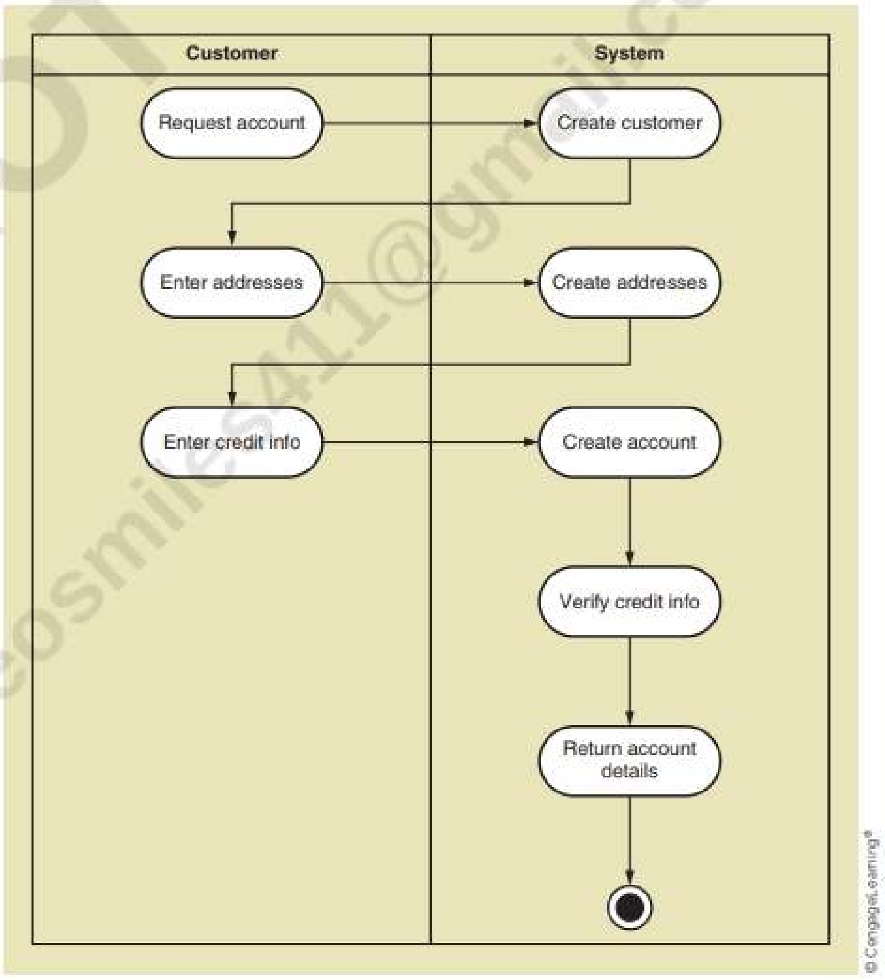
■ Activity Diagrams for Use Cases

Another way to document a use case is with a UML activity diagram. In Chapter 2, you learned about activity diagrams as a form of workflow diagram that might cover several use cases. Activity diagrams are also used to document the flow of activities for one use case.

Figure 5-4 is an activity diagram that documents the flow of activities for the *Create customer account* use case. Sometimes, an activity diagram can take the place of the flow of activities section of a use case description, and sometimes, it is created to supplement the use case description. In this example, there are two swimlanes: one for the customer and one for the system. The customer has three activities, and the system has five activities.

An example is shown in **Figure 5-5** for the *Ship items* use case previously seen in Figure 5-3. One of the strengths of activity diagrams is that it provides a more graphical view of the flow of activities. Figure 5-5 illustrates both a repeating set of steps, that is for each SaleItem in the Sale, and a decision point to choose which set of steps to perform. Even though this flow is described by the use case description, it is more evident in the activity diagram. Figure 5-5 illustrates a correct use of the beginning and ending synchronization bars and the decision diamond. Remember that synchronization

FIGURE 5-4 *Activity diagram for* Create customer account *showing alternate way to model the flow of activities*



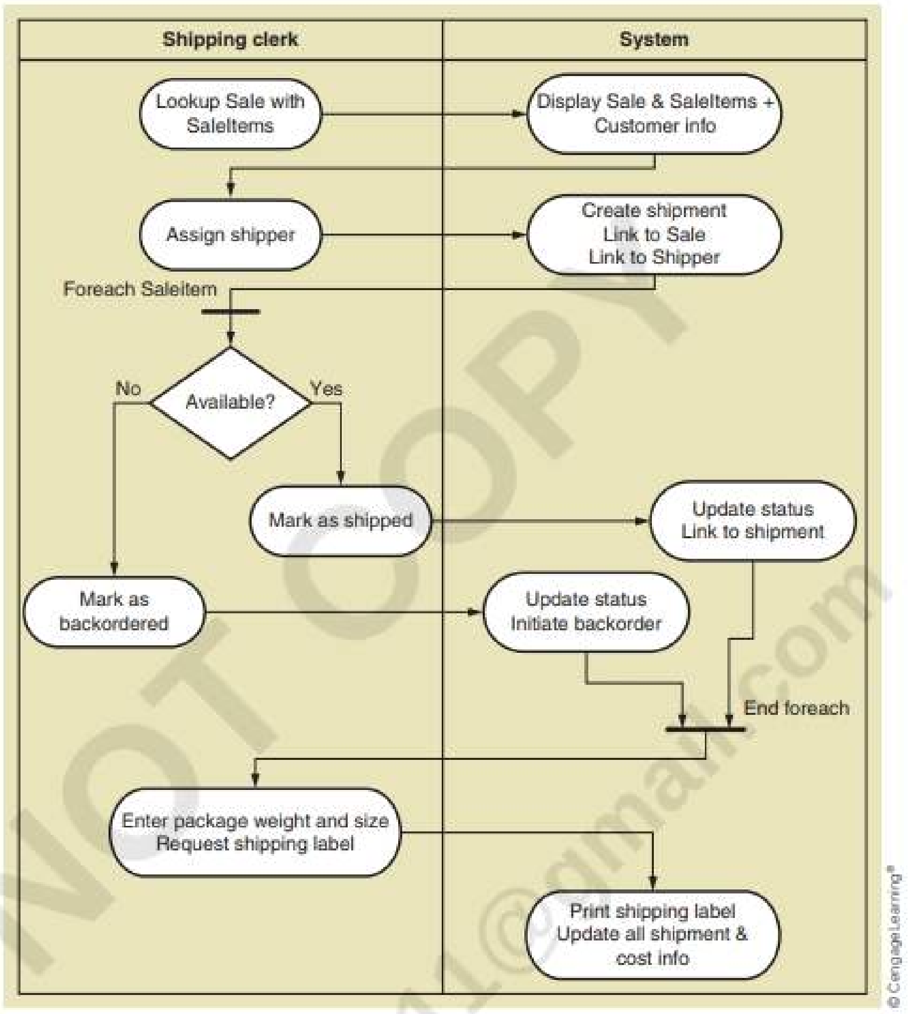
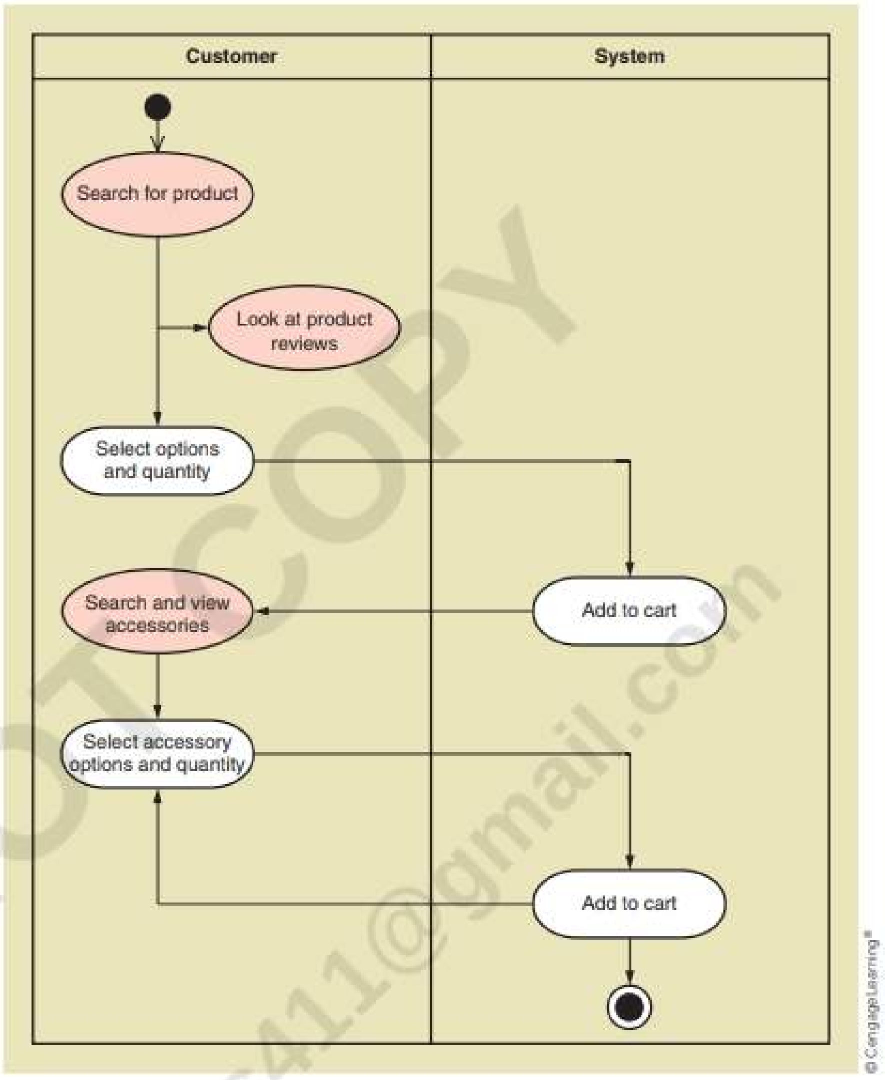

Shipping clerk
System
Lookup Sale with SaleItems
Display Sale & SaleItems + Customer info
Assign shipper
Create shipment Link to Sale Link to Shipper
Foreach SaleItem
Available?
No
Yes
Mark as shipped
Update status Link to shipment
Mark as backordered
Update status Initiate backorder
End foreach
Enter package weight and size Request shipping label
Print shipping label Update all shipment & cost info
© Cengage Learning®

FIGURE 5-5 *Activity diagram for Ship Items use case*

bars can be used either for parallel concurrent paths, or for beginning and ending loops. In this example, we see a loop and inside the loop a decision point to initiate independent paths, which are later rejoined at the ending synchronization bar.

Activity diagrams are helpful when the flow of activities for a use case is complex. The use case *Fill shopping cart* is complex in that three other use cases might be invoked while adding items to the shopping cart. For example, the actor might search for a product and then look at product reviews before adding the item to the cart. Once an item is added, the actor might search for and view available accessories and then add one or more to the cart. The activity diagram shown in **Figure 5-6** shows the *Fill shopping cart* use case flow of activities. The shaded ovals show the other use cases that are invoked while filling the shopping cart. The activities of *Fill shopping cart* go in between the other use cases. For example, after invoking *Search for product* and then *Look at product reviews*, the actor might start *Fill shopping cart* to select options and quantities and add it to the cart. Then the actor might switch to *Search and view accessories* before continuing *Fill shopping cart* to add an accessory. The activity diagram can be used to show a richer user experience in this way.

FIGURE 5-6 *Activity diagram for* Fill shopping cart *showing richer user experience*



■ The System Sequence Diagram—Identifying Inputs and Outputs

system sequence diagram (SSD)
a diagram showing the sequence of messages between an actor and the automated part of the system during a use case or scenario

In the object-oriented approach, the flow of information is achieved through sending messages either to and from actors or back and forth between internal objects. A **system sequence diagram (SSD)** is used to describe this flow of information into and out of the automated portion of the system. Thus, an SSD documents the inputs and the outputs and identifies the interaction between actors and the system. It is an effective tool to help in the initial design of the user interface by identifying the specific information that flows from the user into the system and the information that flows out of the system back to the user. An SSD is a special type of UML sequence diagram. You will learn more about detailed sequence diagrams in Chapter 13.

■ SSD Notation

Figure 5-7 shows a generic SSD with callouts annotating the diagram. As with a use case diagram, the stick figure represents an actor—a person (or role) who interacts with the system. In a use case diagram, the actor "uses" the system, but the emphasis in an SSD is on how the actor "interacts" with the system by entering input data and receiving output data. The box labeled <u>:System</u> is an object that represents the entire automated system. In SSDs and all other interaction diagrams, analysts use object notation instead of class notation. In object notation, a box refers to an individual object, not the class of all similar objects. The notation is simply a rectangle with the name of the object underlined. The colon before the underlined class name is a frequently used but optional part of the object notation to indicate that the object is an unnamed object of the class. In an SSD, the only object included is one representing the entire system: an unnamed object of the System class.

Underneath the actor and <u>:System</u> are vertical dashed lines called *lifelines*. A **lifeline**, or **object lifeline**, is simply the extension of that object—either actor or object—during the use case. The arrows between the lifelines represent the messages that are sent by the actor. Each arrow has an origin and a destination. The origin of the message is the actor or object that sends it, as indicated by the lifeline at the arrow's tail. Similarly, the destination actor or object of a message is indicated by the lifeline that is touched by the arrowhead. The purpose of lifelines is to indicate the sequence of the messages sent and received by the actor and object. The sequence of messages is read from top to bottom in the diagram.

lifeline, or **object lifeline** the vertical line under an object on a sequence diagram to show the passage of time for the object

A message is labeled to describe its purpose and any input data being sent. The message name should follow the verb-noun syntax to make the purpose clear. The syntax of the message label has several options; the simplest forms are shown in Figure 5-7. Remember that the arrows are used to represent a message and input data. But what is meant by the term *message* here? In a sequence diagram, a message is an action that is invoked on the destination object, much like a command. Notice in Figure 5-7 that the input message is called inquireOnItem.

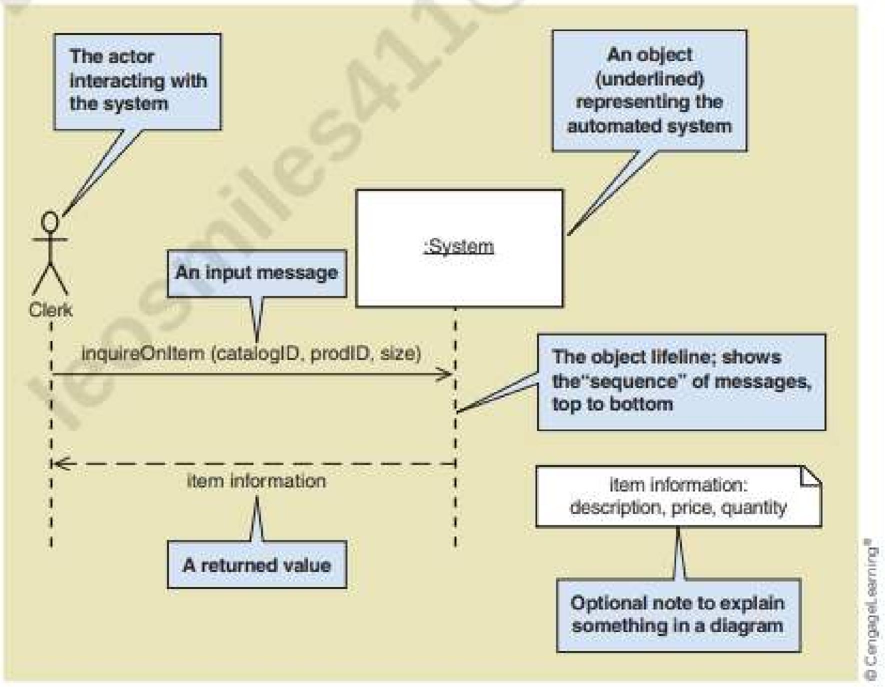

The actor interacting with the system
An object (underlined) representing the automated system
:System
An input message
Clerk
inquireOnItem (catalogID, prodID, size)
The object lifeline; shows the "sequence" of messages, top to bottom
item information
item information: description, price, quantity
A returned value
Optional note to explain something in a diagram
© Cengage Learning®

FIGURE 5-7 *Sample system sequence diagram (SSD)*