CSC 2210

Object Oriented Analysis & Design

Dr. Akinul Islam Jony

Associate Professor

Department of Computer Science, FSIT

American International University - Bangladesh (AIUB)

akinul@aiub.edu

Activity Diagram

What is Activity Diagram?

>> **Activity diagrams** are used for modeling the dynamic aspects of systems.

>> An activity diagram is essentially a flowchart, showing flow of control from activity to activity. Unlike a traditional flowchart, an activity diagram shows concurrency as well as branches of control.

>> Activity diagrams emphasize the flow of control from step to step, whereas interaction diagrams (such as sequence diagrams) emphasize the flow of control from object to object.

What is Activity Diagram?

>> Activity diagrams are object -oriented equivalent of flow charts and data-flow diagrams from structured development .

>> Activity diagrams describe the workflow behavior of a system.

>> Activity diagrams capture the process flow of a system.

>> Activity diagrams illustrate the dynamic nature of a system by modeling flow of control from activity to activity.

What is Activity Diagram?

>> Purpose of Activity Diagram:

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

What is Activity Diagram?

>> When to use activity diagrams?

→ To explore the logic of
- a complex operation.
- a complex business rule.
- a single use case.
- a several use cases.
- a business process.
- software process

→ To show the workflow and parallel processing of a system

Sequence Diagram vs. Activity Diagram

>> Sequence diagrams:

Describe interactions among objects/classes in terms of an exchange of messages over time. Sequence diagrams show a detailed flow for a specific use case or even just part of a specific use case

\>>Activity diagrams:

It illustrate the dynamic nature of a system by modeling the **flow of control from activity to activity**. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation.

Activities and Transitions

>> An ***activity*** *is a step in a process where some work or action is getting done. It can be a calculation,* finding some data, manipulating information, or verifying data.

>> The activity is represented by a rounded rectangle containing freeform text.

>> An *Activity diagram is a series of activities* linked by *transitions (sequence of execution from one activity to another activity), arrows connecting each activity.*

>> *Typically, the transition takes place* because the activity is completed. For example, you're currently in the activity "reading a page." When you finish this activity, you switch to the activity "turning page." . Figure 14-1 shows this idea graphically.

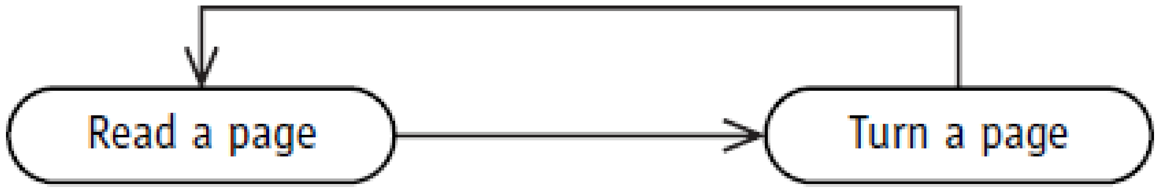


Figure 14-1 Activities and transitions

Guard Condition

>> A ***guard condition*** *can be assigned to a transition to restrict use of the transition.*

>> *Place the condition* within square brackets somewhere near the transition arrow. The condition must test true before you may follow the associated transition to the next activity.

>> The Activity diagram segment in Figure 14-2 tells you that you can't leave the table when you've finished your dinner unless you have finished your vegetables.

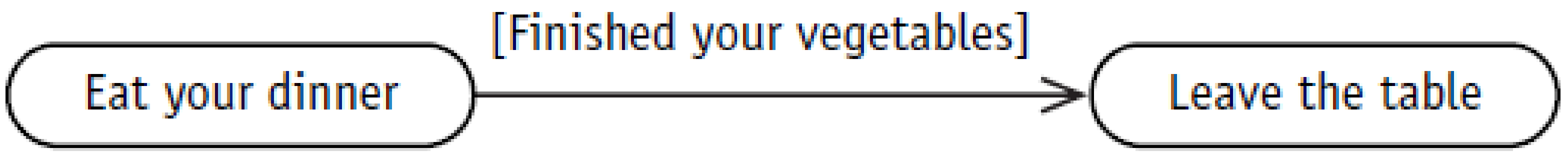


Figure 14-2 A guard condition on a transition

Decisions/Branching

>> In activity diagram, diamond is a **decision** icon, just as it is in flowcharts. In either diagram, one arrow exits the diamond for each possible value of the tested condition.

>> The decision may be as simple as a true/false (left side in figure 14-3) or decision may involve a choice between a set of options (right side in figure 14-3).

>> This is also called branching.

>> Each option is identified using a guard condition. The guard is placed on the transition that shows the direction that the logic follows if that condition is true.

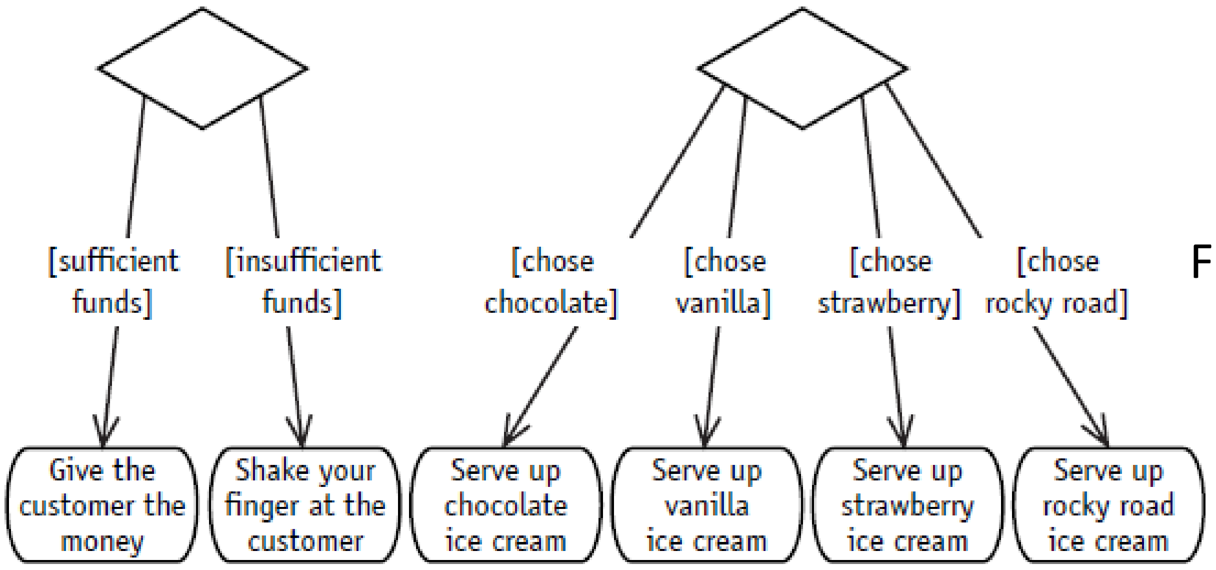

[sufficient funds]
[insufficient funds]
[chose chocolate]
[chose vanilla]
[chose strawberry]
[chose rocky road]
Give the customer the money
Shake your finger at the customer
Serve up chocolate ice cream
Serve up vanilla ice cream
Serve up strawberry ice cream
Serve up rocky road ice cream

Figure 14-3 Making a decision

Merge Point

>> In activity diagram, the **diamond** icon is also used to model a ***merge point****, the place where two alternative paths* come together and continue as one.

>> The two paths in this case are mutually exclusive routes. For example, you and I might each walk from your house to the store. I choose to walk down the left side of the street while you walk down the right. But two blocks before the store we both have to turn right and walk on the right side of that street to the store's
front door.

>> Figure 14-5 shows alternative paths merging and continuing along a single path.

>> The diamond represents the point at which the paths converge/meet/join.

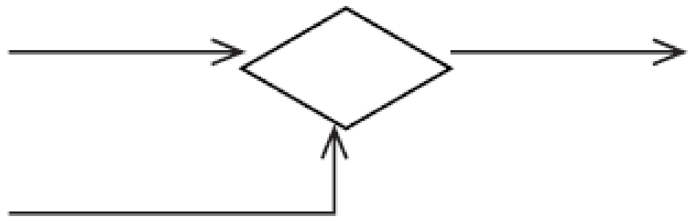


Figure 14-5 The diamond as a merge point

Start and End

>> The UML also provides icons to begin and end an Activity diagram (see Figure 14-6).

>> A solid dot indicates the beginning of the flow of activity. A bull's-eye indicates the end point.

>> There may be more than one end point in an Activity diagram. If you really want to, you can draw all your arrows to the same end point, but there is no need to do so. Every end point means the same thing: Stop here.

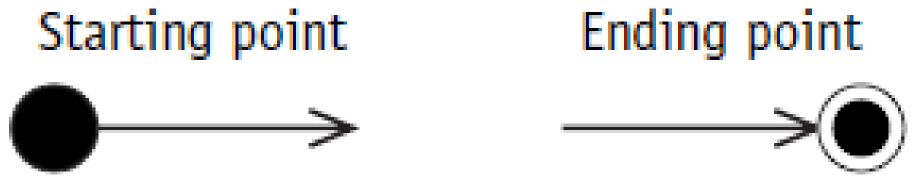


Figure 14-6 Begin and end notation for the Activity diagram

Concurrency

>> The UML notation for the Activity diagram also supports concurrency.

>> To show that a single process starts multiple concurrent threads or processes, the UML uses a simple bar called a ***fork***. *In the examples in Figure 14-7, you see one transition pointing at* the bar and multiple transitions pointing away from the bar. Each outgoing transition is a new thread or process.

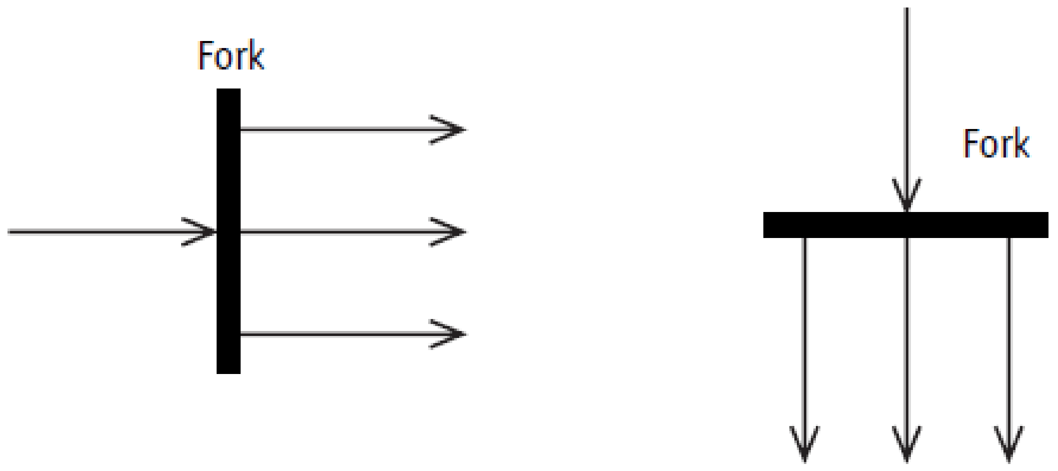


Figure 14-7 Split of control using a fork: initiating multiple threads or processes

Concurrency

>> Synchronization or merging or joining of the concurrent threads or processes is shown in much the same way.

>> Figure 14-8 shows multiple transitions pointing at the bar, this time called a *synchronization bar, and one pointing out of the bar. This indicates that the concurrent processing* has ended and the process continues as a single thread or process.

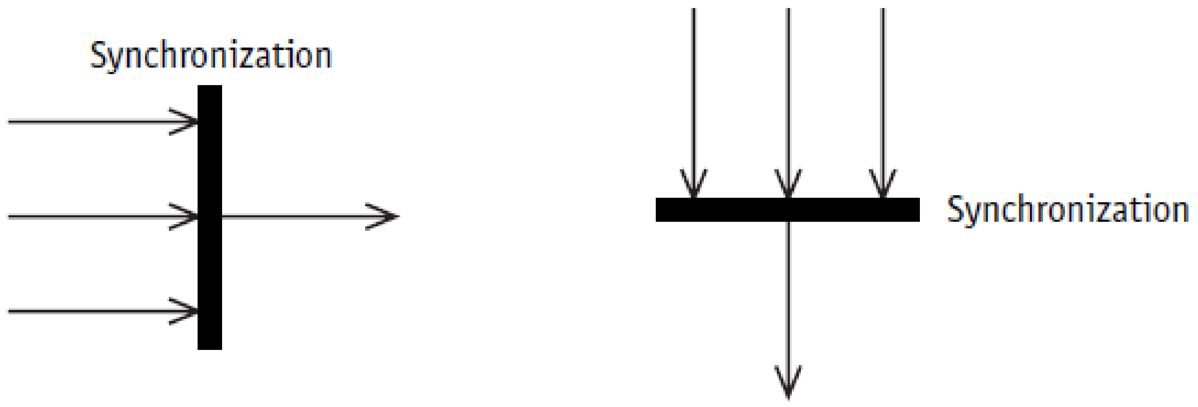


Figure 14-8 Merging or joining control using the synchronization bar

Swimlane

>> A **swimlane** is a way to group activities performed by the same actor on an activity diagram or group activities in a single thread.

>> A swimlane specifies a set of activities that share some organizational property.

>> Each swimlane has a name unique within its diagram.

>> In an activity diagram partitioned into swimlanes, every activity belongs to exactly one swimlane, but transitions may cross lanes.

Swimlane

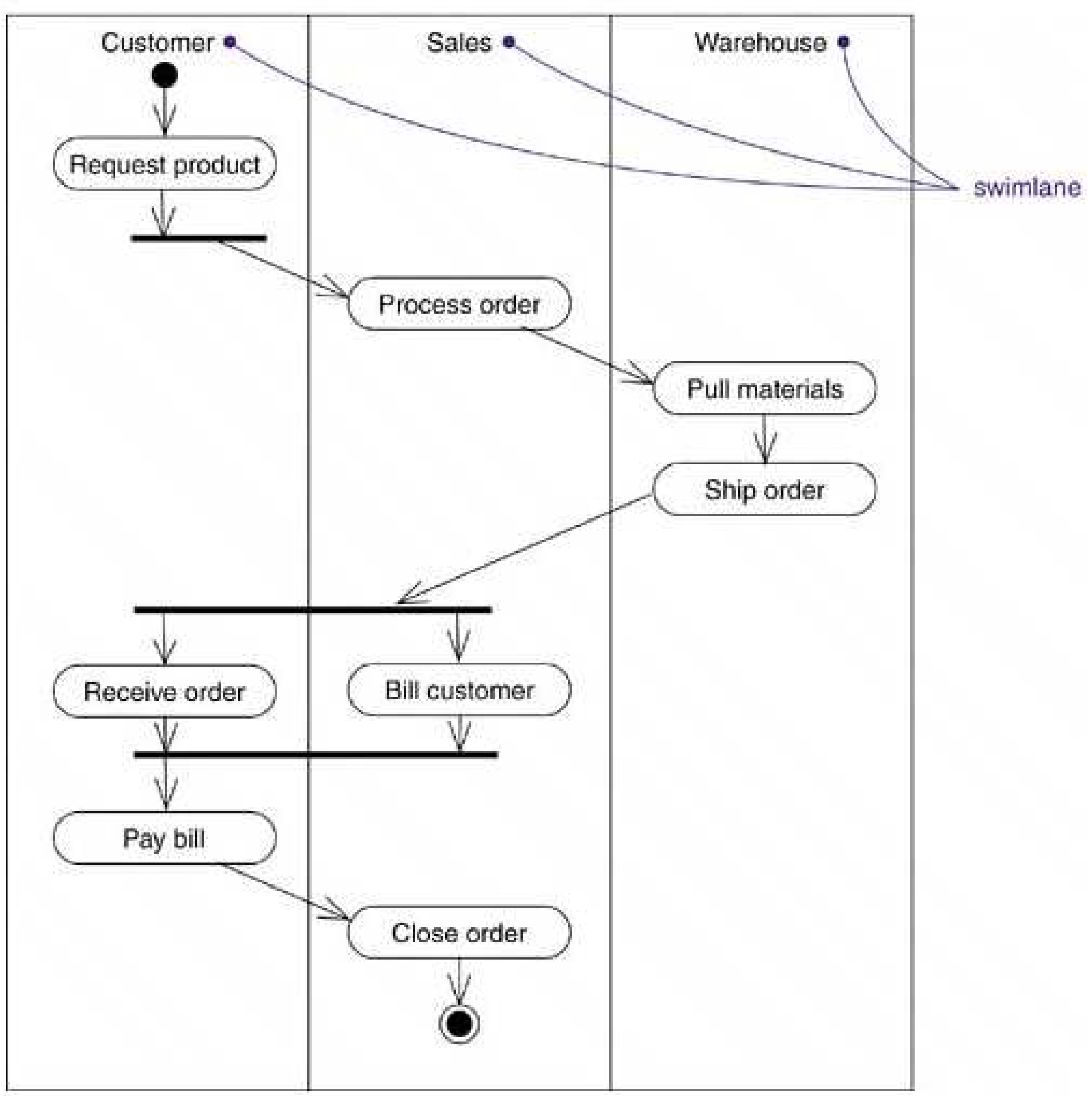


Figure 14-8 Example Of Swimlanes

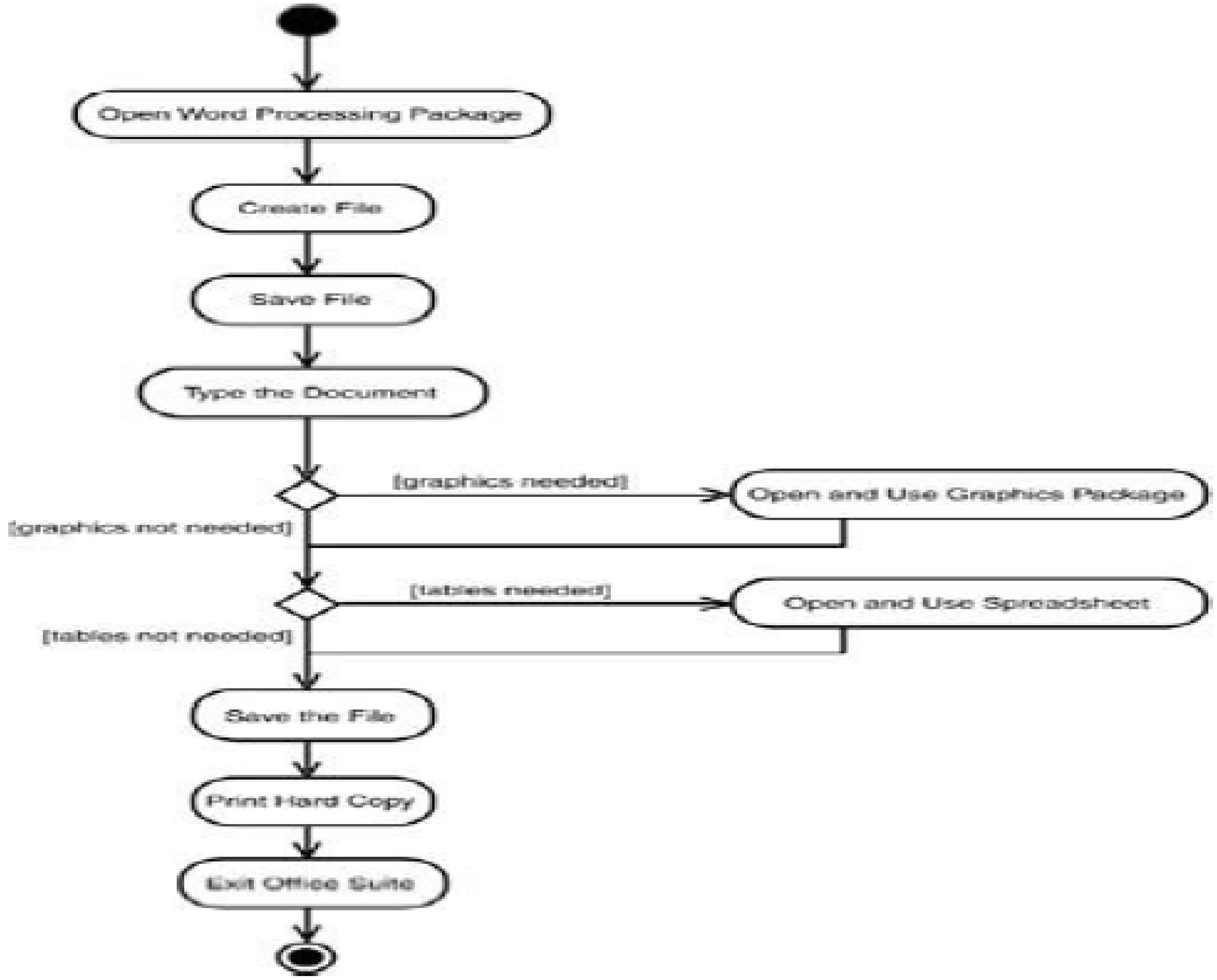
UML : ACTIVITY DIAGRAM

Example 1: creating document .

1. Open the word processing package.
2. Create a file.
3. Save the file under a unique name within its directory.
4. Type the document.
5. If graphics are necessary, open the graphics package, create the graphics, and paste the graphics into the document.
6. If a spreadsheet is necessary, open the spreadsheet package, create the spreadsheet, and paste the spreadsheet into the document.
7. Save the file.
8. Print a hard copy of the document.
9. Exit the word processing package.

UML : ACTIVITY DIAGRAM

Example 1: Creating document .



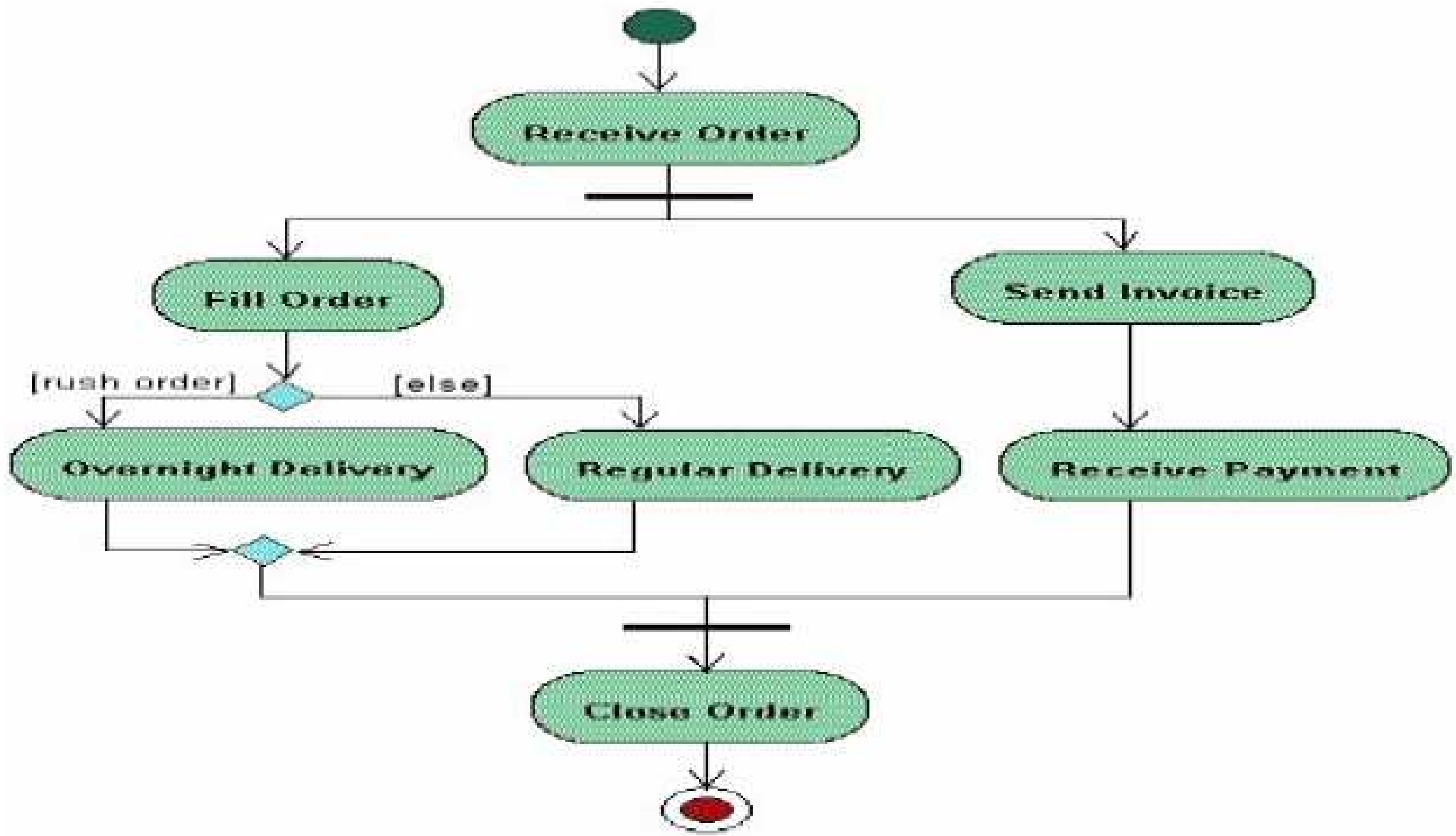
UML : ACTIVITY DIAGRAM

Example 2: processing an order

Once the order is received the activities split into two parallel sets of activities. One side fills and sends the order while the other handles the billing. On the Fill Order side, the method of delivery is decided conditionally. Depending on the condition either the Overnight Delivery activity or the Regular Delivery activity is performed. Finally the parallel activities combine to close the order.

UML : ACTIVITY DIAGRAM

Example 2: processing an order



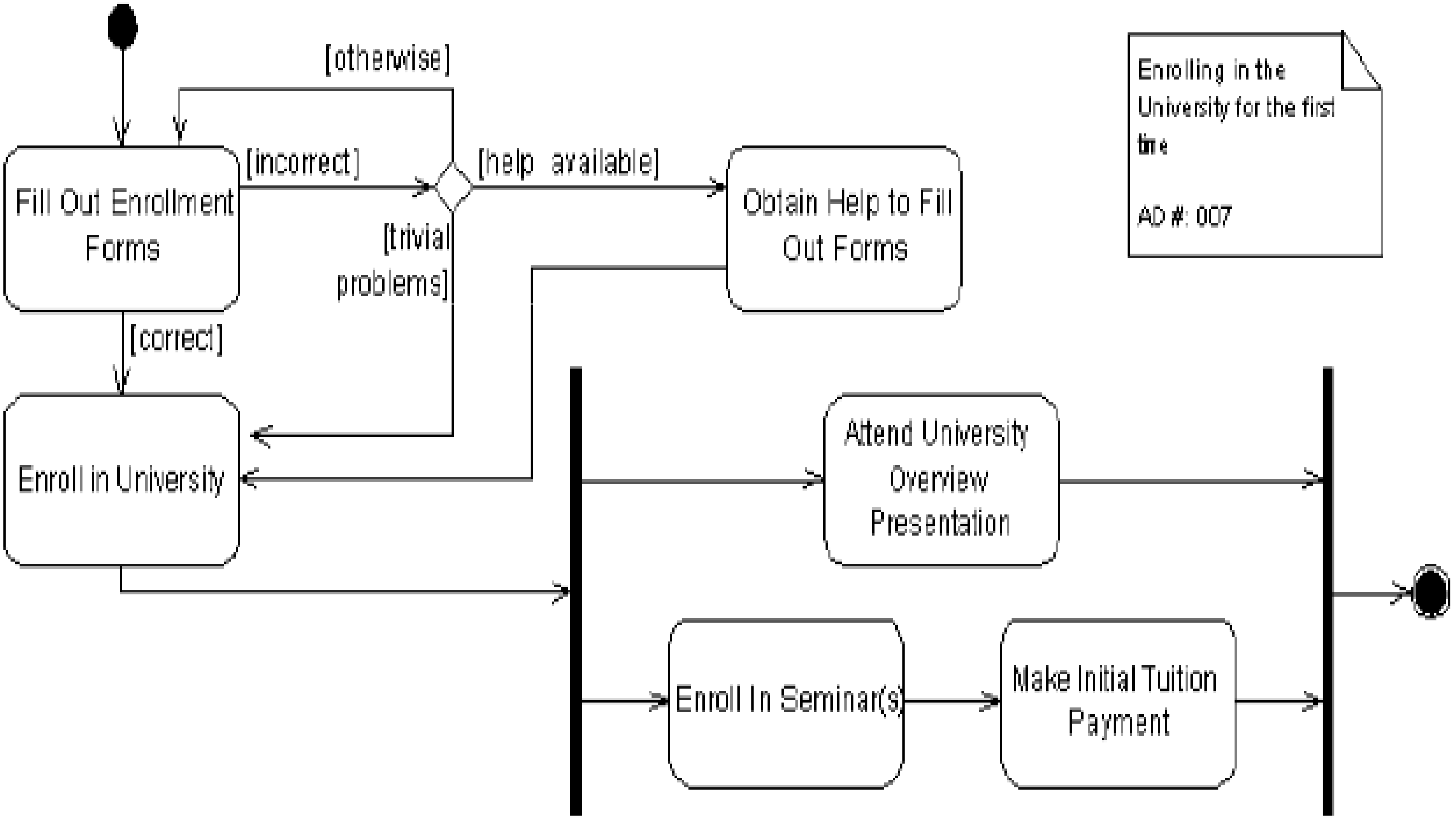
UML : ACTIVITY DIAGRAM

Example 3: Example 3: enrollment in university

1. An applicant wants to enroll in the university.
2. The applicant hands a filled out copy of form *U113 University Application Form* to the registrar.
3. The registrar inspects the forms.
4. The registrar determines that the forms have been filled out properly.
5. The registrar informs student to attend in university overview presentation.
6. The registrar helps the student to enroll in seminars
7. The registrar asks the student to pay the initial.

UML : ACTIVITY DIAGRAM

Example 3: enrollment in university



[otherwise]
Fill Out Enrollment Forms
[incorrect]
[help available]
Obtain Help to Fill Out Forms
[trivial problems]
[correct]
Enroll in University
Attend University Overview Presentation
Enroll In Seminar(s)
Make Initial Tuition Payment
Enrolling in the University for the first time
AD #: 007

UML : ACTIVITY DIAGRAM

Example 5: business process of meeting a new client

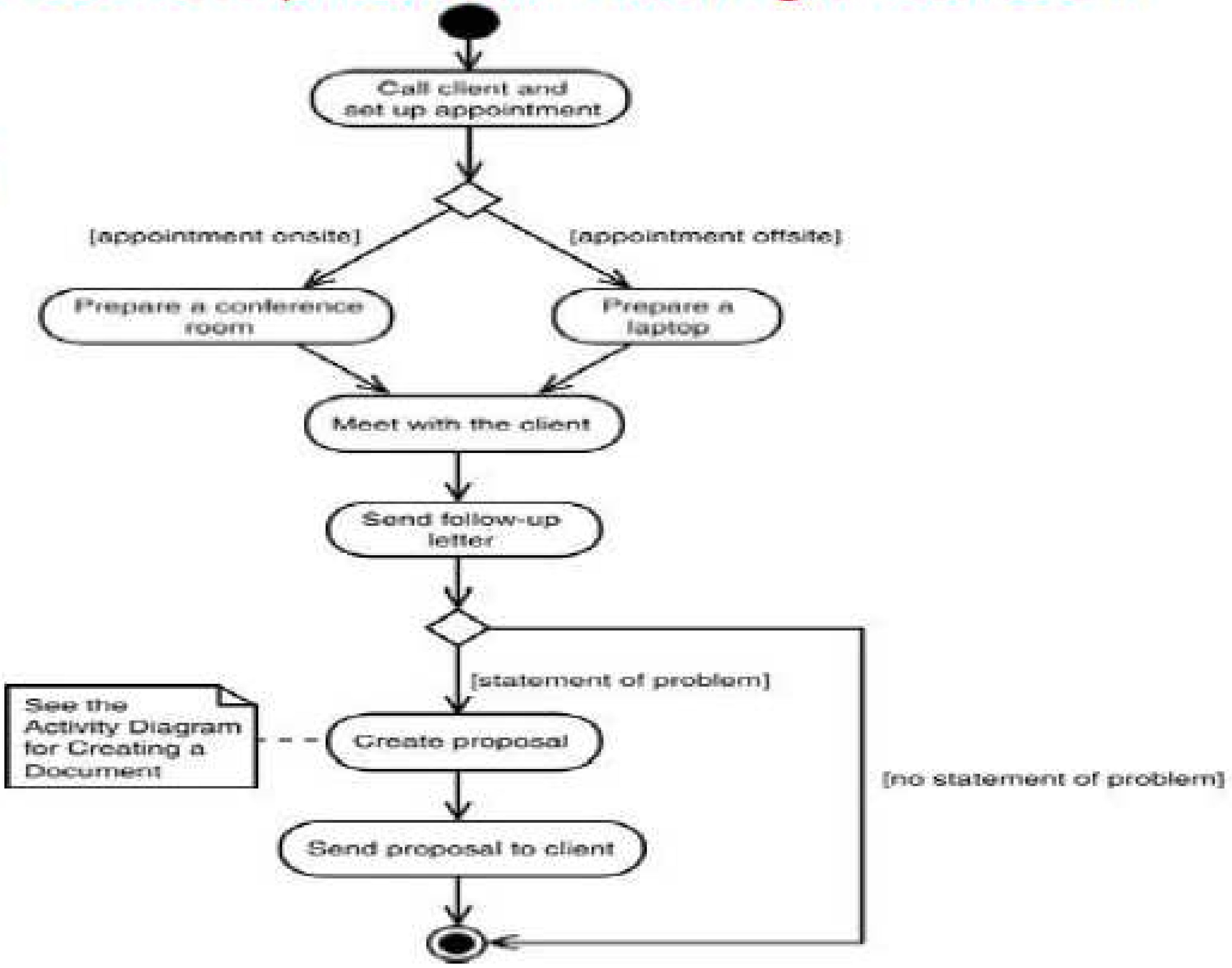
1. A salesperson calls the client and sets up an appointment.
2. If the appointment is onsite (in the consulting firm's office), corporate technicians prepare conference room for a presentation
3. If the appointment is offsite (at the client's office), a consultant prepares a presentation on a laptop.
4. The consultant and the salesperson meet with the client at the agreed-upon location and time.
5. The salesperson follows up with a letter
6. If the meeting has resulted in a statement of a problem, the consultant create a proposal and sends it to the client.

UML : ACTIVITY DIAGRAM

Example 5: business process of meeting a new client

Without Swimlane for Create a Proposal

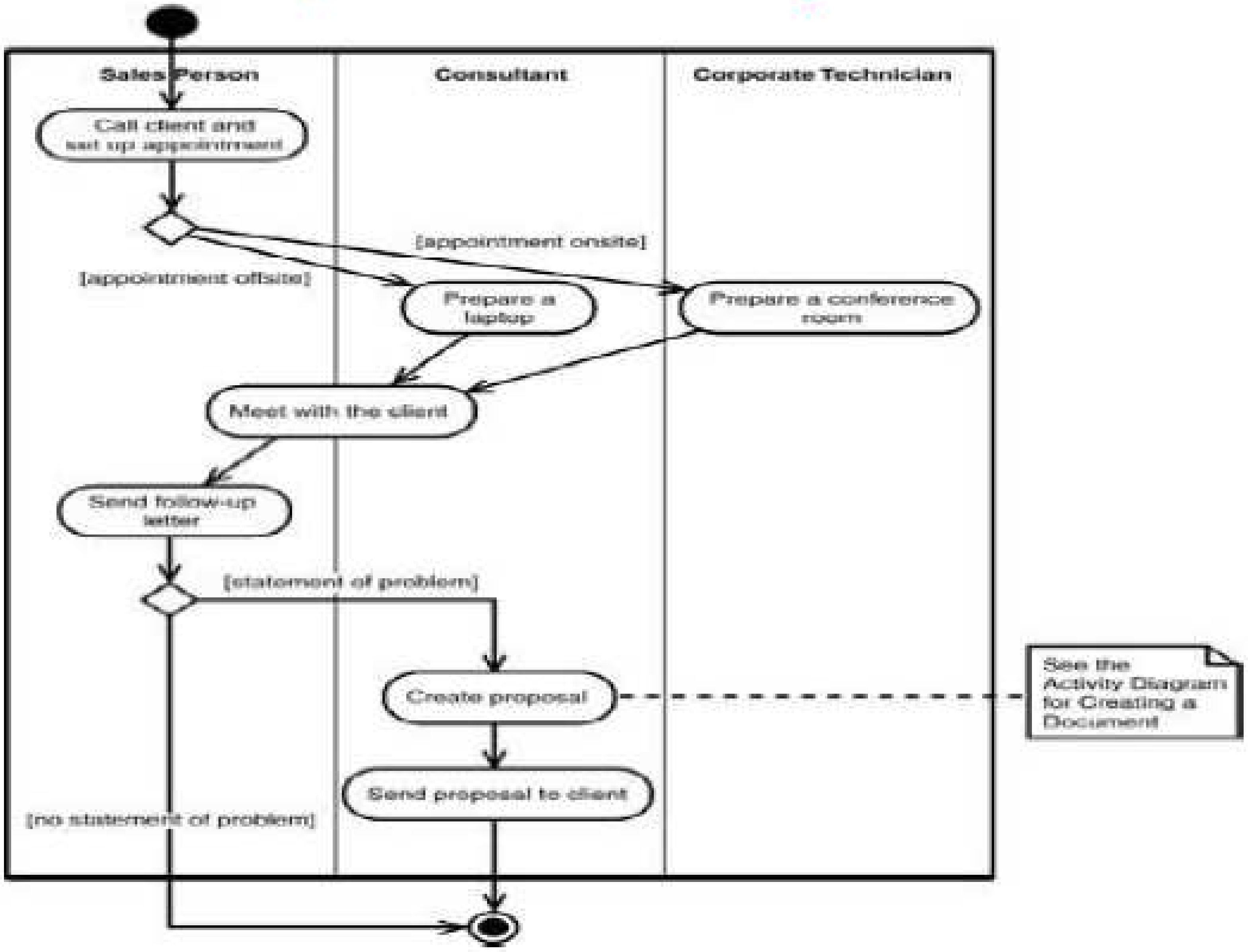
See example1



UML : ACTIVITY DIAGRAM

Example 5: business process of meeting a new client

With a swimlane



Exercises

<u>Case 1:</u>

In an online movie ticket booking system a customer books ticket using an interface. After the customer places a request for current movies he waits for the information. While the customer is waiting the server creates a Session object which then requests a list of currently running movies from the movie database. The list is then sent to customer. Customer then selects the desired movie and the Session object retrieves the available timing of the selected movie. Before displaying the time to the customer each scheduled time of the movie is checked in the Booking Database for the availability of the seats. Only the time schedules in which seats are available are displayed to the customer. The customer then selects the number of seats he wants to book. The rate of the ticket is taken from the Movie Database and the due amount is calculated by the Session object. Then the customer enters his credit card detail for the calculated amount which is verified by a creditCardAgent. If the card is verified the session object sends a receipt to the customer, writes in the booking database and update the movie database at the same time. If the card is not verified the customer request is denied. The server finally destroys the session object.

Exercises

Case 2:

In an online airlines ticket booking system, a customer places a request to the system selecting the departure and arrival destinations and also the date of departure. The system then gets all the carrier names and their time schedule from the flight database. Once the carrier names and their time schedules are received the system displays the options of the carriers and the schedule to the customer. If the customer likes any one of the options, he selects the option. But, if the customer doesn’t like any option he can go back and select a new date of departure or he can close the application. After selecting an option, the system asks for customer details and credit card information. The system verifies the credit card. If the credit card is valid, the system writes all the booking information in the booking database and sends customer information to the carrier simultaneously. If the verification of the credit card fails, the system sends an error message to the customer and requests for credit card information again. The verification process is done again. If the credit card verification fails for three times the system cancels the booking process and sends a warning message to the credit card agent at the same time. Once the booking is done the system passes the message to the interface and the interface generates a bill and sends it to the customer.

Exercises

<u>Case 3:</u>

In an ATM machine a customer starts a withdrawal transaction by inserting the card. Then he enters the pin, which is verified by the bank. If the pin is incorrect the machine requests for the pin again and the customer enters pin number. The verification repeated for 3 times for an incorrect pin number. If the pin is incorrect even in 4th attempt, the card is seized by the machine and the transaction is closed. If the pin is correct customer enters the amount he wishes to withdraw. The bank then checks the account balance of the customer. If the balance is greater than or equals to the withdrawal amount then money is dispatched through the machine and the information is written in the log concurrently. If the money is taken form the slot within 5 seconds, the account is debited and a transaction receipt is printed simultaneously. If money is not taken in 5 seconds, it is taken back by the machine. Then the balance is shown to the customer. The card and the receipt are then ejected at the same time, and the transaction is completed.

Exercises

Case 4:

A regular customer requests for product. The sales department processes the order by creating a new order object in empty state. Materials of the order are pulled out of the inventory at the warehouse and the order is set as filled. Products are then shipped by the warehouse. While the Customer receives the order, sales department prepares the bill. If the customer pays for the bill within a week sales department closes the order. If the customer doesn’t pay the bill within one week the sales department sends warning to the customer making him defaulter and closes the order in unpaid state.

References

→ **Chapter 20**
The Unified Modeling Language User Guide
SECOND EDITION
By Grady Booch, James Rumbaugh, Ivar Jacobson

→ **Session 14 & 15**
UML Weekend Crash Course
Thomas A. Pender