to the waiting state.

In some situations, the process may be “small,” such as the actions that are performed when a DVD is rented in the previous example. In more complex systems (such as the Tune Source example in this book), a use case may require several distinct activities, some of which are performed each time the use case is activated and some of which are performed only occasionally (e.g., consider the return of a rented DVD, which very rarely will be returned with damage). Simple use cases may have only one path through them, while complex use cases may have several possible paths.

---

1 As you will see in Chapter 14, object-oriented techniques take the text-based use cases we describe in this chapter and create use case diagrams before moving to modeling structure and behavior (similar to the data and process models we describe in the next chapters). Use case diagrams are described in Chapter 14. We focus only on the text descriptions of the use cases in this chapter. For a more detailed description of business scenarios, see Karen McGraw and Karen Harbison, *User-Centered Requirements: The Scenario-Based Engineering Process*, Mahwah, NJ: Lawrence Erlbaum Associates, 1997. For a more detailed description of use cases, see I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard, *Object-Oriented Software Engineering: A Use Case Driven Approach*, Reading, MA: Addison-Wesley, 1992.
We create use cases when they are likely to help us better understand the situation and help convey the required user-system interactions. For very simple processes that are well explained in the requirements definition, we often do not need to create a use case. The information in the requirements definition itself is sufficient to describe what the system should do.

It is important to create use cases whenever we are reengineering processes or making any changes to business processes that will significantly alter the way people work. Remember that the use case describes what the system will do from the user’s perspective. Therefore, it is critical to involve the user in the creation of the use case so that the user understands the interactions planned for the new system. Also, the user helps to ensure that no essential steps or tasks are omitted from the use case and that rare, special circumstances are included.

Creation of use cases is often done as a part of interview sessions with users and as a part of JAD sessions. Gathering the information needed for use cases is a relatively straightforward process—as we will see, use cases are fairly simple to understand and interpret. It does take considerable practice, however, to learn to write a meaningful and complete use case. Users work closely with the project team to create the use cases. In some instances, after some practice, experienced users are able to write the use cases themselves.

Elements of a Use Case

Use cases can vary considerably from one organization to another in terms of the content included, the format followed, and the degree of formality employed. We begin with an example use case that is fairly formal and detailed. This use case is based on the scenario of a lawn care company that employs specially trained workers to apply lawn chemicals (fertilizers and pesticides) to customers’ lawns. The company maintains a chemical supply warehouse where the employees obtain the needed chemicals for their lawn care assignments. The process of obtaining lawn chemicals involves three main steps: authenticating the employee and ensuring he has the required training and credentials (a legal requirement for those who work with potentially dangerous materials such as pesticides); submitting a request for the needed chemical; and picking up the chemical from the chemical supply warehouse. The example use case focuses on the second step of this overall process: requesting a chemical. Refer to Figure 4-1 as we describe the sections of the use case. There are numerous pieces of information in the use case, each with an important role to play in describing the response to the triggering event. We will describe each section starting at the top.

Basic Information  Each use case has a name and number. The name should be as simple, yet descriptive, as possible. The number is simply a sequential number that serves to reference each use case (e.g., UC-2). The description briefly conveys the use case’s purpose.

The priority may be assigned to indicate the relative significance of the use case in the overall system. Some use cases will describe essential activities that the system must perform and hence will have a high priority level. Other use cases may describe activities that are less critical, having medium or low priority. Classifying the priority level is especially useful with a methodology that implements the system in a series of versions so that the most essential system features can be targeted first.

The actor refers a person, another software system, or a hardware device that interacts with the system to achieve a useful goal. Some organizations use the term user role rather than actor because there may be several different user groups who
FIGURE 4-1
Request a Chemical Use Case

Use Case Name: Request a chemical
ID: UC-2
Priority: High

Actor: Lawn Chemical Applicator (LCA)

Description: The Lawn Chemical Applicator (LCA) specifies the lawn chemical needed for a job by entering its name or ID number. The system satisfies the request by reserving the quantity requested or the quantity available and notifying the Chemical Supply Warehouse of the pick-up.

Trigger: A Lawn Chemical Applicator (LCA) needs a chemical for a job.

Type: External Temporal

Preconditions:
1. The LCA identity is authenticated.
2. The LCA has necessary training and credentials on file.
3. The Chemical Supply datastore is up-to-date and on-line.

Normal Course:
1.0 Request a lawn chemical from the chemical supply warehouse.
1. The LCA specifies the desired lawn chemical
2. The system verifies the chemical is approved for usage
3. The system displays the quantity of the lawn chemical on hand
4. The LCA specifies the quantity needed
5. The system asks the LCA to confirm the request for the quantity needed or the quantity available (Alternative Course 1.1)
6. The system gives the LCA a Chemical Pick-up Authorization for the quantity requested
7. The system notifies the Chemical Supply Warehouse of the chemical pick-up
8. The system stores the Lawn Chemical Request in the Chemical Request datastore

Alternative Courses:
1.1 Quantity available is less than quantity needed (branch at step 5)
1. The system asks the LCA if he wants the quantity available or to cancel the request
2a. The LCA asks to take the quantity available
3a. The system changes the quantity requested to the quantity available
4a. The system gives the LCA a Chemical Pick-up Authorization for the quantity available
5a. The system notifies the Chemical Supply Warehouse of the chemical pick-up
6a. The system stores the Lawn Chemical Request in the Chemical Management System
7a. The system notifies Purchasing of the chemical outage
2b. The LCA asks to cancel the request
3b. The system terminates the use case

Postconditions:
1. The Lawn Chemical Request is stored in the Chemical Management System.
2. The Chemical Pick-up Authorization is produced for the LCA.
3. The Chemical Supply Warehouse is notified of the chemical pick-up.
4. Purchasing is notified of chemical outage.

Exceptions:
E1: Chemical is no longer approved for use (occurs at step 2)
1. The system displays message, "That chemical is no longer approved for use"
2a. The system asks the LCA if he wants to request another chemical or to exit
3a. The LCA asks to request another chemical
4a. The system starts Normal Course again
3b. The LCA asks to exit
4b. The system terminates the use case

Summary

Inputs
Chemical name or ID
List of approved chemicals
Chemical quantity on hand
Quantity needed
Request confirmation
Request quantity available or cancellation

Outputs
Chemical Pick-up Authorization
Chemical Pick-up Notice
Lawn Chemical Request
Chemical Outage Notice

Destination
LCA
Chemical Supply Warehouse
Chemical Request datastore
Purchasing
interact with the system in the same way. For example, an order entry use case could be performed with either customers or order entry clerks performing the user role. In our example, the actor is the Lawn Chemical Applicator (LCA) who is employed by the lawn care company to apply the lawn chemicals to customers’ lawns.

Another element of basic information is the trigger for the use case—the event that causes the use case to begin. A trigger can be an external trigger, such as a customer placing an order, the fire alarm ringing, or in our example, the LCA needing a chemical for a job. Triggers can also be a temporal trigger, such as a DVD becoming overdue at the video store or time to pay the rent.

Preconditions  Use cases are often performed in a sequence in order to accomplish an overall business task. While it might be possible to describe everything in one very large use case, that use case could become unwieldy. Therefore, it is common practice to create smaller, more focused use cases breaking the whole process down into parts. When this practice is followed, it is important to define clearly what needs to be accomplished before each use case begins. These preconditions define the state the system must be in before the use case commences. In our example, you can see that in order for an LCA to request a chemical, he must be authenticated, his training and credentials must be up to date, and the datastore (a generic data repository) containing Chemical Supply information must be available and up to date. These tasks are taken care of in a different use case prior to the performance of this use case. Once these preconditions are established, the LCA can perform the Request a chemical use case.

Normal Course  The next major part of a use case is the description of the major steps that are performed to execute the response to the event, the inputs used for the steps, and the outputs produced by the steps. The normal course lists the steps that are performed when everything flows smoothly in the system. This is sometimes called the “happy path” because there are no problems or issues that arise when the steps are able to be followed normally.

As you read through the steps, you can clearly understand the interactions that occur between the user and the system. The steps are listed in the order in which they are performed and you can see the “bird’s-eye” perspective illustrated in the steps, describing what an outsider could observe while watching the user and system interact.

We also include a column in which the information that flows in or out of the steps is recorded. By recording the information for the steps, the inputs and outputs to the steps are clarified. We believe this helps to more fully explain the user–system interactions outlined in the steps.

Notice step 5 in which the step defines two possible actions with an “or” clause. This is an example of a conditional step involving a branch in the logical flow. In this case, if the quantity of chemical on hand is not sufficient to fill the request, the LCA is given the option of taking the quantity of chemical that is available. If that choice is made, an alternative course is followed, which is described in the next section of the use case. Any conditional steps are clearly noted in this fashion and alternative courses are fully described.

Alternative Courses  In this section, the steps followed for alternative paths through the use case are outlined. Alternative courses are included to depict branches in logic that also will lead to a successful conclusion of the use case. Notice that the location where the branch in logic from the normal course occurred is clearly stated. The course described in our example also depicts two potential paths through these steps. If the user
decides to accept the quantity of chemical available, steps 2a–7a will be performed; however, if the user decides to cancel the request, steps 2b–3b will be performed.

**Postconditions** As we explained in the preconditions section, use cases may be performed in a series in order to accomplish the overall user goal. In this section of the use case, we define the final products of this use case. In our example, the Lawn Chemical Request is stored, the LCA has the Chemical Pick-up Authorization, the Chemical Supply Warehouse is notified of the pick-up, and Purchasing is notified of any chemical outage. These postconditions also serve to define the preconditions for the next use case in the series. In our example, that would be the use case that describes the chemical pick-up at the Chemical Supply Warehouse.

**Exceptions** In order to be complete, a use case should describe any error conditions or exceptions that may occur as the use case steps are performed. These are not normal branches in decision logic, but are unusual occurrences or errors that could potentially be encountered and will lead to an unsuccessful result. As the use case is written and reviewed, the analyst should ask the user if there are any special situations or errors that could occur with each step. If there are, they should be explained as an exception. We want to be sure that the system does not fail while in use because of an error that no one thought about. As you probably know, in many systems, handling exceptions can require more coding effort than the normal and alternative courses. It is essential to try to identify these trouble spots during the analysis phase so we don’t encounter unexpected error conditions and crashes during testing and implementation.

In our example, at step 2 in the normal course, it is possible that a chemical requested by the LCA is no longer available for use. This happens when a chemical is deemed too harmful in some way and is legally restricted. The E1 exception outlines the steps followed to give the LCA a chance to request a different chemical (steps 3a–4a) or exit (steps 3b–4b).

**Summary Inputs and Outputs** The final section of the use case summarizes the set of major inputs and outputs to the steps of the use case. Each of the major inputs and outputs to the use case are listed, along with its source or destination. These are all possible inputs and outputs, not just those that are part of the normal course. In this area, it is easy to see the inputs supplied by the LCA and the Chemical Supply datastore as the use case is performed, and the outputs produced and where the outputs go.

**Additional Use Case Issues** Some organizations may include additional sections on their use case forms. If appropriate, it may be helpful to include sections devoted to:

- Frequency of use
- Business rules
- Special requirements
- Assumptions
- Notes and issues

These sections enable more detail to be listed about the use case as it is learned.

It is important to know exactly what state the system should be in before the use case can begin and exactly what state the system should be in when the use case is complete. That is the purpose of the precondition and postcondition sections of the use case. In our example scenario, the use case depicted in Figure 4-1 was a part of the larger user goal of obtaining a chemical from the Chemical Supply Warehouse. We chose to divide that major task into three use cases that are performed in
a series so that each use case is less complex and does not become confusingly large. When we take this approach, the preconditions and postconditions are essential, since the state at the conclusion of Use Case 1 (its postconditions) are also the preconditions for Use Case 2 (our example use case), and the postconditions for Use Case 2 are the preconditions for Use Case 3. As Figure 4-2 shows, postconditions of a use case define the required system state (preconditions) for the subsequent use case, essentially establishing the boundaries of each use case.

Another advantage to separating the overall user task into separate use cases is to take advantage of potential reusability of a use case. In our example, it is likely that there is a need to authenticate and validate credentials in several places throughout the system. We do not need to develop separate use cases each time this task is needed; we can simply reuse the one use case we have already created. In situations like this, it is a good idea to add a notation on the use case (in the Notes and issues section, for example) describing the multiple places in the system that will utilize this use case.

As you might imagine after studying Figure 4-1, it takes considerable practice to write use cases well. You should not realistically expect to create a perfect use case on the first try. The process of building use cases is one of gradual refinement: As users and analysts work through the parts of the use case, they often return to previous parts to correct them. As you gain experience, the creation of use cases will become more intuitive. Being detailed and thorough will get you a long way toward a use case that contributes a significant understanding of the system that we need to develop.

Also, keep in mind that use cases are read and used by two very different groups of people, the users/business experts and the system development experts. It is hard to find a middle ground writing style that will provide the precision needed by the development experts without overwhelming the users/business experts. Many organizations have found that use case writing teams are helpful. On the team, there should be at least one person who has a programming perspective in order to ensure adequate precision and accuracy in the use case; another person who has deep knowledge of the business rules that the system must enforce; and another person who is thoroughly familiar with how the system will actually be used.

### Alternative Use Case Formats

The use case in Figure 4-1 represents a *fully dressed* use case. This means that the use case is very thorough, detailed, and highly structured. This use case also is written as an essential use case, so that it depicts the user–system interactions as abstract, technology-independent steps. For example, in step 1 of the Normal Course, “the

---

LCA specifies the desired lawn chemical.” Nothing is said about the specific way in which this will be done. This phrasing keeps our options open in terms how this task will actually be implemented. In the analysis phase, this is the correct perspective to take, since we do not want our users to limit their thinking to just one way for the system to work too early in the process.

The fully dressed use case is not always required, but does provide value in certain circumstances. Fully dressed use cases are especially valuable when:

- User representatives are not closely engaged with the development team throughout the project.
- The application is complex and has a high risk associated with system failures.
- Comprehensive test cases will be based on the user requirements.
- Collaborating remote teams need a detailed, shared understanding of the user requirements.3

The project team may decide that a more casual use case format is acceptable. We show a casual use case for Request a chemical in Figure 4-3. As you can see,

<table>
<thead>
<tr>
<th>Use Case Name: Request a chemical</th>
<th>ID: UC-2</th>
<th>Priority: High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor: Lawn Chemical Applicator (LCA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description: The Lawn Chemical Applicator (LCA) specifies the lawn chemical needed for a job by entering its name or ID number. The system satisfies the request by reserving the quantity requested or the quantity available and notifying the Chemical Supply Warehouse of the pick-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger: A Lawn Chemical Applicator (LCA) needs a chemical for a job.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type: ✓External □Temporal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Preconditions:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. The LCA identity is authenticated.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. The LCA has necessary training and credentials on file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. The Chemical Supply datastore is up-to-date and on-line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Normal Course:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0 Request a lawn chemical from the chemical supply warehouse.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. The LCA specifies a chemical needed and the quantity needed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. The system lists chemical and quantity on hand from Chemical Supply datastore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a. If the quantity on hand is less than the quantity needed, the LCA specifies the quantity he will take</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b. Purchasing is notified of chemical shortage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. The system gives the LCA a Chemical Pick-up Authorization for the quantity requested</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. The system notifies the Chemical Supply Warehouse of the chemical pick-up</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. The system stores the Lawn Chemical Request in the Chemical Request datastore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Postconditions:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. The Lawn Chemical Request is stored in the Chemical Management System.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. The Chemical Pick-up Authorization is produced for the LCA.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. The Chemical Supply Warehouse is notified of the chemical pick-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Purchasing is notified of chemical outage.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exceptions:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E1: Chemical is no longer approved for use (occurs at step 1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. The system displays message, “That chemical is no longer approved for use”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. The system asks the LCA if he wants to request another chemical or to exit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3a. The LCA asks to request another chemical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4a. The system starts Normal Course again</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3b. The LCA asks to exit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4b. The system terminates the use case</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The system shall allow the LCA who is logged in to the Chemical Request system to request one or more chemicals.

The system shall allow the LCA to specify a chemical by entering its ID number or name.

The system shall notify the LCA if the chemical is no longer approved for use.

The system will prompt the LCA for the quantity of the chemical needed.

The system shall search the Chemical Supply datastore for the quantity available of the requested chemical and display the quantity available.

The system shall prompt the user to confirm his request.

When the request is confirmed, the system shall do the following as a single transaction:

- Assign the next Chemical Request number to the Chemical Request, assign the current date and time to the Chemical Request, record the LCA’s name and ID number on the request.
- Update the amount available of the chemical by subtracting the quantity requested from the quantity available in the Chemical Supply datastore.
- Print the Chemical Pick-up Authorization Notice for the LCA.
- Send a message to the Chemical Supply Warehouse of the approved Chemical Pick-up.
- Record the approved Chemical Request in the Chemical Request datastore, marked as “Pending Pick-up.”

The system shall prompt the LCA to exit the system or to make another chemical request.
Building Use Cases

Use cases can be used for both the as-is and the to-be systems; as-is use cases focus on the current system, whereas to-be use cases focus on the desired new system. When used for the to-be system, it is fairly common to identify additional requirements from the use cases that were not completely specified in the requirements definition. This, in fact, is one of the reasons use cases are important. After the use cases have been built, analysts often return to the requirements definition and revise it according to their improved understanding of the system.

The most common ways to gather information for the use cases are through the same requirement determination techniques discussed in the previous chapter, especially interviews and JAD sessions. Observation also is sometimes used for as-is use cases. Regardless of whether interviews or JAD sessions are used, research shows that some ways to gather the information for use cases are better than others. The most effective process has four steps.4 (See Figure 4-5.) These four steps are

<table>
<thead>
<tr>
<th>Step</th>
<th>Activities</th>
<th>Typical Questions Askeda</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Identify the use cases. Start a use case report form for each use case by filling in the name, description and trigger. If there are more than nine use cases, group them into packages.</td>
<td>Ask who, what, when, and where about the use cases (or tasks). What are the major tasks that are performed? What triggers this task? What tells you to perform this task?</td>
</tr>
<tr>
<td>2.</td>
<td>Identify the major steps within each use case. For each use case, fill in the major steps needed to complete the task.</td>
<td>Ask how about each use case. What information/forms/reports do you need to perform this task? Who gives you these information/forms/reports? What information/forms/report does this produce and where do they go? How do you produce this report? How do you change the information on the report? How do you process forms? What tools do you use to do this step (e.g., paper, e-mail, phone)?</td>
</tr>
<tr>
<td>3.</td>
<td>Identify elements within steps. For each step, identify its triggers and its inputs and outputs.</td>
<td>Ask how about each step. How does the person know when to perform this step? What forms/reports/data does this step produce? What forms/reports/data does this step need? What happens when this form/report/data is not available?</td>
</tr>
<tr>
<td>4.</td>
<td>Confirm the use case. For each use case, validate that it is correct and complete.</td>
<td>Ask the user to execute the process, using the written steps in the use case—that is, have the user role-play the use case.</td>
</tr>
</tbody>
</table>

a We have used the typical questions for the as-is model (e.g., “What are the…”). These same questions can be used for the to-be model, but they would be phrased in the future tense (e.g., “What should be the…”).

FIGURE 4-5
Steps for Writing for Use Cases

performed in order, but, of course, the analyst often cycles among them in an iterative fashion as he or she moves from use case to use case.

**Identify the Major Use Cases**  
As stated previously, use cases document one or more functional requirements outlined in the requirements definition. Therefore, identification of use cases begins with the requirements definition. The process-oriented functional requirements—things the system must do—suggest a direct action resulting from an external or temporal event. The information-oriented functional requirements—content the system must have—suggest things that happen involving information or time triggers to collect or produce information. Let’s begin an example of building use cases by revisiting the Holiday Travel Vehicles scenario. We have already seen a requirements definition for this situation (Figure 3-3). How was this information obtained? Figure 4-6 contains a transcript of an initial interview.
Use Cases

Sarah: So then, does the salesperson make out a new offer form or does he just change the original one?

Hal: He usually writes out a new one if the customer agrees to modify his offer. It’s less confusing that way.

Sarah: What happens to the original offer form?

Hal: It just gets torn up and thrown away. We don’t want it floating around and someone accidentally finds out the details of a customer’s offer.

Sarah: What if the customer doesn’t want to change his offer? Does the form just get thrown away then, too?

Hal: No, in that case the salesperson usually keeps the offer form in his own customer file. That way, he has a record of the customer’s offer and he can use that down the road when he follows up with the customer and tries to persuade him to submit another offer.

Sarah: So, let’s say the customer finally gets his offer accepted. Then what happens?

Hal: Well, things get a lot more formal now. Once the offer is accepted, the salesperson fills out a sales contract. This sales contract form contains full customer information, a complete description of the purchase vehicle, complete details of the trade-in vehicle and trade-in allowance, and a full description of any dealer-installed options. Then we list …

Sarah: (interrupting): Sorry, Hal, but that’s the first time I’ve heard you mention dealer-installed options. Tell me about them.

Hal: Oh, right, I kind of skipped that, didn’t I? Well, we sell base-model vehicles only. If customers want them fancied-up with extras and options, we can add them, for a price, of course! Any options that the customer wants should have been listed on the offer forms I mentioned earlier.

Sarah: Okay. So let’s go back to when I interrupted. The sales contract is filled out with customer information, purchased vehicle information, trade-in information, dealer-installed options … anything else?

Hal: Just the final negotiated price, taxes, and license fees, and the amount of the required customer deposit. Once we’ve received the deposit check, we settle on a delivery date that gives us the time we need to install the options, and then all parties sign the purchase contract, and we have ourselves a deal! Oh, and we make sure we list the salesperson’s name so he can get his commission on the sale later.

Sarah: What happens then?

Hal: Well, the customer typically goes off to arrange financing for the balance due on the purchase. We don’t provide financing ourselves in-house. If the customer needs help with that, we have a couple of local banks we direct him to that are interested in that kind of business.

We pull the new vehicle form out of our files and staple it to a new form we call the vehicle purchase record. The vehicle purchase record is kind of a summary of the main points of the purchase: the customer info, the vehicle info, the options added, and the final price info. These forms go into our files, ordered by customer, so we have a record of every customer’s vehicle purchase. At this point, we also write up a work order for the shop that lists all the work that needs to be done to get the vehicle ready for delivery to the customer.

Sarah: So, when it’s time for a customer to take delivery on the vehicle, what happens?

Hal: The customer comes in with the money needed to finalize the sale and the trade-in vehicle, if there is one. We go through the new vehicle with him and make sure it is satisfactory. We then collect his money, get a final signature from him, and give him a copy of the sales contract form. He gets the keys and is on his way! We then staple the last copy of the sales contract with the vehicle purchase record, and it gets filed by customer name.

Sarah: What about the trade-in vehicle?

FIGURE 4-6 (continued)
between Hal, the owner of Holiday Travel Vehicles, and Sarah, a systems analyst who is working on a project to provide an improved information system for the dealership. This interview took place early in the project when Sarah was just getting familiar with the organization, and basically focuses on the as-is system. Take a moment and read the transcript now.

As you read the interview transcript, look for things that happen that cause the dealership and its people to have to perform some tasks. These will be the major events of the system. Once you have identified an event, try to discover how the response to that event is produced. Chances are, the details will be obscure at this stage, but they will be discovered later as Sarah digs deeper into the operations of the organization. Make a list of the forms, reports, and files that are mentioned by Hal. They will become significant as the use cases are filled out. Finally, try to determine how the event concludes. How do we know it is complete? Is there a final tangible result? If so, make note of that. So, go ahead and study Figure 4-6. Make your list before continuing your reading here.

This interview gave Sarah quite a bit of information about the way the dealership operates. Following the meeting with Hal, Sarah began to organize what she learned in the interview by identifying the major events that occur in the typical operations of Holiday Travel Vehicles and the responses made to the events. The events suggest the primary things the users must accomplish with the system, and the responses describe the final results of the activities performed when the event occurs.

Before looking at Sarah’s event-response list in Figure 4-7, if you have not already done so, develop your own list based on your study of the interview transcript in Figure 4-6.

As shown in Figure 4-7, Sarah identified six major events from her initial conversation with Hal. The first two events deal with new vehicles added to inventory: identifying the need for additional inventory and placing orders and recording vehicles arriving from the manufacturers. Events 3, 4, and 5 are associated with selling vehicles. Finally, event 6 is focused on dealing with trade-in vehicles. Sarah has also listed, in the Response column, the things that signify that the response to an event is concluded. How does her list compare to yours?

As Sarah studied the event-response list, she decided that the three events associated with vehicle sales (events 3, 4, and 5) involved significant user–system interactions and deserved to be expanded upon with use cases. She decided to focus on these events first. The other events (1, 2, and 6) may be straightforward enough that she can create detailed functional requirements without the need for use cases.
If that doesn’t work, she will develop use cases for them by working with the new vehicle manager and used vehicle manager at a later time.

As she reflected on events 3, 4, and 5, Sarah could see that these events are three parts of the overall user goal of selling a vehicle to a customer. As shown in Figure 4-8, each event is an independent, but related part of the overall goal.

After the use cases are identified, the top parts of the use case form should be filled in with name, ID, primary actor, short description, and trigger—it may be too early to assign the importance level of the use case. The goal is to develop a set of major use cases with the major information about each, rather than jumping into one use case and describing it completely. This prevents the users and analysts from forgetting key use cases and helps the users explain the overall set of business processes that they are responsible for. It also helps users understand how to describe the use cases and reduces the chance of overlap between use cases. In this step, the analysts and users identify a set of major use cases that could benefit from additional definition beyond the requirements definition.

Identifying use cases is an iterative process, with users often changing their minds about what a use case is and what it includes. It is very easy to get trapped in the details at this point, so you need to remember that the goal at this step is to just identify the major use cases. For example, in the list of events shown in Figure 4-7, we have defined one event as “Customer makes an offer.” This event includes offers from customers who have trade-in vehicles as well as...
those who don’t have trade-in vehicles. We could describe these two situations as separate use cases, but this would create a larger set of smaller use cases. Therefore, these two possible variations of the event will be combined into a single use case. The trick is to select the right size so that you end up with the major use cases that need additional explanation beyond the requirements definition. Remember that a use case is a set of end-to-end activities that starts with a trigger event and continues through many possible paths until some output has been produced and the system is again at rest.

If the project team discovers more than eight or nine major use cases, this suggests that the system is complex (or that the use cases are not defined at the right level of detail). If there really are more than eight or nine major use cases, the use cases are grouped together into packages of related use cases. For example, if we were to do a more thorough study of a recreational vehicle dealership, we would likely find more than the six events discussed in our example. The events leading to uses cases could be grouped logically together in packages, such as all use cases for inventory, all use cases for sales, all use cases for the shop, etc. These packages are then treated as the major processes for the top level of the process model, with the use cases appearing on lower levels, or are treated as separate systems and modeled as separate systems. (Process modeling will be described in the next chapter.)

Since Sarah was focusing on three use cases, she prepared use case forms for each with the basic information on the top of the forms (see Figure 4-9). She then began to complete the use cases by working with a small group of salespeople from the dealership.

**Identify the Major Steps for Each Use Case**  At this point, the major use cases have been defined. In short, you have filled in the top portion of the use case (basic information). The next step is to complete the main body of the use case form. The users and analysts work together to describe the envisioned interactions between the user and the system in order to complete the response to the event.

Before beginning a discussion of the steps, the analyst should ask the users what tasks need to be completed before the use case steps can begin. This helps clarify the preconditions that are necessary for the use case. Remember that the preconditions help define the starting state of the system. Record the preconditions in the proper section on the use case form.

Next, the user–system interactions should be outlined as a series of steps in the Normal Course section of the form. The steps focus on what an independent observer would see the user and system do in response to the event. The users should concentrate on the steps that are followed when everything flows smoothly, however, make note of places where branches in logic may occur. In general, the steps should be listed in the order in which they are performed, from first to last, but there also may be steps that are performed only occasionally, have no formal sequence in which they are done, or loop back and forth. The order of steps implies a sequence, but does not require it. It is fine to list steps that have no sequence in any order you like, but if there is a sequence, you should list the steps in that way.

Each step should be about the same size as the others. For example, if we were writing steps for preparing a meal, steps such as “Take fork out of drawer” and “Put fork on table” are much smaller than “Prepare cake, using mix.” If you end up with more than nine steps or steps that vary greatly in size, you must go back and adjust the steps. Recognizing the size of the steps takes practice, but will become natural in time.

One good approach to producing the steps for a use case is to have the users visualize themselves actually performing the use case and write down the steps as if they
were writing a recipe for a cookbook. In most cases, the users will be able to quickly define what they do in as-is use cases. Defining the steps for to-be use cases may take a bit more coaching. In our experience, the descriptions of the steps change greatly as the users work through a use case. Our advice is to use a blackboard or whiteboard that easily can be erased (or paper with pencil) to develop the list of steps. Once the set of steps is fairly well defined, only then do you write it on the use case form.

Occasionally, a use case is so simple that further refinement is not needed. The analyst simply writes a brief description and does not bother to develop the steps within the use case. The information at the top of the use case form is sufficient, because the use case need not be explained in more detail. Some of the use cases presented in the exercises at the end of this chapter are simple enough that they do not need information beyond what is at the top of the use case form.

FIGURE 4-9
Major Use Cases with Basic Information

Sarah decided that the best way to understand the use case steps for this part of the system was to hold a JAD workshop that involved the sales manager and two
senior salespeople. In the workshop, the participants began by describing the initial state of the system. Sarah asked them to think about what needed to be accomplished before the use case steps could begin. Then, she asked them to describe how they envisioned working with the system to complete the task. Sarah was careful to guide them to think in terms of essential steps that did not assume a particular form of system implementation. Since the goal was to describe the user-system interactions in a new system, Sarah also helped the participants think of what could be done using technology rather than just thinking about the “old way” the steps were performed. As the team worked, it became clear that initially, Sarah had only envisioned recording new offers on vehicles. She did not think about the revision of an offer after it had been rejected. However, after discussion, the team felt that there were only minor differences in recording a new offer versus modifying a previous offer following an offer rejection. Therefore, use case 3 (Record an offer) was written to apply to either situation. After a number of iterations and revisions, the team settled on the partial use cases shown in Figure 4-10. Notice as you

<table>
<thead>
<tr>
<th>Use Case Name: Record an offer</th>
<th>ID: UC-3</th>
<th>Priority: High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor: Salesperson</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description: This use case describes how the salesperson records a customer offer on a vehicle. The offer may be a new offer or a revision of a previously rejected offer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger: Customer decides to make an offer on a vehicle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type:</td>
<td>X External</td>
<td>X Temporal</td>
</tr>
<tr>
<td>Preconditions:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Salesperson is authenticated.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Pending offers datastore is available and on-line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Vehicle inventory datastore is available and on-line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Rejected offers datastore is available and on-line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Normal Course:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Salesperson specifies the offer vehicle using the vehicle ID number.</td>
<td>Information for Steps:</td>
<td></td>
</tr>
<tr>
<td>2. The system checks for any pending offers on the vehicle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. If there is an offer pending on the vehicle, the system notifies the salesperson and the use case ends.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. If there are no pending offers on the vehicle, the system asks if this is a new offer or an offer revision.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. If this is an offer revision,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a. The salesperson specifies the ID of the previous offer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b. The system fills the offer form with the content of the previous offer from the Rejected Offers datastore.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Otherwise,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a. The system fills the offer form with details on the offer vehicle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Salesperson supplies/modifies additional information for the offer, including customer information and the specific offer details (Cash plus trade-in value, desired dealer options).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7. The system displays offer summary.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. The salesperson is asked to obtain customer permission to confirm the offer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9. If not confirmed, the offer is discarded, otherwise, the confirmed offer is stored as a Pending Offer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. A copy of the Pending Offer is printed for the customer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11. A Pending Offer Notice is sent to the sales manager for evaluation and approval.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Postconditions:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Pending Offer is stored.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Sales manager is sent notice of pending offer.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 4-10**
Major Use Cases with Steps Completed
Use Case Name: Evaluate an offer

ID: UC-4

Priority: High

Actor: Sales manager

Description: This use case describes how the sales manager evaluates an offer and accepts it or rejects it with a reason.

Trigger: A Pending offer is created and the sales manager is notified.

Type: External

Preconditions:
1. Sales manager is authenticated
2. Pending offer is available in the Pending Offers datastore

Normal Course:
1. The sales manager retrieves the Pending Offer from the Pending Offer datastore.
2. The sales manager uses the Vehicle ID number to retrieve the Vehicle Record on the vehicle.
3. The system prompts the sales manager to Accept or Reject the offer.
4. If the offer is rejected,
   a. The system prompts the sales manager to provide a reason for the rejection.
   b. An offer rejection notice including the reason is sent to the salesperson.
   c. The Pending Offer is removed from the Pending Offers datastore and stored as a Rejected Offer in the Rejected Offers datastore accessible only to the logged in salesperson.
5. If the offer is accepted,
   a. The system uses information from the Pending Offer to produce a Sales Contract.
   b. The Sales Contract is stored in the Pending Sales Contracts datastore.
   c. Two copies of the Sales Contract are printed for the salesperson and customer.
   d. The Pending Offer is removed from the Pending Offers datastore and stored in the Accepted Offers datastore.
   e. The customer deposit is recorded in the Deposits datastore.
   f. Any dealer options specified in the offer are used to prepare a Shop Work Order, which is stored in the Shop work Orders datastore and sent to the Shop Manager.

Postconditions:
1. Sales Contract is recorded in Pending Sales Contract datastore.
2. Payment is recorded.
3. Work to be done on the sale vehicle is recorded as a show Work Order and Shop Manager is notified.

Use Case Name: Take delivery

ID: UC-5

Priority: High

Actor: Salesperson

Description: This use case describes how the salesperson completes the vehicle sale to the customer.

Trigger: Customer has the final payment for the vehicle.

Type: External

Preconditions:
1. Salesperson is authenticated.
2. Sales Contract is available in Pending Sales Contract datastore.

Normal Course:
1. The salesperson retrieves the Sales Contract using the contract number.
2. The system asks the salesperson to confirm that the customer accepts the vehicle and has provided the required payment (cash plus trade-in).
3. If confirmed,
   a. The system stores the Sales contract in the Final Sales Contract datastore.
   b. A Final Sales Contract is printed for the customer.
   c. Payment is recorded.
   Otherwise, the use case ends.

Postconditions:
1. The Sales Contract is recorded in the Final Sales Contract datastore.
2. Payment is recorded.
look at the examples in Figure 4-10 that Sarah has opted for a style that is not quite as formal as the use case in Figure 4-1, but also not quite as casual as the use case in Figure 4-3. Sarah’s style is suitable for her situation and is sufficient to provide the detail that her team requires.

**Identify Elements within Steps** At this point, the steps have been described, but not the elements that further define and link the steps. In other words, the use case forms in Figure 4-10 require some final work before they are complete. The last column (“Information for Steps”) must be completed and arrows may be drawn to describe inputs and outputs from the steps. See Figure 4-11 for the completed Holiday Travel Vehicles use cases.

The goal at this point is to identify the major inputs and outputs for each step. One could identify the inputs and outputs in great detail, but this would make it difficult to list them concisely in the summary area at the bottom of the form. In our example, we have chosen to refer to the inputs and outputs broadly rather than specifying great detail. Another solution would be to identify detailed information for the steps, but to provide only general categories in the summary area of the use case form. For example, if a step needs the customer name, address, and phone number, we might note these in the step description but list only “customer information” as the major input at the top of the form.

The users and analysts now return to the steps in the use case and begin tracing the flow of the steps. Typically, this means asking what inputs (e.g., information, forms, reports) are used by each step or what outputs it produces. These are written in the last column on the use case form, with an arrow pointing into or out of a step (see Figure 4-11). Sometimes, forms, reports, and information will flow from one step to the next; these can be shown by arrows pointing from step to step.

It is not unusual at this point for users to discover that they forgot to list entire steps during their first time through the use case. These previously omitted steps are simply added to a revised use case. Our experience has shown that users can forget to include seldom used activities that occur in special cases (e.g., when data is not available or when something unexpected occurs), so it is helpful to carefully

---

**Y O U R T U R N**

Create a set of use cases for the following high-level requirements in a housing system run by the Campus Housing Service. The Campus Housing Service helps students find apartments. Owners of apartments fill in information forms about the rental units they have available (e.g., location, number of bedrooms, monthly rent), which are entered into a database. Students can search through this database via the Web to find apartments that meet their needs (e.g., a two-bedroom apartment for $800 or less per month within 1/2 mile of campus). They then contact the apartment owners directly to see the apartment and possibly rent it. Apartment owners call the service to delete their listing when they have rented their apartment(s).

In building the major use cases, follow the four-step process: Identify the use cases, identify the steps within them, identify the elements within the steps, and confirm the use cases.
**Use Case Name:** Record an offer  
**ID:** UC-3  
**Priority:** High

**Actor:** Salesperson

**Description:** This use case describes how the salesperson records a customer offer on a vehicle. The offer may be a new offer or a revision of a previously rejected offer.

**Trigger:** Customer decides to make an offer on a vehicle.

**Type:** ☑ External ☐ Temporal

**Preconditions:**
1. Salesperson is authenticated.
2. Pending offers datastore is available and on-line.
3. Vehicle inventory datastore is available and on-line.
4. Rejected offers datastore is available and on-line.

**Normal Course:**
1. Salesperson specifies the offer vehicle using the Vehicle ID number.
2. The system checks for any pending offers on the vehicle.
3. If there is an offer pending on the vehicle, the system notifies the salesperson and the use case ends.
4. If there are no pending offers on the vehicle, the system asks if this is a new offer or an offer revision.
5. If this is an offer revision,
   a. The salesperson specifies the ID of the previous offer.
   b. The system fills the offer form with the content of the previous offer from the Rejected Offers datastore.
   Otherwise,
   a. The system fills the offer form with details on the offer vehicle.
6. Salesperson supplies/modifies additional information for the offer, including customer information and the specific offer details (Cash Plus trade-in Value, desired dealer options).
7. The system displays offer summary.
8. The salesperson is asked to obtain customer permission to confirm the offer.
9. If not confirmed, the offer is discarded, otherwise, the confirmed offer is stored as a Pending Offer.
10. A Copy of the Pending Offer is printed for the customer.
11. A Pending Offer Notice is sent to the Sales Manager for evaluation and approval.

**Postconditions:**
1. Pending Offer is stored.
2. Sales Manager is sent notice of pending offer.

**Summary**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle ID</td>
<td>Salesperson</td>
<td>Offer Pending Notice</td>
<td>Salesperson</td>
</tr>
<tr>
<td>Existing Pending Offers</td>
<td>Pending Offers datastore</td>
<td>Offer Pending Notice</td>
<td>Customer</td>
</tr>
<tr>
<td>Offer Type</td>
<td>Salesperson</td>
<td>Offer Summary</td>
<td>Pending Offer</td>
</tr>
<tr>
<td>Offer ID</td>
<td>Salesperson</td>
<td>New Pending Offer</td>
<td>Pending Offer</td>
</tr>
<tr>
<td>Previous Offer details</td>
<td>Rejected Offers datastore</td>
<td>Pending Offer</td>
<td>Pending Offer Notice</td>
</tr>
<tr>
<td>Vehicle datastore</td>
<td>Vehicle details</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer details</td>
<td>Customer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Offer details</td>
<td>Salesperson</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 4-11**
Major Use Cases with Information for Steps Completed
Use Case Name: Evaluate an offer

ID: UC-4
Priority: High

Actor: Sales manager

Description: This use case describes how the sales manager evaluates an offer and accepts it or rejects it with a reason.

Trigger: A Pending offer is created and the sales manager is notified.

Type: ☑ External ☐ Temporal

Preconditions:
1. Sales manager is authenticated.
2. Pending offer is available in the Pending Offers datastore.

Normal Course: Information for Steps:
1. The Sales Manager retrieves the Pending Offer from the Pending Offer datastore.
2. The sales Manager uses the Vehicle ID number to retrieve the Vehicle Record on the vehicle.
3. The system prompts the Sales Manager to Accept or Reject the offer.
4. If the offer is rejected,
   a. The system prompts the Sales Manager to provide a reason for the rejection.
   b. An offer rejection notice including the reason is sent to the salesperson.
   c. The Pending Offer is removed from the Pending Offers datastore and stored as a Rejected Offer in the Rejected Offers datastore accessible only to the logged in salesperson.
5. If the offer is accepted,
   a. The system uses information from the Pending Offer to produce a Sales Contract.
   b. The Sales Contract is stored in the Pending Sales Contracts datastore.
   c. Two copies of the Sales Contract are printed for the Salesperson and customer.
   d. The Pending Offer is removed from the Pending Offers datastore and stored in the Accepted Offers datastore.
   e. The customer deposit is recorded in the Deposits datastore.
   f. Any dealer options specified in the offer are used to prepare a Shop Work Order, which is stored in the Shop Work Orders datastore and sent to the Shop Manager.

Postconditions:
1. Sales Contract is recorded in Pending Sales Contract datastore.
2. Pending Offer is removed from Pending Offers and added to Accepted Offers or to Rejected Offers.
3. Customer deposit amount is recorded for bookkeeper.
4. Work to be done on the sale vehicle is recorded as a Show Work Order and Shop Manager is notified.

Summary

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pending offer ID</td>
<td>Sales Manager</td>
<td>Offer Rejection Notice</td>
<td>Salesperson</td>
</tr>
<tr>
<td>Pending offer</td>
<td>Pending Offers datastore</td>
<td>New Rejected Offer</td>
<td>Rejected Offers datastore</td>
</tr>
<tr>
<td>Vehicle ID</td>
<td>Sales Manager</td>
<td>New Sales Contract</td>
<td>Sales Contract datastore</td>
</tr>
<tr>
<td>Vehicle details</td>
<td>Vehicle datastore</td>
<td></td>
<td>Sales Contract datastore</td>
</tr>
<tr>
<td>Offer decision</td>
<td>Sales Manager</td>
<td>New Accepted Offer</td>
<td>Customer/Salesperson</td>
</tr>
<tr>
<td>Reason for Rejection</td>
<td>Sales Manager</td>
<td>Purchase Deposit</td>
<td>Accepted Offers datastore</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shop Work Order</td>
<td>Deposits datastore</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shop Work Order Notice</td>
<td>Shop Work Orders</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Shop Work Orders datastore</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Salesperson</td>
</tr>
</tbody>
</table>

FIGURE 4-11 (continued)
Use Case Name: Take delivery

Actor: Salesperson

ID: UC-5

Priority: High

Description: This use case describes how the salesperson completes the vehicle sale to the customer.

Trigger: Customer has the final payment for the vehicle.

Type: ☑ External ☐ Temporal

Postconditions:
1. Salesperson is authenticated.
2. Sales Contract is available in Pending Sales Contract datastore.

Normal Course:
1. The Salesperson retrieves the Sales Contract using the contract number.
2. The System asks the salesperson to confirm that the customer accepts the vehicle and has provided the required payment (cash plus trade-in).
3. If confirmed,
   a. the system stores the Sales Contract in the Final Sales Contract datastore.
   b. A Final Sales Contract is printed for the customer.
   c. Payment is recorded.

Otherwise, the use case ends.

Postconditions:
1. The Sales Contract is recorded in the Final Sales Contract datastore.
2. Payment is recorded.

Summary

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales Contract ID</td>
<td>Salesperson</td>
<td>New Final Sales Contract</td>
<td>Final Sales Contract datastore</td>
</tr>
<tr>
<td>Vehicle accepted confirmation</td>
<td>Customer</td>
<td>Final Sales Contract</td>
<td>Customer</td>
</tr>
<tr>
<td>Payment submission verification</td>
<td>Salesperson</td>
<td>Final Payment</td>
<td>Payments datastore</td>
</tr>
</tbody>
</table>

FIGURE 4-11 (continued)

challenge the user about each step to make sure that nothing has been omitted. Remember our process of gradual refinement; it definitely applies to the creation of the use cases.

The Summary area for inputs and outputs found at the end of the use case form is completed once the team is satisfied with the steps, inflow, and outflows listed previously. In this section, all the input flows are listed in the left-most column and their source is specified in the adjacent column. In the third column, all the output flows are listed and their destination is specified in the right-most column. As we have mentioned, this summary area allows the team to easily view all the inputs that must be included to complete the use case and all the outputs that will be produced by the use case. This area of the use case form will be especially useful if the team decides to depict the system with data flow diagrams, which will be explained in Chapter 5.

Confirm the Use Case The final step is for the users to confirm that the use case is correct as written. Review the use case with the users to make sure that each step and each input and output are correct and that the final result of the use case is consistent with the final result in the event-action list. The most powerful approach is to ask the user to role-play, or execute the use case by using the written steps in the use case. The analyst will hand the user pieces of paper labeled as the major inputs.
Several years ago, a well-known national real estate company built a computer-based system to help its real estate agents sell houses more quickly. The system, which worked in many ways like an early version of realtor.com, enabled its agents to search the database of houses for sale to find houses matching the buyer’s criteria using a much easier interface than the traditional system. The system also enabled the agent to show the buyer a virtual tour of selected houses listed by the company itself. It was believed that by more quickly finding a small set of houses more closely matching the buyer’s desires, and by providing a virtual tour, the buyers (and the agent) would waste less time looking at unappealing houses. This would result in happier buyers and in agents who were able to close sales more quickly, leading to more sales for the company and higher commissions for the agent.

The system was designed with input from agents from around the country and was launched with great hoopla. The initial training of agents met with a surge of interest and satisfaction among the agents, and the project team received many congratulations.

Six months later, satisfaction with the system had dropped dramatically, absenteeism had increased by 300%, and agents were quitting in record numbers; turnover among agents had risen by 500%, and in exit interviews, many agents mentioned the system as the primary reason for leaving. The company responded by eliminating the system—with great embarrassment.

One of an agent’s key skills was the ability to find houses that match the buyer’s needs. The system destroyed the value of this skill by providing a system that could enable less skilled agents to perform almost as well as highly skilled ones. Worse still—from the viewpoint of the agent—the buyer could interact directly with the system, thus bypassing the “expertise” of the agent.

Questions:
1. How were the problems with the system missed?
2. How might these problems have been foreseen and possibly avoided?
3. In perfect hindsight, given the widespread availability of such systems on the Internet today, what should the company have done?


to the use case. The user follows the written steps like a recipe to make sure that those steps and inputs really can produce the outputs and final result defined for the use case.

Revise Functional Requirements Based on Use Cases   We have stressed in our discussion that developing use cases enables the project team to clarify and outline in detail the user–system interaction that is needed in the new system. As a result, the

Review the initial Holiday Travel Vehicle functional requirements 2-3–2-6 in Figure 3-3. Now, based on your study of UC-4 in Figure 4-11, revise the list of functional requirements to provide more clarity and detail for the task of evaluating a customer offer.
Use Cases

171

Initial Functional Requirements for Creating a Customer Offer (from Figure 3-3)

- The system will enable salespersons to create a customer offer (2.1).
- The system will allow salespeople to know whether an offer is pending on a specific vehicle (2.2).

Revised Functional Requirements for Creating a Customer Offer (based on UC-3, Figure 4-11)

- The system shall obtain the offer vehicle from the salesperson.
- The system shall search all Pending Offers to determine if the offer vehicle has a Pending Offer.
- The system shall notify the salesperson if a pending offer found for the offer vehicle, and the process terminates.
- The system shall use the salesperson’s entry of “new offer” or “revised offer” to create a new offer with vehicle details supplied from the Vehicle datastore or will fill the offer with the previous offer details obtained from the Rejected Offers datastore.
- The system shall allow the salesperson to complete and/or modify information on the offer.
- The system shall display a complete summary of the offer before it is confirmed by the customer.
- The system allows the offer to be confirmed by the customer or cancelled.
- The system shall store new confirmed offers as a new Pending Offer in the Pending Offers datastore.
- The system shall enable copies of the Pending Offer to be printed.
- The system shall send a notice of a new Pending Offer to the Sales Manager.

system to be developed is better understood. The functional requirements in the requirements definition may be modified to reflect this more detailed understanding and to provide insight to the development team on some “back-end” processing that will be needed that may not be obvious from the use cases alone.

In Figure 4-12, we revisit a portion of the Holiday Travel Vehicles functional requirements. In Figure 3-3, we had listed two requirements dealing with a customer offer (2.1 and 2.2), shown at the top of Figure 4-12. Based on UC-3 (shown in Figure 4-11), the functional requirements for recording an offer can be revised as shown in the lower portion of Figure 4-12. As you compare the two versions of functional requirements (initial and following development of the use case), the value of creating the use case should be apparent. The new version of functional requirements tells a much more detailed story about this task and will be very helpful to the members of the development team.

4-3 Functional Requirements for HTV Deliver Vehicle

Review the initial Holiday Travel Vehicle functional requirements 2-7–2-8 in Figure 3-3. Now, based on your study of UC-5 in Figure 4-11, revise the list of functional requirements to provide more clarity and detail for the task of delivering the vehicle to the customer.
APPLYING THE CONCEPTS AT TUNE SOURCE

Identifying the Major Use Cases

The first step in creating the use cases is to identify the major use cases according to the requirements definition, which was developed in the last chapter and shown in Figure 3-13. Take a minute and carefully read the requirements definition. Identify the major use cases that you think need additional definition before you continue reading.

It is important that you think about the use cases before you read what we have to say about them. So, if you haven’t tried to do this, take five minutes now and do it. We’ll wait.

The information in the functional requirements definition sometimes just flows into the use cases, but it usually requires some thought as to how to structure the use cases. After you read the requirements definition, you may be tempted to identify use cases that correspond directly to the requirement categories, such as (1) search and browse, (2) purchase, and (3) promote. However, creating an event-response list helps to clarify the number and scope of the use cases. (See Figure 4-13.)

Thinking carefully about these requirements, we can see that there are three significant triggering events: A customer arrives at the site to search and/or browse music selections; a customer selects a tune to download and buy; and the marketing department wishes to create special promotions. Let’s look at each event in turn.

When a customer arrives at the site, he or she will normally browse a predefined category of music (1.1) or enter a search for a particular title, artist, or genre of music (1.2). If the customer has visited the site and created entries on a Favorites list or has purchased any tunes in the past, the display of tunes on the site will be tailored to the customer’s interests (3.1, 3.3). The customer may select one or more music samples to which to listen (1.3, 3.1). The customer may add tunes to his Favorites list at any time (1.4). As you can see, this event encompasses requirements from both category 1 and category 3.

The second event, a customer triggering the purchase process, is kept separate from the search and browse event, although both events involve the customer. Purchasing involves gathering information about the customer (2.1), the music selection (2.2), and the method of payment (2.1, 2.3) and verifies the payment information (2.4) before the download process is triggered.

Finally, on a periodic basis, customer Favorites lists and purchase records are reviewed by the marketing department so that promotions and Web specials can be developed (3.2). Targeted promotions are created for when customers revisit the site (3.3). Specific e-mails will be directed to customers, offering additional special promotions (3.4).

<table>
<thead>
<tr>
<th>Event</th>
<th>Response</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer searches and browses Web site.</td>
<td>New entries in Favorites list and/or interests.</td>
<td>1.1, 1.2, 1.3, 1.4, 3.1, 3.3</td>
</tr>
<tr>
<td>Music is selected for purchase.</td>
<td>Purchase and download transaction is completed.</td>
<td>2.1, 2.2, 2.3, 2.4</td>
</tr>
<tr>
<td>Promotions are created.</td>
<td>Promotions are created for customers.</td>
<td>3.2, 3.4</td>
</tr>
</tbody>
</table>
Applying the Concepts at Tune Source

The project team felt satisfied that three use cases were sufficient to capture the major events associated with version 1 of the new system. These use cases were named search and browse tunes, purchase tunes, and promote tunes. The names were chosen because they describe how the system handles each of the events. Notice too that each use case name begins with a verb because the use case describes the act of doing something.

The project team then began to gather additional information to define each use case more completely. This was done on the basis of the results of the earlier analyses described in Chapter 3, as well as through the JAD sessions held with Carly, members of her marketing staff, plus some store managers and staff who are familiar with Tune Source’s existing Web-based sales system.

Elaborating on the Use Cases

During the JAD sessions, the team followed the steps of the process we outlined earlier in the chapter. For each use case, the primary actor and trigger were identified and a brief description was written. The next step was to define the major steps for each use case. The goal at this point is to describe how the use case operates. The best way to begin to understand these use cases is to visualize yourself browsing a sales-oriented Web site, searching for particular items, investigating specific items further, finally making a decision to buy, and completing the purchase. The techniques of visualizing your interaction with the process and thinking about how other systems work (informal benchmarking) are important techniques that help analysts and users understand how processes work and how to write the use cases. Both visualization and informal benchmarking are commonly used in practice. The next step is to add more detail to the steps by identifying their inputs and outputs. This means identifying what inputs are needed to complete the step (e.g., information, forms, reports) and what outputs are produced by each step. Alternative branches in logic were discussed and the team looked for error conditions that might occur. As the inputs and outputs were described, they were written in the summary area at the end of the form. Once all the use cases had been defined, the final step in the JAD session was to confirm that they were accurate. The project team had the users role-play the use cases. A few minor problems were discovered and easily fixed.

Figure 4-14 shows the completed use cases. Refer to these use cases as you read the remaining material in this chapter. Can you follow the steps? Do they seem logical? If you find something that you think may be missing, remember that use cases are created with gradual refinement, and errors and omissions can be corrected as they are discovered. Also, we have purposely tried to avoid getting lost in the details. Our goal is to include the major activities that are performed, but not necessarily every tiny detail at this point.

Search and Browse Tunes For the Search and Browse Tunes use case, the primary trigger is the Tune Shopper’s arrival on the Web site. The actor is specified as “Tune Shopper” because this person may not necessarily purchase a tune from Tune Source. The preconditions for this use case are that the Web site is up and running and the Tunes database is available. After you connect to the Web site, you may browse through the categories of selections that are featured on the page. If you are a first-time visitor, the page displays generic information. However, if you have visited the site before, any interests that were created on your previous visit will be used to customize your page and display selections that are tailored to you. In
### Use Case Analysis

#### Use Case Name: Search and browse tunes

**ID:** UC -1  
**Priority:** High

**Actor:** Tune Shopper

**Description:** This use case describes a tune shopper who searches and browses through tunes

**Trigger:** Tune shopper arrives at Web site to search and browse through tunes

**Type:** ☑ External  ☐ Temporal

**Preconditions:**
- Web site is available
- Tune database is on-line

**Normal Course:**

1.0 Search and browse tunes and select tune to purchase

1. System displays default home page or customized page
2. Tune Shopper browses links on page or enters account username and password
3. Tune Shopper wants to create an account: perform Create Account use case
4. Tune Shopper enters search request
5. System displays tune(s) matching search request
6. Tune Shopper selects a tune and wants to hear a sample
7. Tune Shopper selects a tune to add to Favorites
8. Tune Shopper selects a tune to remove from Favorites
9. Tune Shopper selects a tune to buy by placing it in shopping cart
10. Tune Shopper selects a tune to remove from shopping cart

**Information for Steps:**

- Username/password
- Search criteria
- Tune samples
- New Interest
- New Favorites
- New Shopping Cart Entry
- Modified Shopping Cart

**Alternative courses:**

1.1 Tune Shopper is a return visitor (branch at step 1)

1. System displays page customized for the return visitor using Interests from prior visits

1.2 Tune Shopper has created an account (branch at step 2)

1. System displays welcome message to account holder
2. Page is customized for the account holder using Favorites List and Targeted Promotions

**Postconditions:**

1. One or more tunes are added to shopper Interests
2. Account holder favorites list may be modified
3. Shopping cart contents may be modified

**Exceptions:**

E1: Account is not valid (occurs at step 2)

1. System displays message that username/password is not valid.
2. System asks Tune Shopper to re-enter username/password or contact customer service for help.

E2: Search request returns no results (occurs at step 3)

1. System displays message that no results were found for that search
2. System asks Tune Shopper to try another search

**Summary**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username/password</td>
<td>Tune Shopper</td>
<td>New Interest</td>
<td>Interests database</td>
</tr>
<tr>
<td>Search criteria</td>
<td>Tune Shopper</td>
<td>New Favorites</td>
<td>Favorites database</td>
</tr>
<tr>
<td>Tunes matching search</td>
<td>Tunes database</td>
<td>Modified Favorites</td>
<td>Favorites database</td>
</tr>
<tr>
<td>Tune samples</td>
<td>Tune Samples database</td>
<td>New Shopping Cart Entry</td>
<td>Shopping Cart database</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modified Shopping Cart</td>
<td>Shopping Cart database</td>
</tr>
</tbody>
</table>
### Use Case Name:
**Purchase Tune**

**ID:** UC -2  
**Priority:** High

**Actor:** Tune Buyer

**Description:** This use case describes the Tune Buyer’s purchase and download of selected tune(s)

**Trigger:** Tune Shopper has placed one or more tunes in shopping cart and is ready to check-out

**Type:** ☑ External  ☐ Temporal

**Preconditions:**
1. One or more tunes are in shopping cart
2. Tune Buyer has specified readiness to check out and buy tune(s) in cart

**Normal Course:**
1.0 Tune Buyer confirms intent to buy, supplies payment, and downloads tunes
   1. System displays shopping cart contents with prices of tunes included.  
   2. Tune Buyer specifies intention to purchase tunes in cart  
   3. System collects payment information or account information  
   4. System obtains payment authorization  
   5. Tune Buyer confirms payment transaction  
   6. System processes payment  
   7. System confirms payment acceptance  
   8. System releases tunes in cart for download  
   9. Tune Buyer selects download process for each tune; as each tune is successfully downloaded the system removes it from cart.

**Alternative courses:**
1.1 Tune buyer has established account (branch at step 3)
   1. Tune buyer specifies username and password  
   2. System retrieves account information including stored payment information  
   3. System verifies the buyer's intention to use stored payment information or modify stored payment information  

**Postconditions:**
1. Shopping cart is empty  
2. Tune purchase is recorded  
3. Tune sales transaction is recorded

**Exceptions:**
E1: Payment is not authorized (occurs at step 4)
   1. System displays message that payment is not accepted.  
   2. System asks Tune buyer to enter new payment information or exit  
   3. System terminates use case if tune buyers specifies exit; otherwise return to Normal Course step 4

E2: Tune buyer cancels payment (occurs at step 5)
   1. Tune buyer cancels payment transaction  
   2. System terminates use case

**Summary**

<table>
<thead>
<tr>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shopping cart contents</td>
<td>Tune Sale details</td>
<td>Tune Sales database</td>
</tr>
<tr>
<td>Purchase authorization</td>
<td>Account modification</td>
<td>Account database</td>
</tr>
<tr>
<td>Payment Information</td>
<td>Tune Buyer</td>
<td>Tune Buyer</td>
</tr>
<tr>
<td>Payment authorization</td>
<td>Payment Clearinghouse</td>
<td>Tune Buyer</td>
</tr>
<tr>
<td>Payment confirmation</td>
<td>Shopping cart database</td>
<td>Tune Buyer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Source</th>
<th>Outputs</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shopping cart contents</td>
<td>Shopping cart database</td>
<td>Tune Sale details</td>
<td>Tune Sales database</td>
</tr>
<tr>
<td>Purchase authorization</td>
<td>Tune Buyer</td>
<td>Account modification</td>
<td>Account database</td>
</tr>
<tr>
<td>Payment Information</td>
<td>Tune Buyer</td>
<td>Tune Buyer</td>
<td>Tune Buyer</td>
</tr>
<tr>
<td>Payment authorization</td>
<td>Payment Clearinghouse</td>
<td>Tune Buyer</td>
<td>Tune Buyer</td>
</tr>
<tr>
<td>Payment confirmation</td>
<td>Shopping cart database</td>
<td>Tune Buyer</td>
<td>Tune Buyer</td>
</tr>
</tbody>
</table>

**FIGURE 4-14 (continued)**
Use Case Name: Promote Tunes  
ID: UC -3  
Priority: High  

Actor: Marketing Department Staff  

Description: This use case describes how marketing staff periodically creates new targeted promotions  

Trigger: Time for marketing department to replace current promotions/specials with new promotions/specials  
Type: External  Temporal  

Preconditions:  
1. Marketing staff person is authenticated  
2. Promotions database is available and on-line  
3. Favorites database is available and on-line  
4. Sales database is available and on-line  

Normal Course:  
1.0 Prepare promotion or special offer based on analysis of customer activity  
1. Marketing staff specifies time period for analyses  
2. System performs customer activity analysis and sales analysis  
3. System accepts promotional details  
4. Targeted promotions are created  
5. Email messages for sales and promotions are created and sent  

Information for Steps:  
Time interval for analyses  
Favorites activity  
Sales activity  
New promotion information  
New promotions  
Email messages  

Postconditions:  
1. New Promotions are created  
2. Email messages are sent to customers  

Summary  
Inputs  
Time interval for analyses  
Favorites activity  
Sales activity  
New promotion information  
Source  
Marketing staff  
Favorites database  
Sales database  
Marketing staff  

Outputs  
New promotions  
Email messages  

Destination  
Promotions database  
Customers  

FIGURE 4-14 (continued)

addition, if you choose to open your account, you will be able to view the Favorites list you created in your account. The Tune Shopper may request searches for tunes based on title, artist, or genre. The Tune Shopper may select tunes so that they can listen to samples, automatically adding those selected tunes to the file that tracks each customer’s interests. If you like what you hear, but are not ready to buy, you may add the tune to your Favorites list so that you don’t lose track of it. You can also remove tunes you had previously added to your Favorites list. If you are ready to buy, you signal that decision, usually by placing the item in a “shopping cart.” You can continue to browse and search, adding more tunes to your shopping cart or removing tunes from the cart, or you may be ready to complete the purchase and “check out.”

One of the challenges in creating this use case is that users do not follow a particular pattern when browsing a Web site. Although the steps listed under the Normal Course are numbered, they are not necessarily performed in order. Therefore, each step is somewhat independent from the other steps.

The postconditions tell us that several things may occur as a result of this use case: there may be new entries made for shopper interests; there may be modifications to an account holder’s Favorites; and there may be items placed in the shopper’s shopping cart.

Purchase Tunes  
For the Purchase Tunes use case, the actor is specified as the Tune Buyer. This designation is made because the user of the Web site has indicated an
intention to actually purchase the item(s) in the cart. Therefore, a precondition is that there must be one or more items in the shopping cart. After the user specified he is ready to purchase, the Tune Buyer must supply payment information. The Tune Buyer may enter a username and password if he has an account; otherwise, he may establish an account or just provide purchase information for the current session only. If the Tune Buyer chooses to create an account, customer details will be gathered from the customer and a new account record will be created. Payment information will be gathered from the Tune Buyer and will be stored in the account (if there is one) or just used for the current session. Once the payment information is verified, the customer authorizes the transaction, a new purchase record is written to the sales file, and the tune is released for download by the customer.

**Promote Tunes** Finally, for the Promote Tunes use case, the marketing staff regularly performs an analysis of the files of recent customer purchases and additions to the customer Favorites list. On the basis of these analyses, Web promotions are created. In addition, e-mails are created to promote sales and specials on the regular CD sales Web site and in the stores.

**SUMMARY**

**Use Cases**

A use case contains all the information needed to build one part of a process model, expressed in an informal, simple way. A use case has a name, number, importance level, brief description, primary actor, trigger(s), preconditions, postconditions, major inputs and outputs, and a list of the major steps required to perform it. Use cases can be identified by reviewing the functional requirements. An event-response list also is useful in identifying the significant events that should be described in a use case. Once the use case is completed, often new and expanded functional requirements can be derived.

**Creating Use Cases**

When writing a use case, first identify the triggering event (external or temporal) and the primary actor. Next, develop a list of the major steps involved in using the input(s) to produce the needed output(s) and desired response(s) to the event. Now, think more deeply about each step and identify the specific input(s) and output(s) for every step. Finally, have the users role-play the use case to verify that it is correct as written.

**KEY TERMS**

<table>
<thead>
<tr>
<th>Actor</th>
<th>Input</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business scenario</td>
<td>Iteration</td>
<td>Temporal trigger</td>
</tr>
<tr>
<td>Essential use case</td>
<td>Output</td>
<td>Trigger</td>
</tr>
<tr>
<td>Event</td>
<td>Postconditions</td>
<td>Use case</td>
</tr>
<tr>
<td>Event-driven modeling</td>
<td>Preconditions</td>
<td>Use case package</td>
</tr>
<tr>
<td>External trigger</td>
<td>Primary actor</td>
<td>User role</td>
</tr>
<tr>
<td>Fully dressed use case</td>
<td>Priority</td>
<td>Visualization</td>
</tr>
<tr>
<td>Happy path</td>
<td>Role-play</td>
<td></td>
</tr>
</tbody>
</table>
QUESTIONS

1. What is the purpose of developing use cases during systems analysis?
2. How do use cases relate to the requirements stated in the requirements definition?
3. Describe the elements of the use case’s basic information section.
4. What is the purpose of the inputs and outputs section of the use case?
5. What is the purpose of stating the primary actor for the use case?
6. Why is it important to state the priority level for a use case?
7. What is the distinction between an external trigger and a temporal trigger? Give two examples of each.
8. Why do we outline the major steps performed in the use case?
9. What is the purpose of an event-response list in the process of developing use cases?
10. Should a use case be prepared for every item on the event-response list? Why or why not?
11. Describe two ways to handle a situation in which there are a large number of use cases.
12. What role does iteration play in developing use cases?
13. Describe the best way to validate the content of the use cases.

EXERCISES

A. Create a set of use cases for the process of buying glasses from the viewpoint of the patient, but do not bother to identify the steps within each use case. (Just complete the information at the top of the use case form.) The first step is to see an eye doctor who will give you a prescription. Once you have a prescription, you go to a glasses store, where you select your frames and place the order for your glasses. Once the glasses have been made, you return to the store for a fitting and pay for the glasses.

B. Create a set of use cases for the accompanying dentist office system, but do not bother to identify the steps within each use case. (Just complete the information at the top of the use case form.) When new patients are seen for the first time, they complete a patient information form that asks for their name, address, phone number, and brief medical history, which are stored in the patient information file. When a patient calls to schedule a new appointment or change an existing appointment, the receptionist checks the appointment file for an available time. Once a good time is found for the patient, the appointment is scheduled. If the patient is a new patient, an incomplete entry is made in the patient file; the full information will be collected when the patient arrives for the appointment. Because appointments are often made far in advance, the receptionist usually mails a reminder postcard to each patient two weeks before the appointment.

C. Complete the use cases for the dentist office system in exercise B by identifying the steps and the data flows within the use cases.

D. Create a set of use cases for an online university registration system. The system should enable the staff of each academic department to examine the courses offered by their department, add and remove courses, and change the information about them (e.g., the maximum number of students permitted). It should permit students to examine currently available courses, add and drop courses to and from their schedules, and examine the courses for which they are enrolled. Department staff should be able to print a variety of reports about the courses and the students enrolled in them. The system should ensure that no student takes too many courses and that students who have any unpaid fees are not permitted to register. (Assume that a fees data store is maintained by the university’s financial office, which the registration system accesses but does not change).

E. Create a set of use cases for the following system: A Real Estate, Inc. (AREI), sells houses. People who want to sell their houses sign a contract with AREI and provide information on their house. This information is kept in a database by AREI, and a subset of this information is sent to the citywide multiple listing service used by all real estate agents. AREI works with two types of potential buyers. Some
G. Create a set of use cases for the following health club membership system: When members join the health club, they pay a fee for a certain length of time. Most memberships are for one year, but memberships as short as two months are available. Throughout the year, the health club offers a variety of discounts on its regular membership prices (e.g., two memberships for the price of one for Valentine’s Day). It is common for members to pay different amounts for the same length of membership. The club wants to mail out reminder letters to members asking them to renew their memberships one month before their memberships expire. Some members have become angry when asked to renew at a much higher rate than their original membership contract, so that the club wants to track the price paid so that the manager can override the regular prices with special prices when members are asked to renew. The system must track these new prices so that renewals can be processed accurately. One of the problems in the health club industry is the high turnover rate of members. While some members remain active for many years, about half of the members do not renew their memberships. This is a major problem because the health club spends a lot in advertising to attract each new member. The manager wants the system to track each time a member comes into the club. The system will then identify the heavy users and generate a report so that the manager can ask them to renew their memberships early, perhaps offering them a reduced rate for early renewal. Likewise, the system should identify members who have not visited the club in more than a month so that the manager can call them and attempt to reinterest them in the club.

F. Create a set of use cases for the following system: A Video Store (AVS) runs a series of fairly standard video stores. Before a video can be put on the shelf, it must be catalogued and entered into the video database. Every customer must have a valid AVS customer card in order to rent a video. Customers rent videos for three days at a time. Every time a customer rents a video, the system must ensure that this customer does not have any overdue videos. If so, the overdue videos must be returned and an overdue fee paid before the customer can rent more videos. Likewise, if the customer has returned overdue videos, but has not paid the overdue fee, the fee must be paid before new videos can be rented. Every morning, the store manager prints a report that lists overdue videos; if a video is two or more days overdue, the manager calls the customer to remind him or her to return the video. If a video is returned in damaged condition, the manager removes it from the video database and may sometimes charge the customer.

H. Create a set of use cases for the following system: Picnics R Us (PRU) is a small catering firm with five employees. During a typical summer weekend, PRU caters 15 picnics with 20 to 50 people each. The business has grown rapidly over the past year, and the owner wants to install a new computer system for managing the ordering and buying process. PRU has a set of 10 standard menus. When potential customers call, the receptionist describes the menus to them. If the customer decides to book a picnic, the receptionist records the customer information (e.g., name, address, phone number, etc.) and the information about the picnic (e.g., place, date, time, which one of the standard menus, total price) on a contract. The customer is then faxed a copy of the contract and must sign and return it along with a deposit (often by credit card or check) before the picnic is officially booked. The remaining money is collected when the picnic is delivered. Sometimes, the customer wants something special (e.g., birthday cake). In this case, the receptionist takes the information and gives it to the owner who determines the cost; the receptionist then calls the customer back with the price information. Sometimes the customer accepts the price; other times, the customer requests some changes, which have to go back to the owner for a new cost estimate. Each week, the owner looks through the picnics scheduled for that weekend and orders the supplies (e.g., plates and food (e.g., bread, chicken) needed to make them. The owner would like to use the system for marketing as well.
Chapter 4  Use Case Analysis

It should be able to track how customers learned about PRU and identify repeat customers so that PRU can mail special offers to them. The owner also wants to track the picnics on which PRU sent a contract, but the customer never signed the contract or actually booked a picnic.

I. Create a set of use cases for the following system: Of-the-Month Club (OTMC) is an innovative young firm that sells memberships to people who have an interest in certain products. People pay membership fees for one year and each month receive a product by mail. For example, OTMC has a coffee-of-the-month club that sends members one pound of special coffee each month. OTMC currently has six memberships (coffee, wine, beer, cigars, flowers, and computer games), each of which costs a different amount. Customers usually belong to just one, but some belong to two or more. When people join OTMC, the telephone operator records the name, mailing address, phone number, e-mail address, credit card information, start date, and membership service(s) (e.g., coffee). Some customers request a double or triple membership (e.g., two pounds of coffee, three cases of beer). The computer game membership operates a bit differently from the others. In this case, the member must also select the type of game (action, arcade, fantasy/science fiction, educational, etc.) and age level. OTMC is planning to greatly expand the number of memberships it offers (e.g., video games, movies, toys, cheese, fruit, vegetables), so the system needs to accommodate this future expansion. OTMC is also planning to offer three-month and six-month memberships.

J. Create a set of use cases for a university library borrowing system. (Do not worry about catalogue searching, etc.) The system will record the books owned by the library and will record who has borrowed what books. Before someone can borrow a book, he or she must show a valid ID card that is checked to ensure that it is still valid against the student database maintained by the registrar’s office (for student borrowers), the faculty/staff database maintained by the personnel office (for faculty/staff borrowers), or against the library’s own guest database (for individuals issued a “guest” card by the library). The system must also check to ensure that the borrower does not have any overdue books or unpaid fines before he or she can borrow another book. Every Monday, the library prints and mails postcards to those people with overdue books. If a book is overdue by more than two weeks, a fine will be imposed and a librarian will telephone the borrower to remind him or her to return the book(s). Sometimes books are lost or are returned in damaged condition. The manager must then remove them from the database and will sometimes impose a fine on the borrower.

MINICASES

I. Williams Specialty Company is a small printing and engraving organization. When Pat Williams, the owner, brought computers into the business office eight years ago, the business was very small and very simple. Pat was able to utilize an inexpensive PC-based accounting system to handle the basic information processing needs of the firm. As time has gone on, however, the business has grown and the work being performed has become significantly more complex. The simple accounting software still in use is no longer adequate to keep track of many of the company’s sophisticated deals and arrangements with its customers.

Pat has a staff of four people in the business office who are familiar with the intricacies of the company’s record-keeping requirements. Pat recently met with her staff to discuss her plan to hire an IS consulting firm to evaluate their information system needs and recommend a strategy for upgrading their computer system. The staff are excited about the prospect of a new system, since the current system causes them much aggravation. No one on the staff has ever done anything like this before, however, and they are a little wary of the consultants who will be conducting the project.

Assume that you are a systems analyst on the consulting team assigned to the Williams Specialty Co. engagement. At your first meeting with the Williams staff, you want to be sure that they understand the work that your team will be performing and how they will participate in that work.

a. Explain, in clear, nontechnical terms, the goals of the analysis phase of the project.

b. Explain, in clear, nontechnical terms, how use cases will be used by the project team. Explain what these models are, what they represent in the system, and how they will be used by the team.