ACTIVITY DIAGRAMS

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using an activity diagram.

They are used in business and process modeling where their primary use is to depict the dynamic aspects of a system.

An activity diagram is very similar to a flowchart.

Purpose of Activity Diagrams

The purpose of an activity diagram can be described as −

- 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.

Activity Diagram Notations

1. **Initial State –** The starting state before an activity takes place is depicted using the initial state.

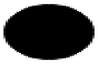


Figure – notation for initial state or start state

A process can have only one initial state unless we are depicting nested activities. We use a black filled circle to depict the initial state of a system. For objects, this is the state when they are instantiated. The Initial State from the UML Activity Diagram marks the entry point and the initial Activity State.

For example – Here the initial state is the state of the system before the application is opened.

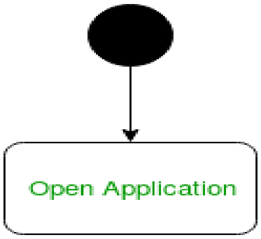


Figure – initial state symbol being used

2. **Action or Activity State –** An activity represents execution of an action on objects or by objects. We represent an activity using a rectangle with rounded corners. Basically any action or event that takes place is represented using an activity.

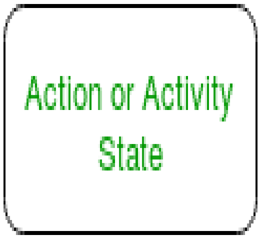


Figure – notation for an activity state

For example – Consider the previous example of opening an application opening the application is an activity state in the activity diagram.

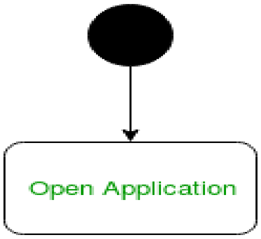


Figure – activity state symbol being used

3. **Action Flow or Control flows –** Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.

Figure – notation for control Flow

An activity state can have multiple incoming and outgoing action flows. We use a line with an arrow head to depict a Control Flow. If there is a constraint to be adhered to while making the transition it is mentioned on the arrow.

Consider the example – Here both the states transit into one final state using action flow symbols i.e. arrows.

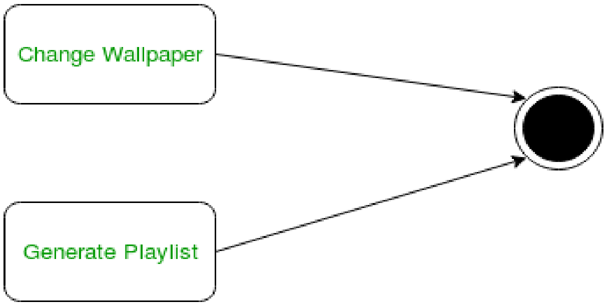


Figure – using action flows for transitions

4. **Decision node and Branching –** When we need to make a decision before deciding the flow of control, we use the decision node.

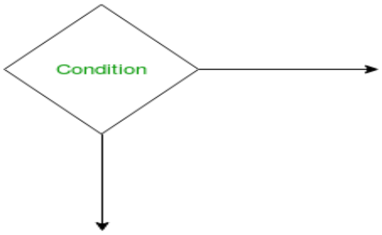


Figure – notation for decision node

The outgoing arrows from the decision node can be labeled with conditions or guard expressions. It always includes two or more output arrows.

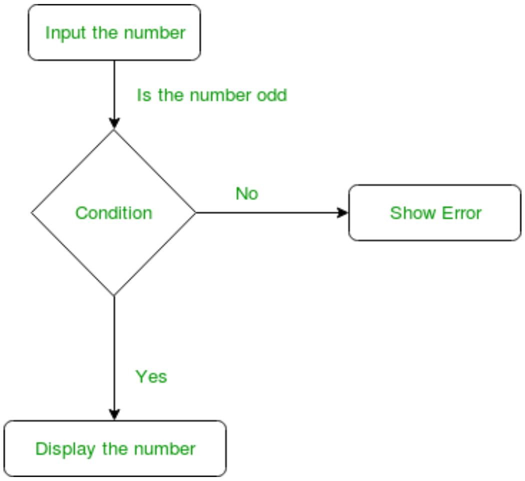


Figure – an activity diagram using decision node

5. **Guards –** A Guard refers to a statement written next to a decision node on an arrow sometimes within square brackets.

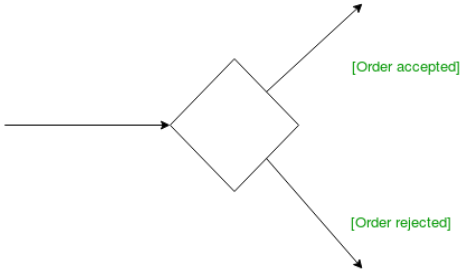


Figure – guards being used next to a decision node

The statement must be true for the control to shift along a particular direction. Guards help us know the constraints and conditions which determine the flow of a process.

6. **Fork –** Fork nodes are used to support concurrent activities.

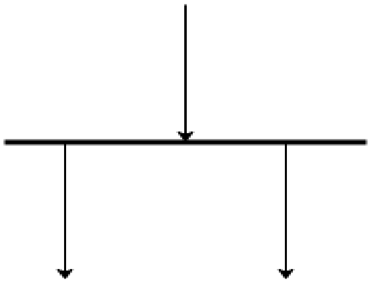


Figure – fork notation

We a fork node when both the activities get executed concurrently i.e. no decision is made before splitting the activity into two parts. Both parts need to be executed in case of a fork statement.

We use a rounded solid rectangular bar to represent a Fork notation with incoming arrow from the parent activity state and outgoing arrows towards the newly created activities.

For example: In the example below, the activity of making coffee can be split into two concurrent activities and hence we use the fork notation.

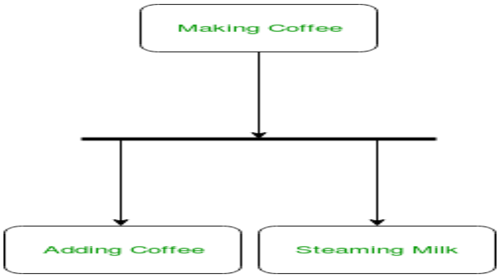


Figure – a diagram using fork

7. **Join –** Join nodes are used to support concurrent activities converging into one. For join notations we have two or more incoming edges and one outgoing edge.

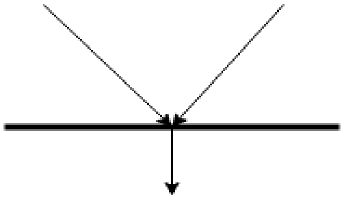


Figure – join notation

For example – When both activities i.e. steaming the milk and adding coffee get completed, we converge them into one final activity.

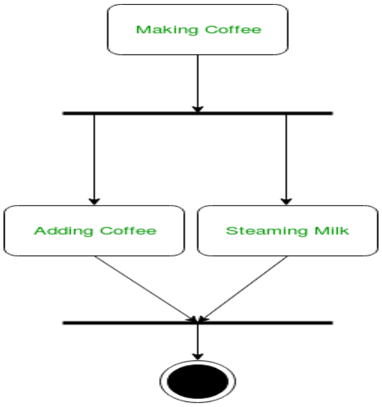


Figure – a diagram using join notation

8. **Merge or Merge Event –** Scenarios arise when activities which are not being executed concurrently have to be merged. We use the merge notation for such scenarios. We can merge two or more activities into one if the control proceeds onto the next activity irrespective of the path chosen.

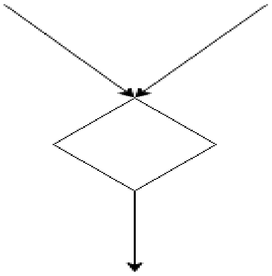


Figure – merge notation

For example – In the diagram below: we can’t have both sides executing concurrently, but they finally merge into one. A number can’t be both odd and even at the same time.

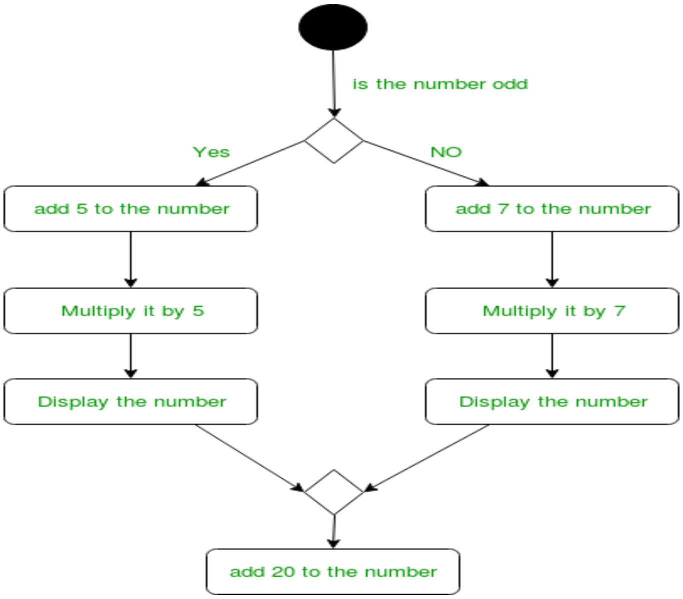


Figure – an activity diagram using merge notation

9. **Swimlanes –** We use swimlanes for grouping related activities in one column. Swimlanes group related activities into one column or one row. Swimlanes can be vertical and horizontal. Swimlanes are used to add modularity to the activity diagram. It is not mandatory to use swimlanes. They usually give more clarity to the activity diagram. It's similar to creating a function in a program. It's not mandatory to do so, but, it is a recommended practice.

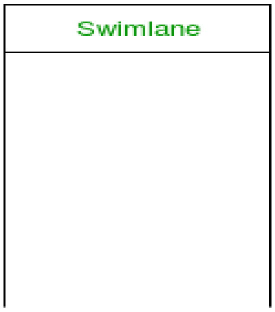


Figure – swimlanes notation

We use a rectangular column to represent a swimlane as shown in the figure above.

For example – Here different set of activities are executed based on if the number is odd or even. These activities are grouped into a swimlane.

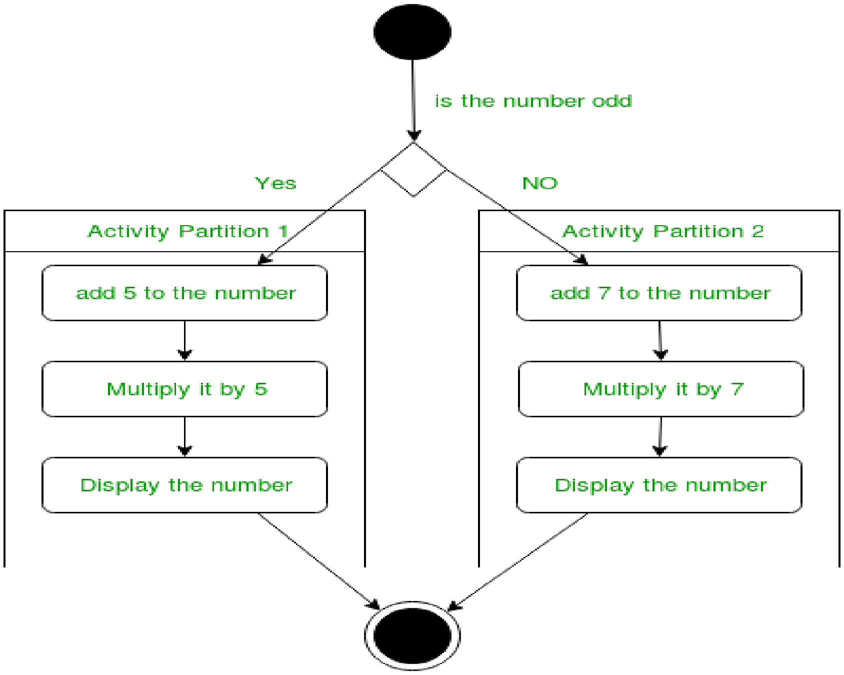


Figure – an activity diagram making use of swimlanes

10. **Time Event**

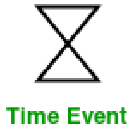


Figure – time event notation

We can have a scenario where an event takes some time to complete. We use an hourglass to represent a time event.

For example – Let us assume that the processing of an image takes a lot of time. Then it can be represented as shown below.

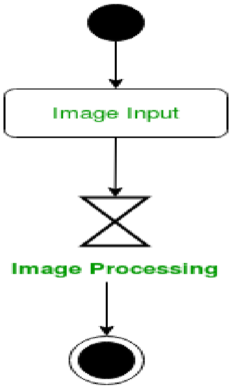


Figure – an activity diagram using time event

11. **Final State or End State –** The state which the system reaches when a particular process or activity ends is known as a Final State or End State. We use a filled circle within a circle notation to represent the final state in a state machine diagram. A system or a process can have multiple final states.

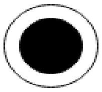


Figure – notation for final state

How to draw an activity diagram

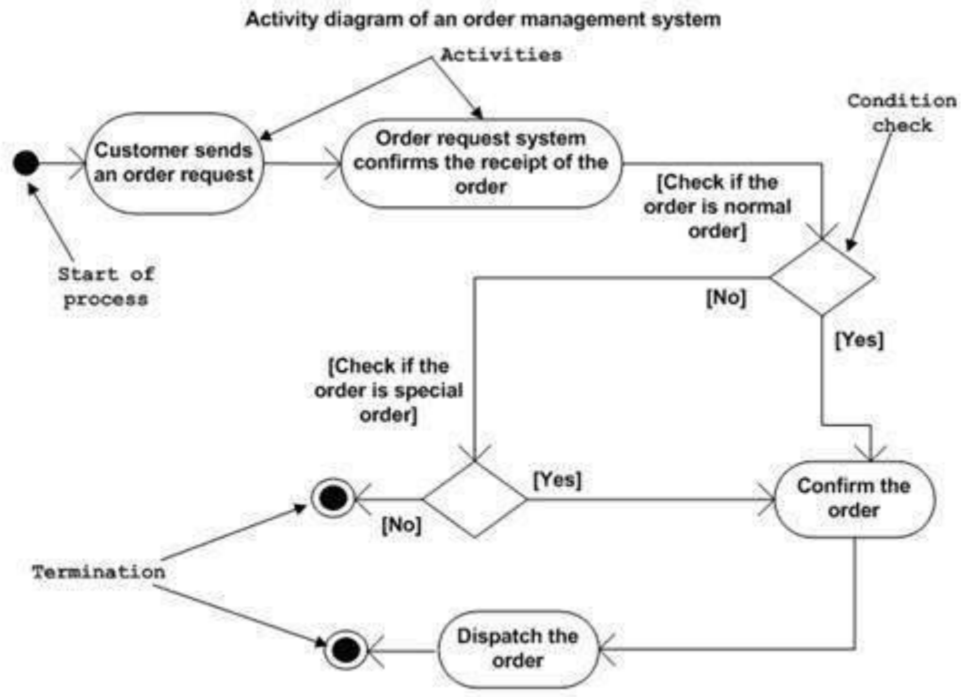
1. Identify the initial state and the final states.
2. Identify the intermediate activities needed to reach the final state from the initial state.
3. Identify the conditions or constraints which cause the system to change control flow.
4. Draw the diagram with appropriate notations.

Following is an example of an activity diagram for order management system. In the diagram, four activities are identified which are associated with conditions. The activity diagram is made to understand the flow of activities and is mainly used by the business users

Following diagram is drawn with the four main activities −

- Send order by the customer
- Receipt of the order
- Confirm the order
- Dispatch the order

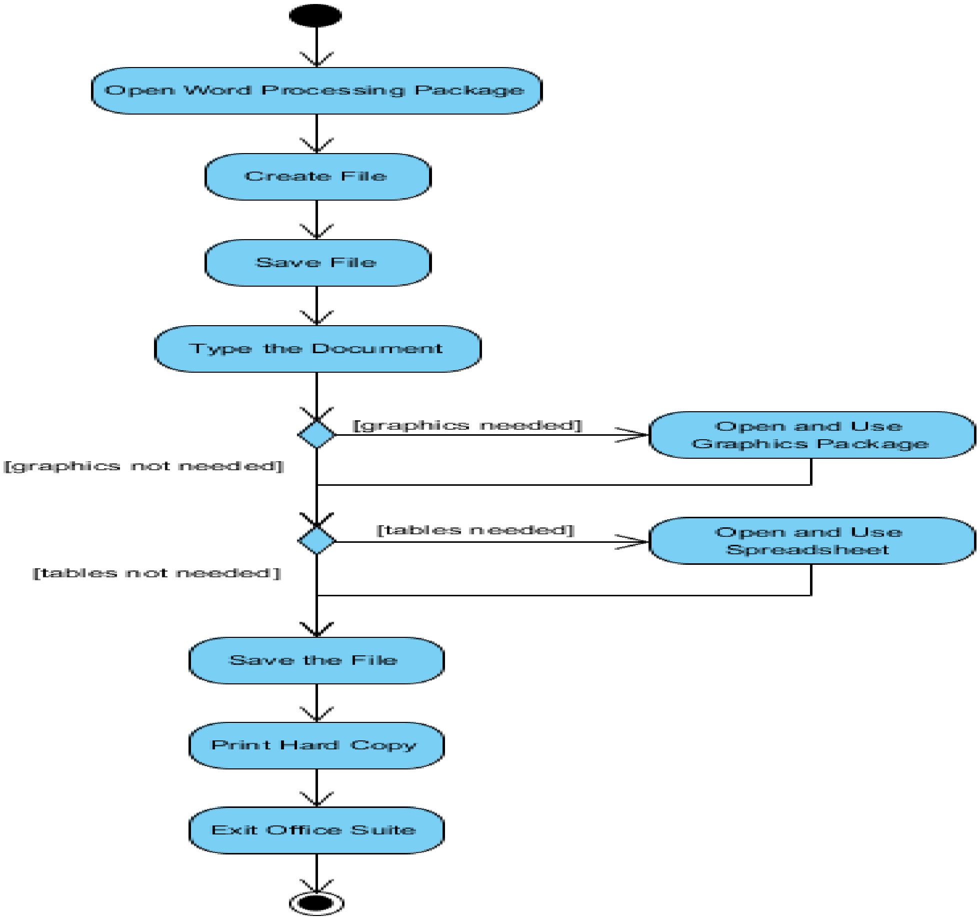
After receiving the order request, condition checks are performed to check if it is normal or special order. After the type of order is identified, dispatch activity is performed and that is marked as the termination of the process.


Activity diagram of an order management system
Activities
Condition check
Customer sends an order request
Order request system confirms the receipt of the order
[Check if the order is normal order]
Start of process
[No]
[Yes]
[Check if the order is special order]
[Yes]
[No]
Confirm the order
Termination
Dispatch the order

Activity Diagram - Modeling a Word Processor

The activity diagram example below describes the workflow for a word process to create a document through the following steps:

- Open the word processing package.
- Create a file.
- Save the file under a unique name within its directory.
- Type the document.
- If graphics are necessary, open the graphics package, create the graphics, and paste the graphics into the document.
- If a spreadsheet is necessary, open the spreadsheet package, create the spreadsheet, and paste the spreadsheet into the document.
- Save the file.
- Print a hard copy of the document.
- Exit the word processing package.



Activity Diagram Example - Process Order

Given the problem description related to the workflow for processing an order, let's model the description in visual representation using an activity diagram:

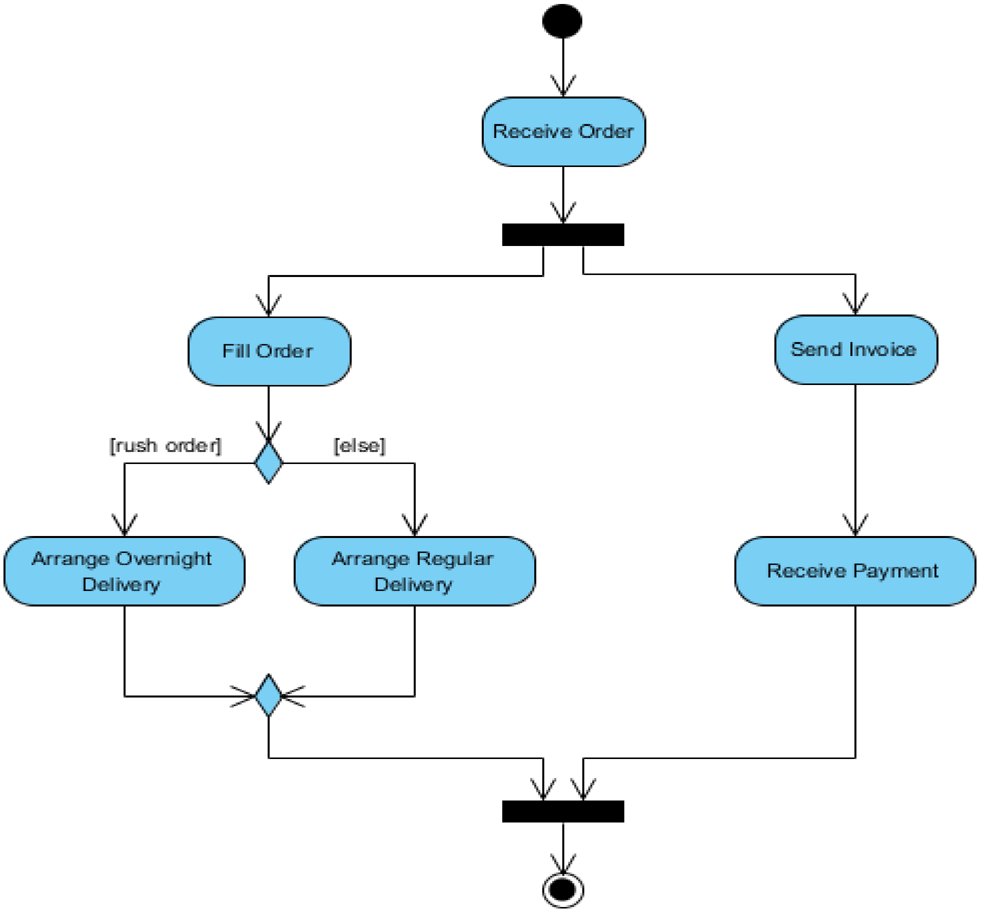
Process Order - Problem Description

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.

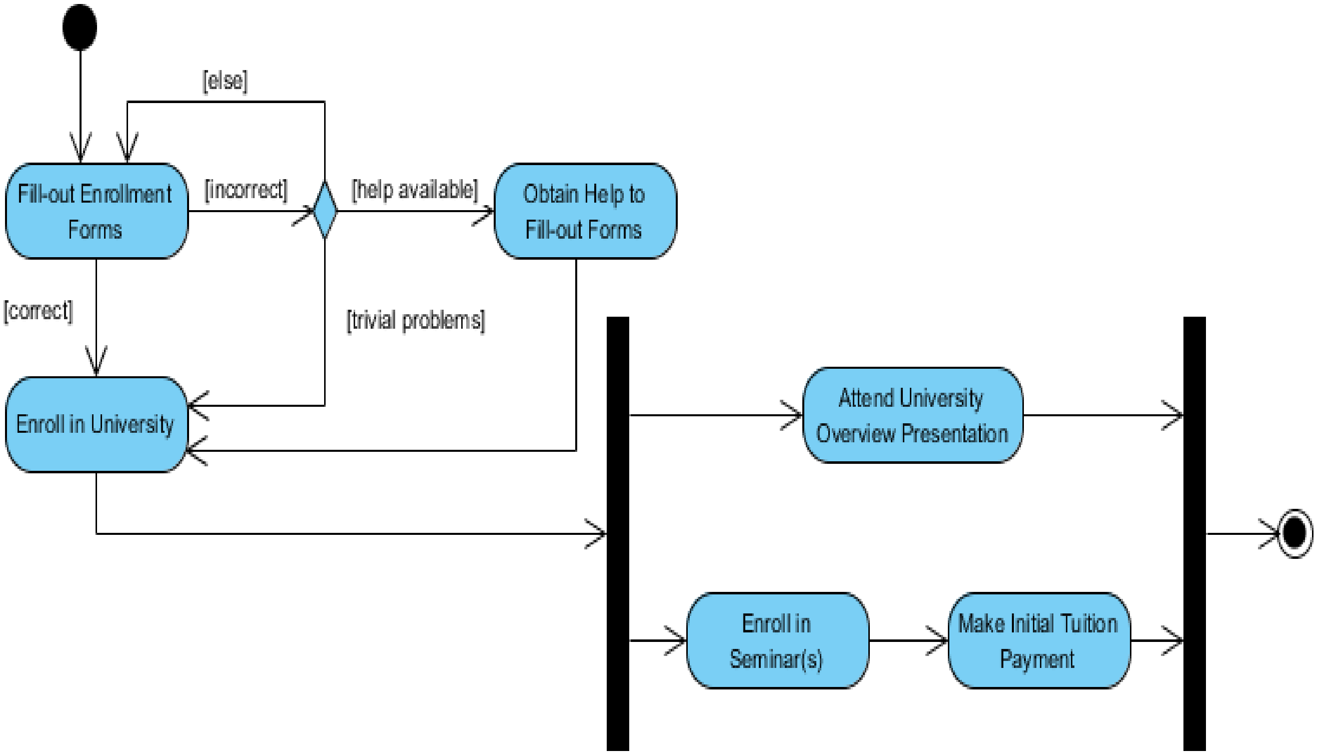
The activity diagram example below visualizes the flow in graphical form.



Activity Diagram Example - Student Enrollment

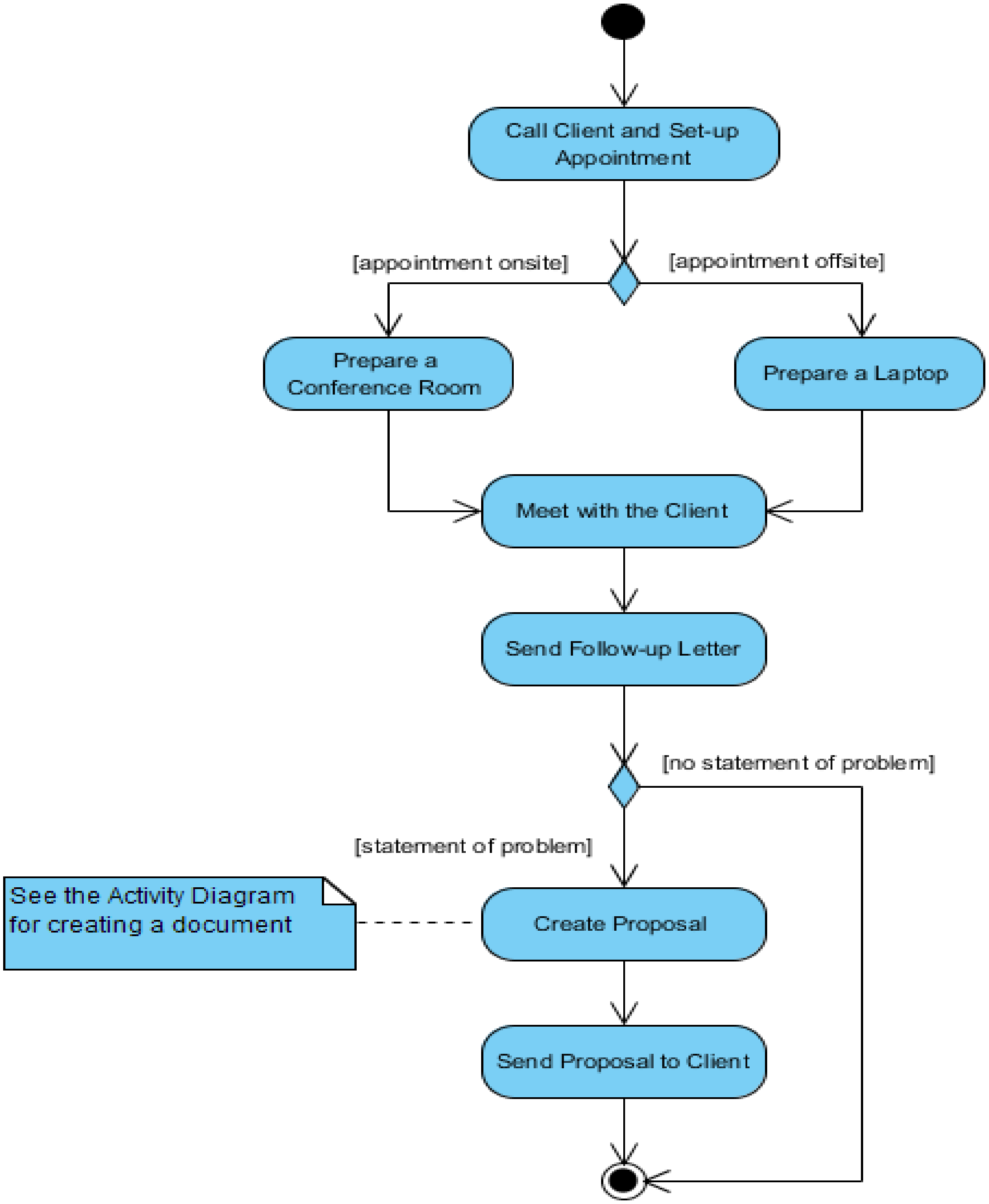
This UML activity diagram example describes a process for student enrollment in a university as follows:

- An applicant wants to enroll in the university.
- The applicant hands a filled out copy of Enrollment Form.
- The registrar inspects the forms.
- The registrar determines that the forms have been filled out properly.
- The registrar informs student to attend in university overview presentation.
- The registrar helps the student to enroll in seminars
- The registrar asks the student to pay for the initial tuition.

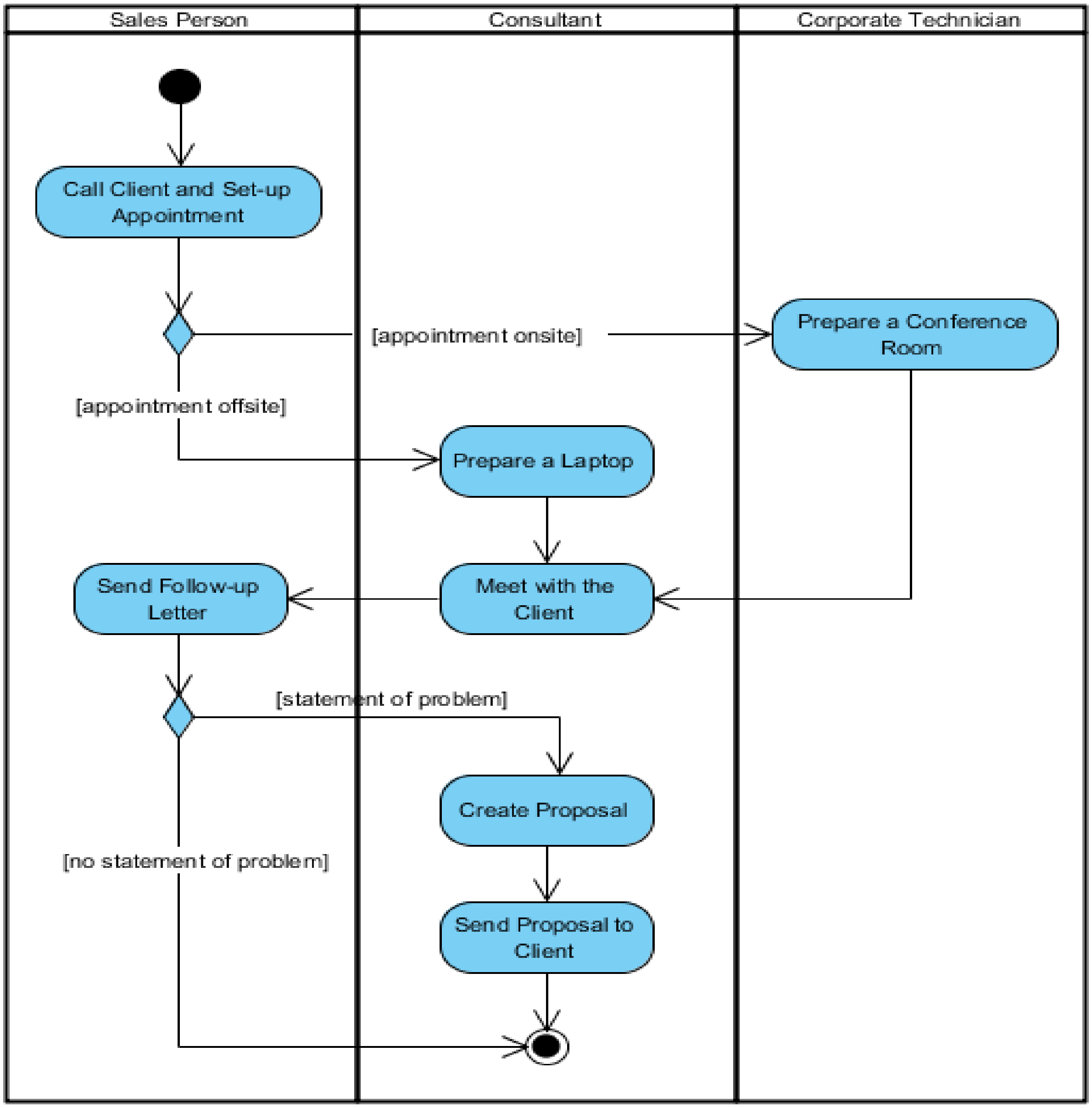

[else]
Fill-out Enrollment Forms
[incorrect]
[help available]
Obtain Help to Fill-out Forms
[correct]
[trivial problems]
Enroll in University
Attend University Overview Presentation
Enroll in Seminar(s)
Make Initial Tuition Payment

Swimlane and Non-Swimlane Activity Diagram

The activity diagram example below describes the business process for meeting a new client using an activity Diagram without swimlane.

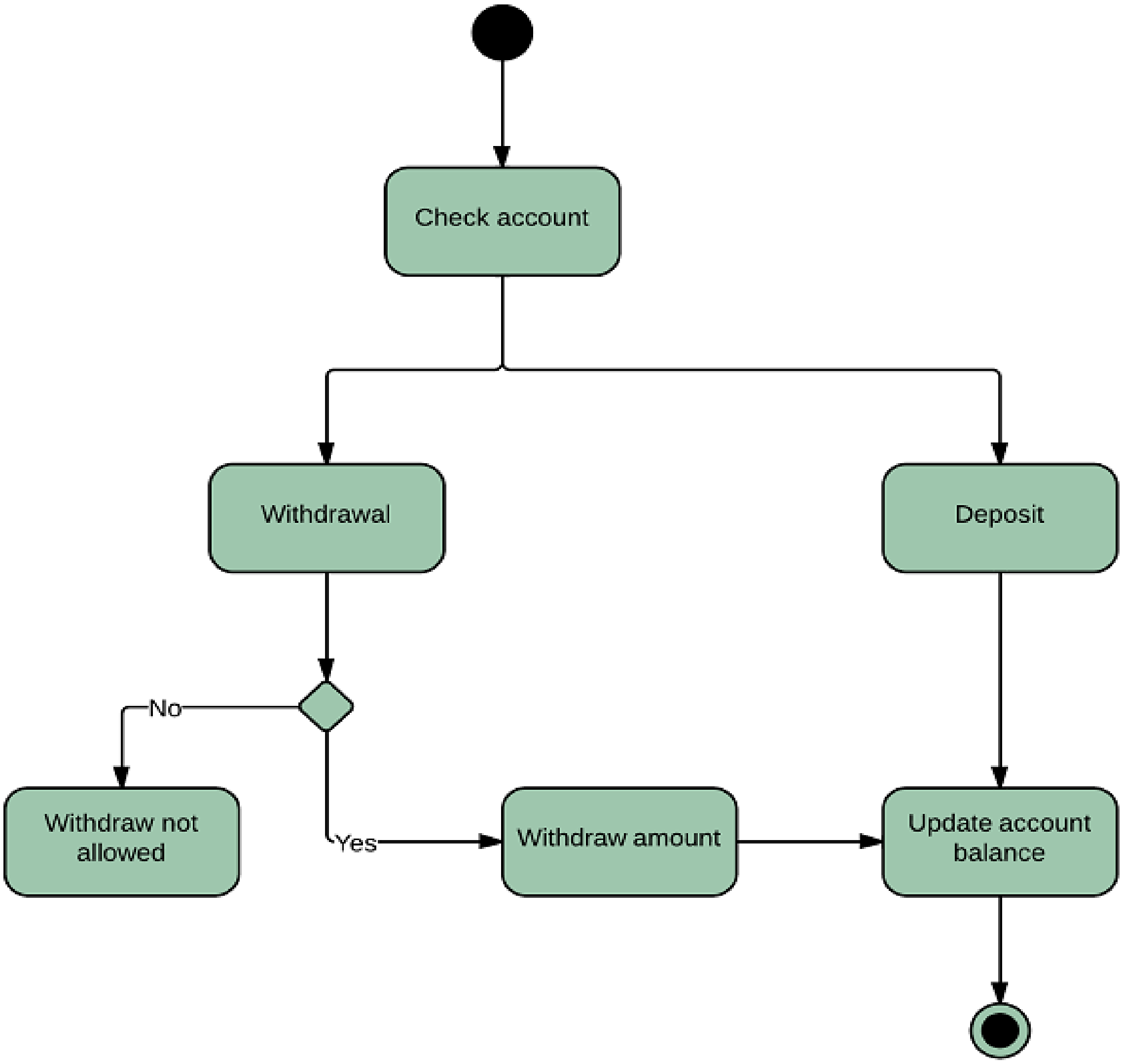


This figure below describes the business process for meeting a new client using an activity Diagram with swimlane.

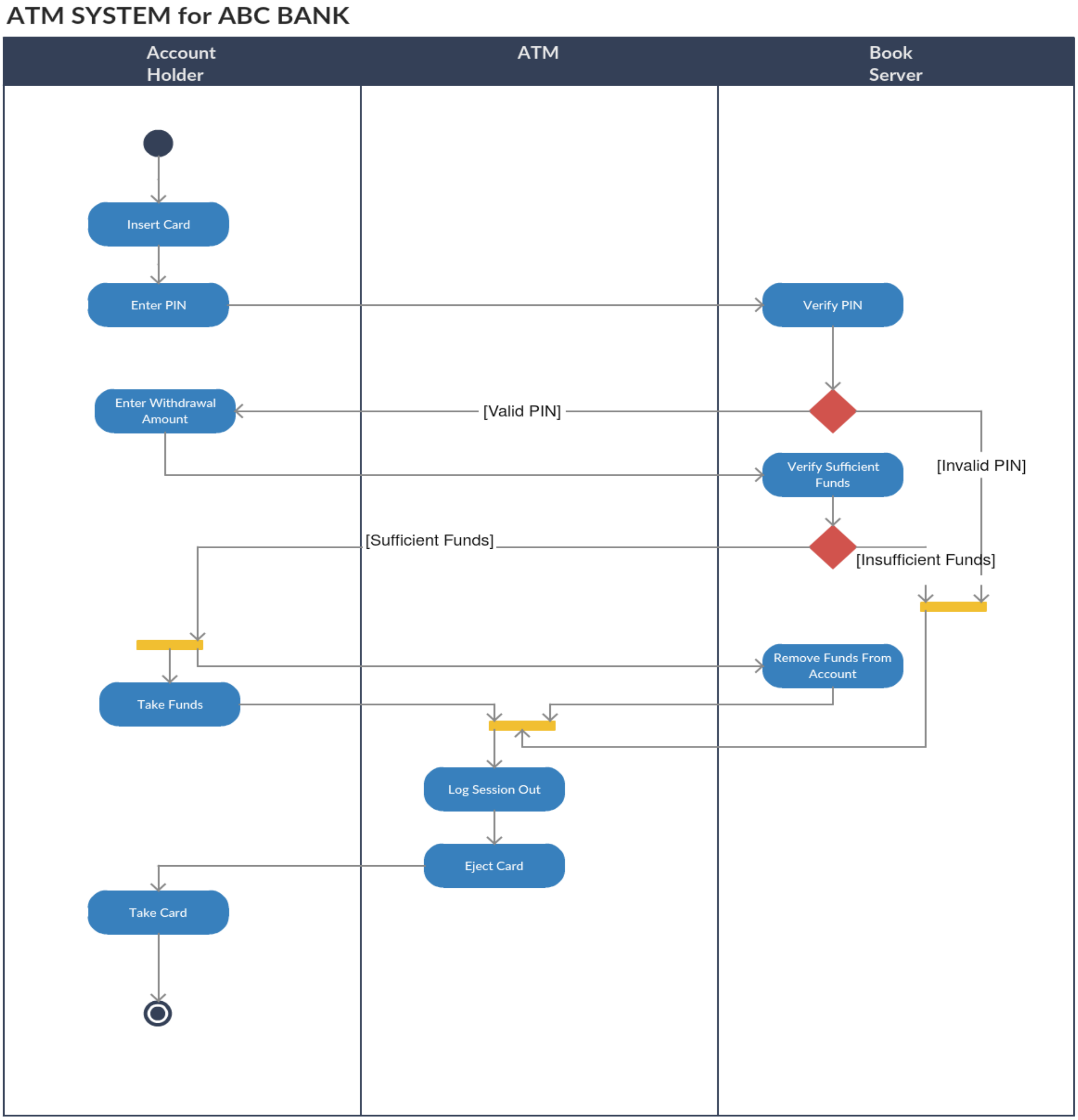


Activity diagram for a banking system

This diagram shows the process of either withdrawing money from or depositing money into a bank account. An advantage of representing the workflow visually in UML is the ability to show withdrawals and deposits on one chart.



Activity Diagram for ATM


ATM SYSTEM for ABC BANK
Account Holder
ATM
Book Server
Insert Card
Enter PIN
Verify PIN
Enter Withdrawal Amount
[Valid PIN]
[Invalid PIN]
Verify Sufficient Funds
[Sufficient Funds]
[Insufficient Funds]
Remove Funds From Account
Take Funds
Log Session Out
Eject Card
Take Card