BRAC UNIVERSITY

Inspiring Excellence

UML Diagrams

Ref: Whitten et all, *Systems Analysis and Design Methods 7e*. McGraw-Hill Higher Education

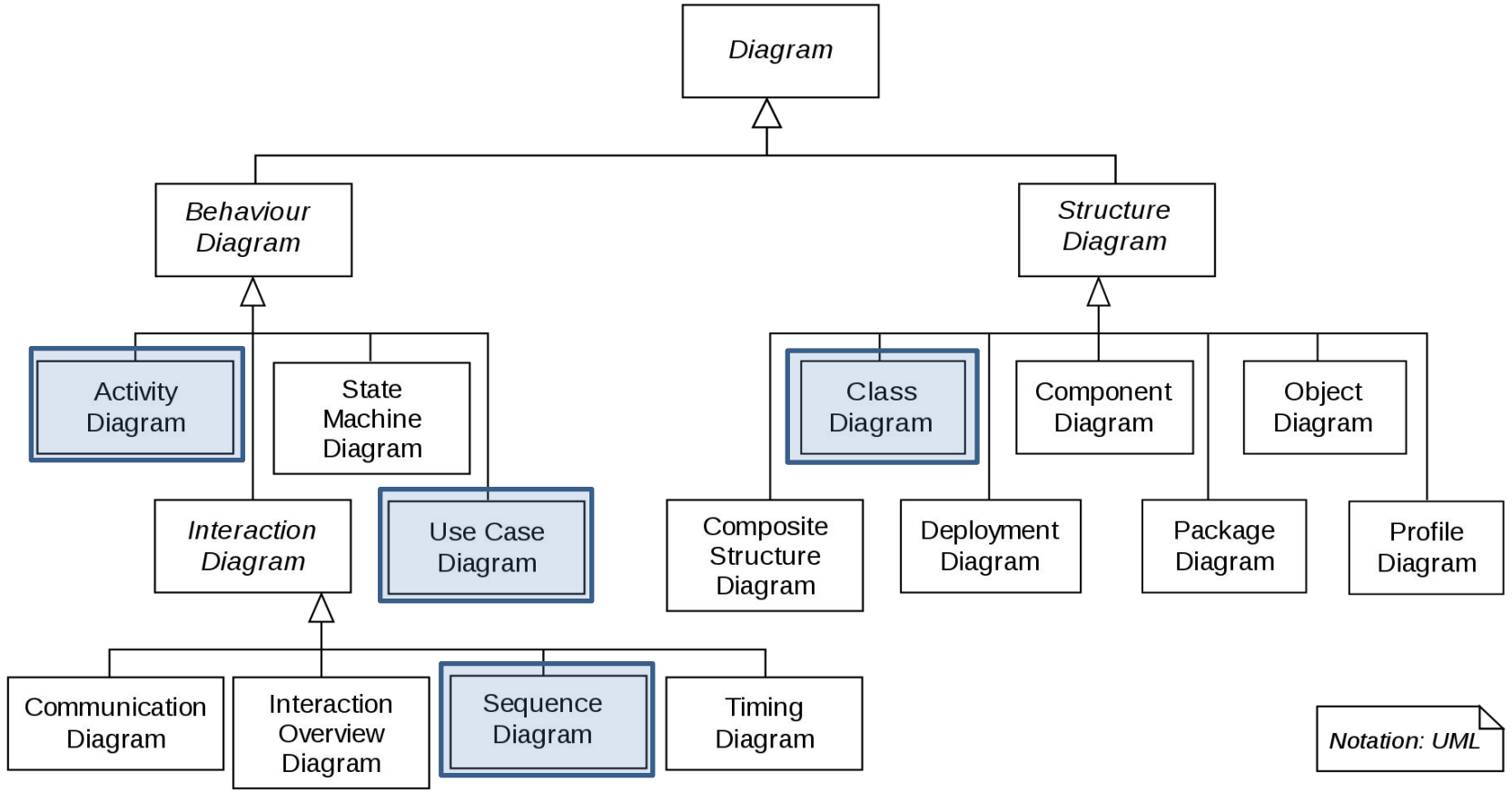
Key Ideas

- Introduction to UML diagrams
- Use case diagram
- Sequence diagram
- Class diagram
- & Activity diagram

UML diagram

- Stands for Unified Modeling Language
  - A *language* with notion which can be *understand* by all the parties involved with software
- Initiated by Rational Software in 1994-95
- Popular tools –
  - Rational Rose
  - Microsoft Visio
  - Draw.io

Different Types of UML



BRAC UNIVERSITY
Inspiring Excellence

Use case diagram

What is Use Case ?

- Use case diagrams considers mostly as a requirement analysis tools
- It identifies the *uses* of the system based on a *case*
  - Use indicates the actions
  - Case indicates the action linked with *actor* (who performs the action)
  - All the action verbs mentioned in requirement specification is an use case in this diagram
- To draw the use case diagram, one need to
  - Identify all the entity who will performs action
  - And all the actions needed to support by the system

Components of Use Case diagram

- Use case diagram is composed of ***four*** components
  - Actor
  - Use case
  - System boundary
  - Relations

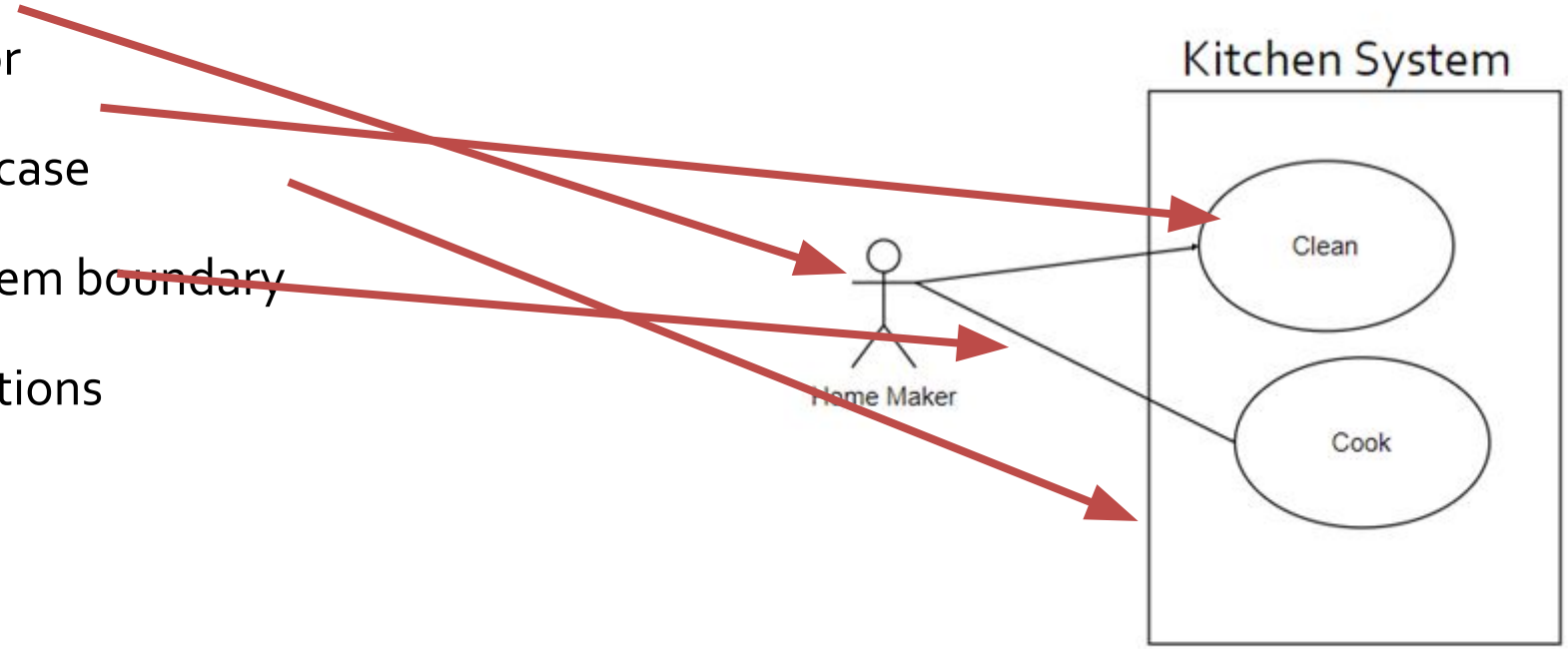


Figure: Example Use Case diagram

Actor

- The entity which *performs the actions* or roles in the system
- Actor is responsible for *giving input* to the system
- Responsible to *use processed output* for performing particular action
- Actor must be connected with at least one use case
-
  - Primary actor
  - Secondary actor
  - External hardware
  - Other System

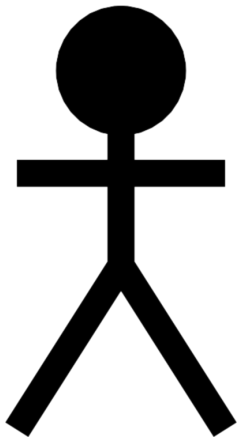


Figure: Actor in Usecase

Types of Actor

BRAC UNIVERSITY
Inspiring Excellence

Primary actor

- People who performs the **main system functions**.
- For example **rider** of a ride sharing system. As s/he is requesting for ride, or paying the money using the system.

Secondary actor

- People who performs the **administrative functions**.
- For the aforementioned system a **manager** who sets the discounts is an example of Secondary actor.

Types of Actor (cont.)

BRAC UNIVERSITY
Inspiring Excellence

External hardware

- Any **external hardware device** which is a part of the application.
- If the system using **amazon datastore** as their database.

Other System

- Any **external system** which has interaction with the current system.
- **Payment gateway** is an example of such actor.

Use Cases

- Indicates the *system functions* performed by an actor
- It can also describes the *sequence of actions* in a system
- Every Use case must have a *unique name*
- Use case must be started with *principal verb*
- Use cases in the diagram must be *enclosed* by the system boundary
- Every Use case should be *connected* with either *actor* or another *use*

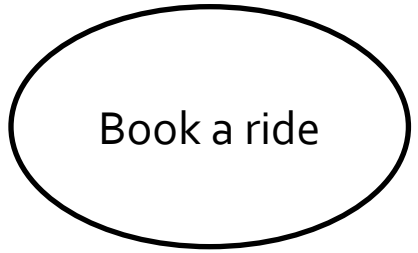


Figure: An Use case

System Boundary

- Shows how the system *interacts* with the user
- Class in which *use case are executed*
- Represented by the use cases within a *rectangle* and actors will *outside* of the system boundary

Figure: The system boundary of Use case

Relation

- Also Known as by *communication line*
- It represents the connection between any two components of use case diagram
- Can be of three types
  - Association
  - Generalization
  - Dependency

Association

- Connects an actor with the use case
- Identifies the actor(s) are responsible/user of the use case
- Represented by a *straight solid line*
  - No arrow
  - Not dashed or curved line
- A actor must have at least one association in the diagram
- An use case can be associated with zero or more actors

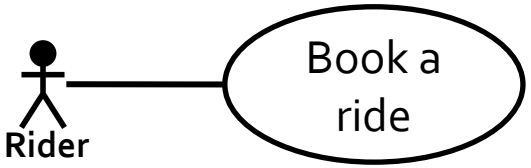


Figure: association between an actor and an use case

Generalization

- Represents the *parent-child relation*
- Represented by a straight line with hollow arrow
- Can indicate the relation between
  - Either **Actors**
  - Or **Use cases**

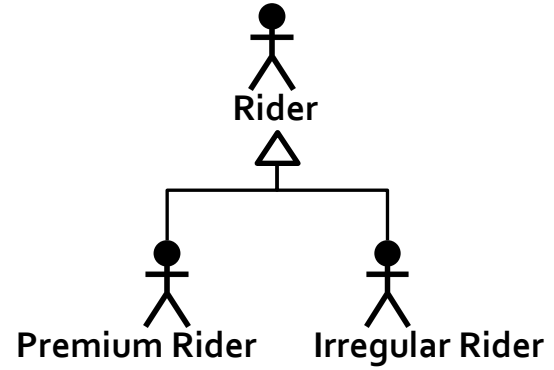


Figure: Generalization relation between actors

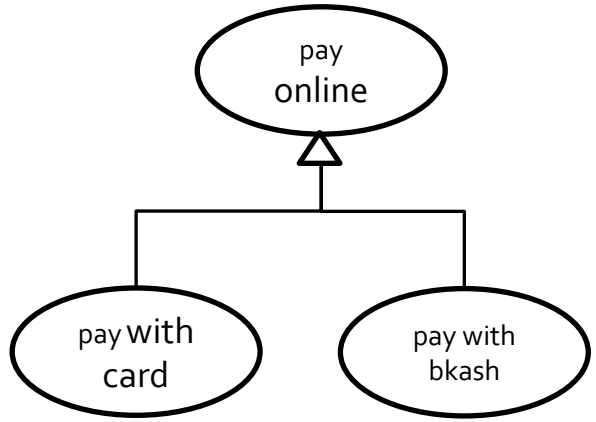


Figure: Generalization relation between use cases

Dependency

- Indicates the dependency relationship between two use cases.
- Two types of dependencies : Include & Extend

- **Include relationships**
  - One use case (base) includes the functionality of another (inclusion case)
  - Supports re-use of functionality

- **Extend relationships**
  - One use case (extension) extends the behavior of another (base)

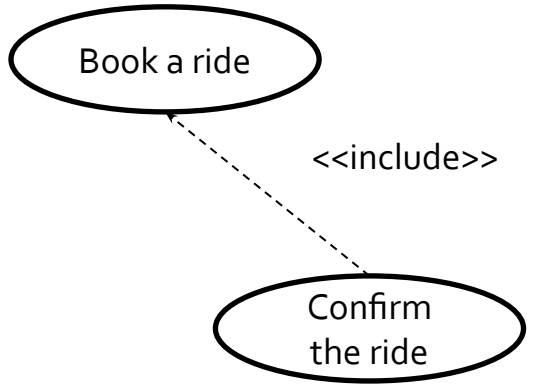


Figure: Include relation

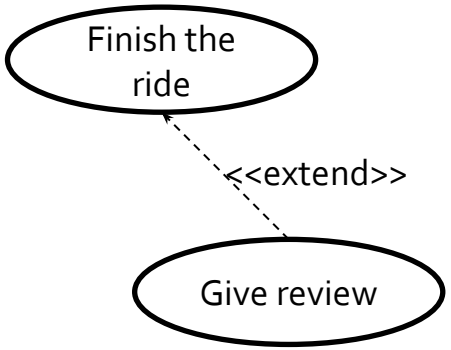


Figure: extend relation

Dependency

Arrow Position

- The arrow should be placed with the use case which execute first.
  - In first example, You need to book a ride first then you can confirm. So the arrow is with book a ride.
  - in second example, You need to finish the ride first then you can give review or not. So arrow is with Finish the ride.

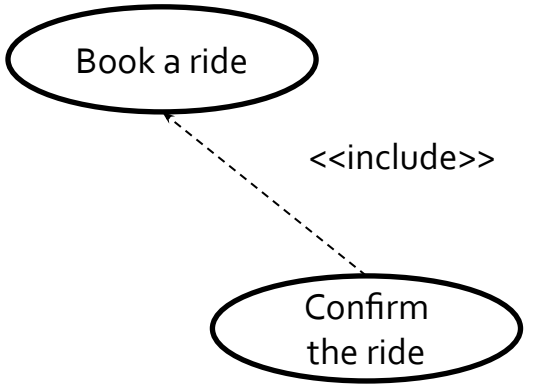


Figure: Include relation

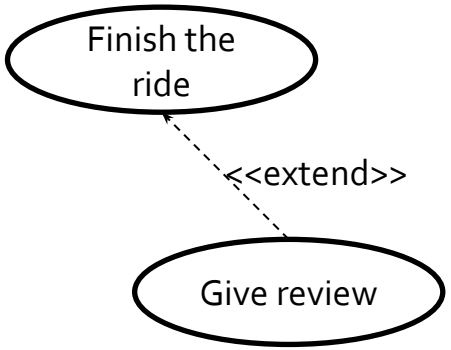


Figure: extend relation

Use case Description

- Every use case diagram must have its description
- Usually description is presented in tabular form
- The diagram should have a unique id
- Typically the description form include the fields –

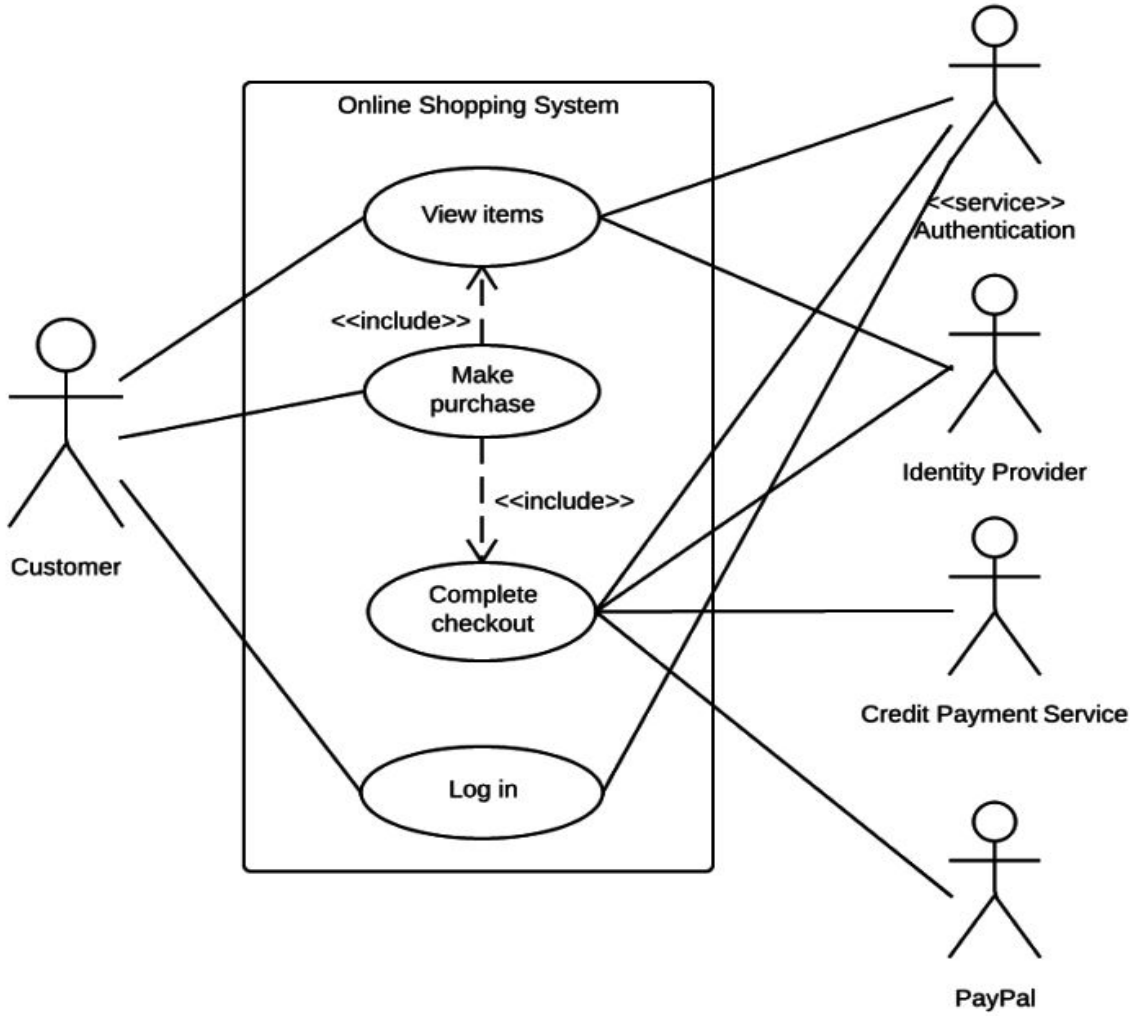
  Use case Name, Id, Actor(s), Description, Precondition, Postcondition, Action Flow, Exceptions, etc.

- All the fields might not

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.		
Trigger: Customer decides to make an offer on a vehicle. Type: ☑ External ☐ Temporal		
Preconditions: Salesperson is authenticated Pending offers datastore is available and on-line Vehicle inventory datastore is available and on-line		

Figure: An Use case description table

Use case example 1 : Purchasing an item



How to Create a Use Case Diagram

BRAC UNIVERSITY
Inspiring Excellence

1. **Identifying Actors**
2. **Identifying Use Cases**
3. **Look for Common Functionality to use Include**
4. **Is it Possible to Generalize Actors and Use Cases**
5. **Optional Functions or Additional Functions**

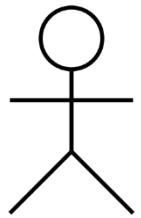
How to Create a Use Case Diagram

Identifying Actors

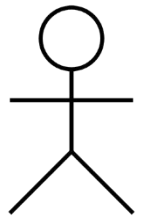
Customer

Bank employee

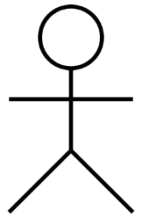
NFRC Customer



customer



NFRC customer



Bank Employee

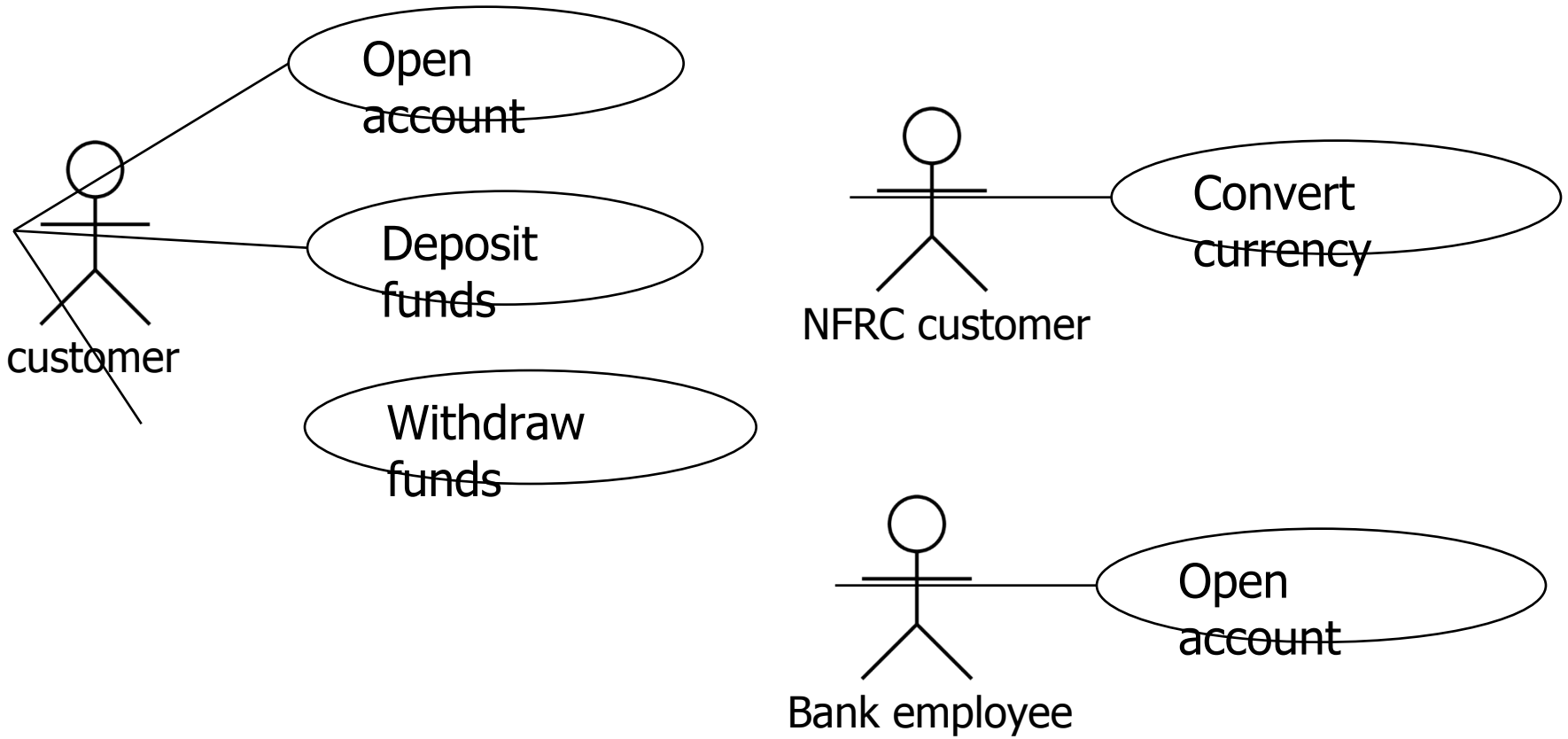
How to Create a Use Case Diagram

BRAC UNIVERSITY
Inspiring Excellence

Identifying Use Cases

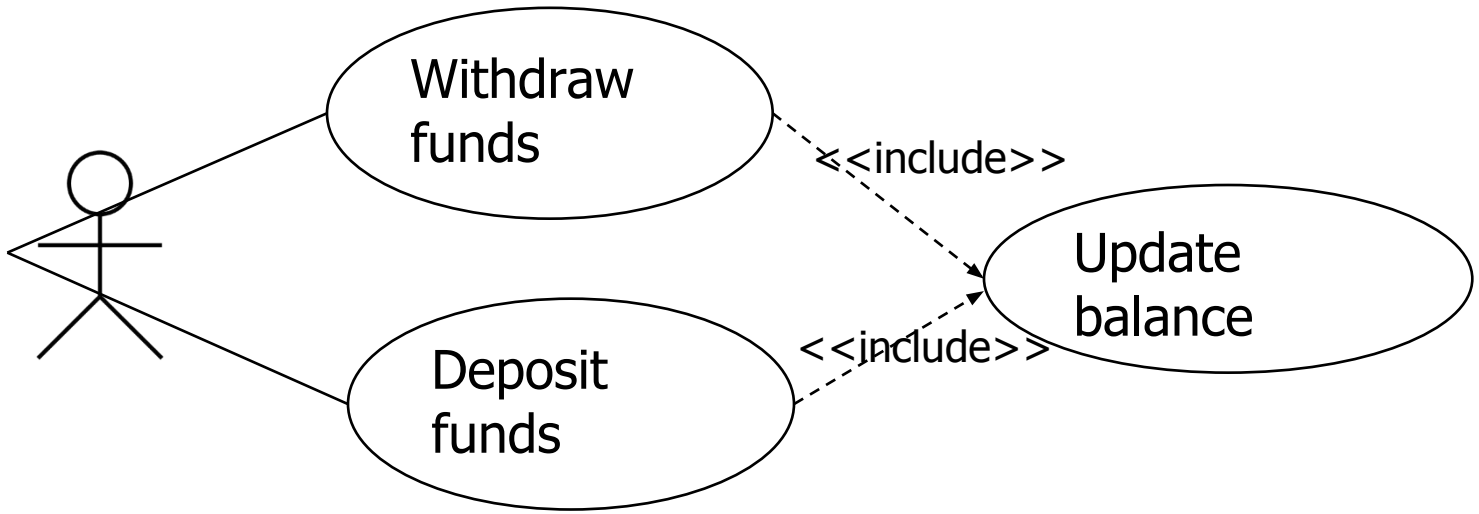
- A good way to do this is to identify what the actors need from the system
- A customer will need to
  1. open accounts
  2. Deposit funds
  3. withdraw funds
  4. request check books

How to Create a Use Case Diagram



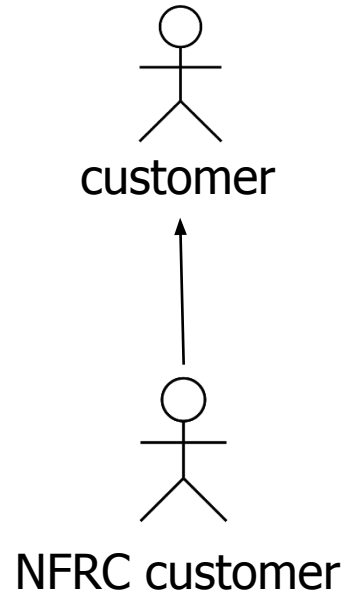
How to Create a Use Case Diagram

- **Look for Common Functionality to use Include**
  - find two or more use cases that share common functionality



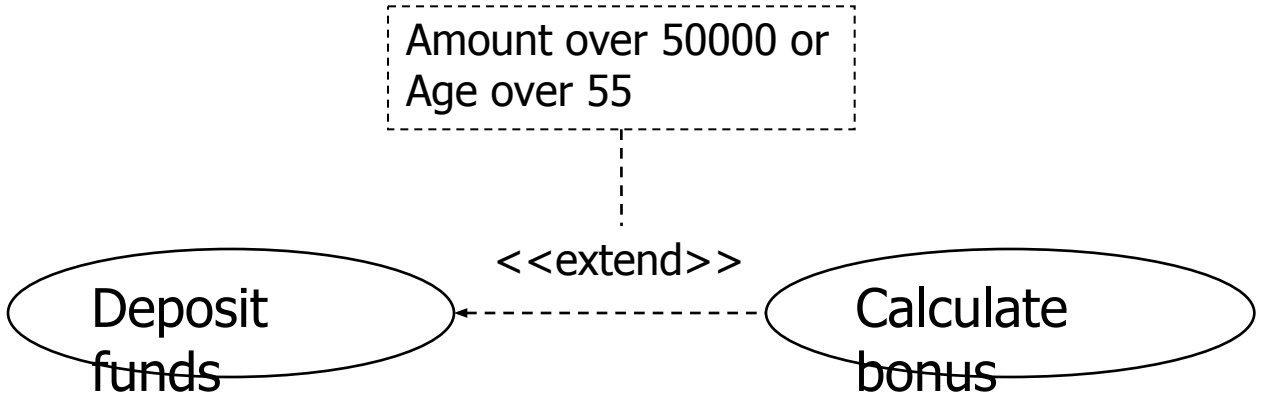
How to Create a Use Case Diagram

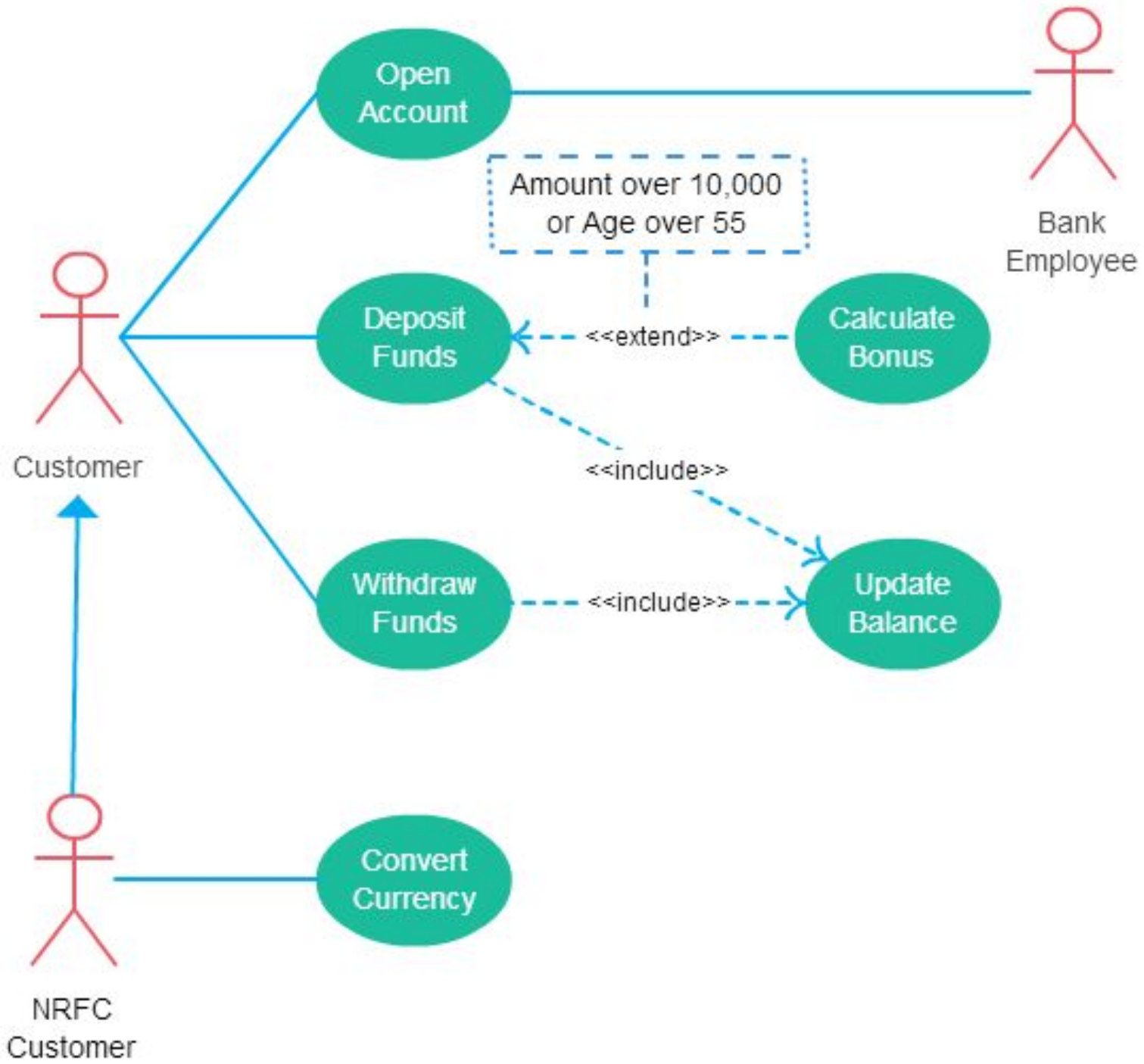
- **Is it Possible to Generalize Actors ?**



How to Create a Use Case Diagram

- **Optional Functions or Additional Functions**
  - There are some functions that are triggered optionally

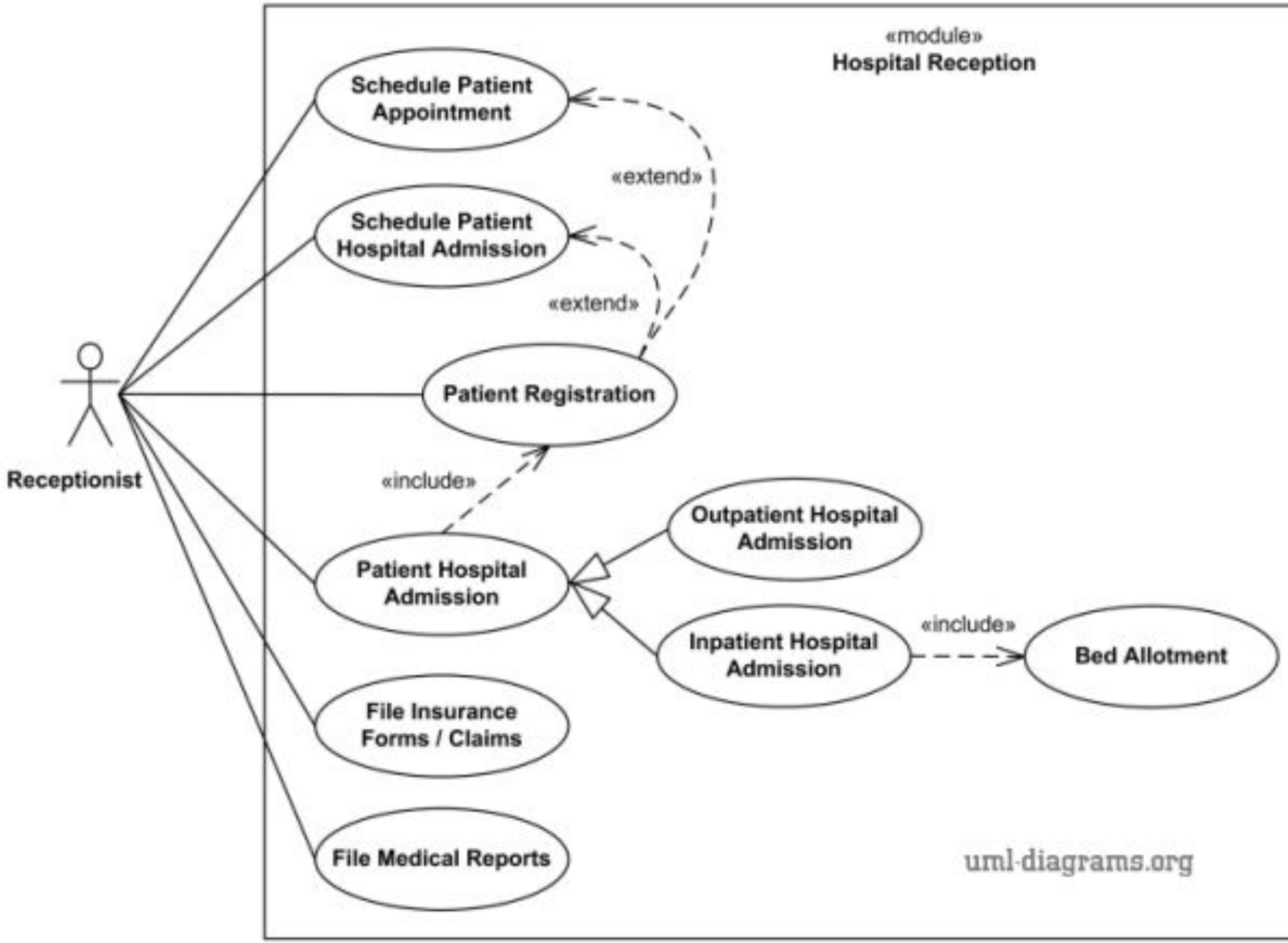




BRAC UNIVERSITY
Inspiring Excellence

Practice

- Hospital Management System is a large system including several subsystems or modules providing variety of functions. UML use case diagram example below shows actor and use cases for a hospital's reception.
- Purpose: Describe major services (functionality) provided by a hospital's reception.
- Hospital Reception subsystem or module supports some of the many job duties of hospital receptionist. Receptionist schedules patient's appointments and admission to the hospital, collects information from patient upon patient's arrival and/or by phone. For the patient that will stay in the hospital ("inpatient") she or he should have a bed allotted in a ward. Receptionists might also receive patient's payments, record them in a database and provide receipts, file insurance claims and medical reports.



BRAC UNIVERSITY
spiring Excellence

- Bank customer has to provide pin for login, verified by bank, there could be mistake while entering the pin, customer can do transactions like (fund transfer, withdraw, change pin, balance check, deposit) etc., ATM Machine will provide a print out if customer wants to get a receipt of their transaction considering some charges, charges will also apply for each transaction. There is system administrator who monitors users' transaction and report if any suspicious activities is noticed. Also maintains ATM machine money loading and maintenance.

BRAC UNIVERSITY

Inspiring Excellence

UML Activity Diagrams

OUTLINE

- Introduction
- Activity Diagrams - notation
- How to apply activity diagrams
- Guidelines
- Examples

BRAC UNIVERSITY
Inspiring Excellence

What is an Activity Diagram?

BRAC UNIVERSITY
Inspiring Excellence

- An Activity Diagram is one of the Behavior diagrams.
- Activity modelling is the sequence and conditions for coordinating lower-level behaviors, rather than which classifiers own those behaviors.
- These are commonly called control flow and object flow models.
- The behaviors coordinated by these models can be initiated because other behaviors finish executing, because objects and data become available, or because events occur external to the flow.
- A UML Activity Diagram shows sequential and parallel activities in a process.
- Useful for modelling:
  - Business processes
  - Workflows
  - Data flows
  - Complex algorithms

Initial and Final Nodes

- Initial Node:
  - An initial node is a control node at which flow starts when the activity is invoked.
  - An activity may have more than one initial node.

- Final Node:
  - An activity may have more than one activity final node;

    the first one reached stops all flows in the activity.

Action

- Action:
  - An action represents a single step within an activity that is not further decomposed within the activity.
  - An activity represents a behavior that is composed of individual elements that are actions.
  - An action is simple from the point of view of the activity containing it, but may be complex in its effect and not be atomic.
  - An activity can be reused in many places, whereas an instance of an action is only used once at a particular
  - An action will not begin execution until all of its input conditions are satisfied.

Merge and Decision Nodes

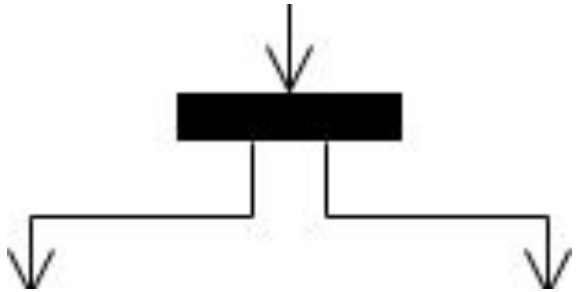
- Merge Node:
  - A merge node is a control node that brings together multiple alternate flows.
  - It is not used to synchronize concurrent flows but to accept one among several alternate flows.
  - A merge node has multiple incoming edges and a single outgoing edge.

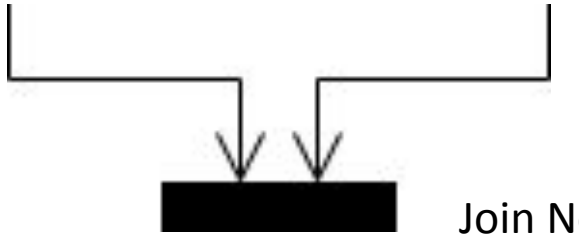
- Decision Node:
  - A decision node accepts tokens on an incoming edge and presents them to multiple outgoing edges.
  - Which of the edges is actually traversed depends on the evaluation of the guards on the outgoing edges.

Join and Fork Nodes

- Join Node:
  - A join node is a control node that synchronizes multiple flows.
  - A join node has multiple incoming edges and one outgoing edge.
- Fork Node:
  - A fork node is a control node that splits a flow into multiple concurrent flows.
  - A fork node has one incoming edge and multiple outgoing

Fork Node





Join Node

Object Node

- Object Node:
  - An object node is an activity node that indicates an instance of a particular classifier, possibly in a particular state, may be available at a particular point in the activity.
  - Object nodes can be used in a variety of ways, depending on where objects are flowing from and to.

NOTE

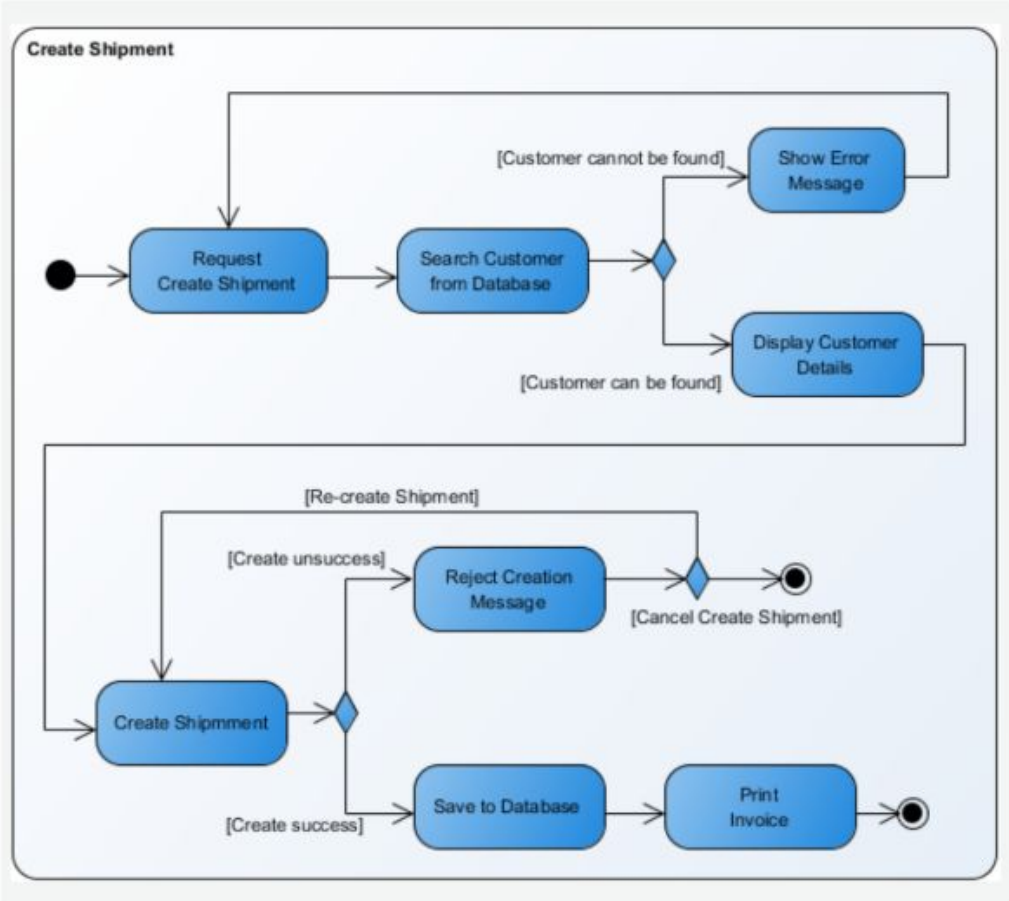
- Note:
  - A note (comment) gives the ability to attach various remarks to elements.
  - A comment carries no semantic force, but may contain information that is useful to a modeler.

Business Process Modelling

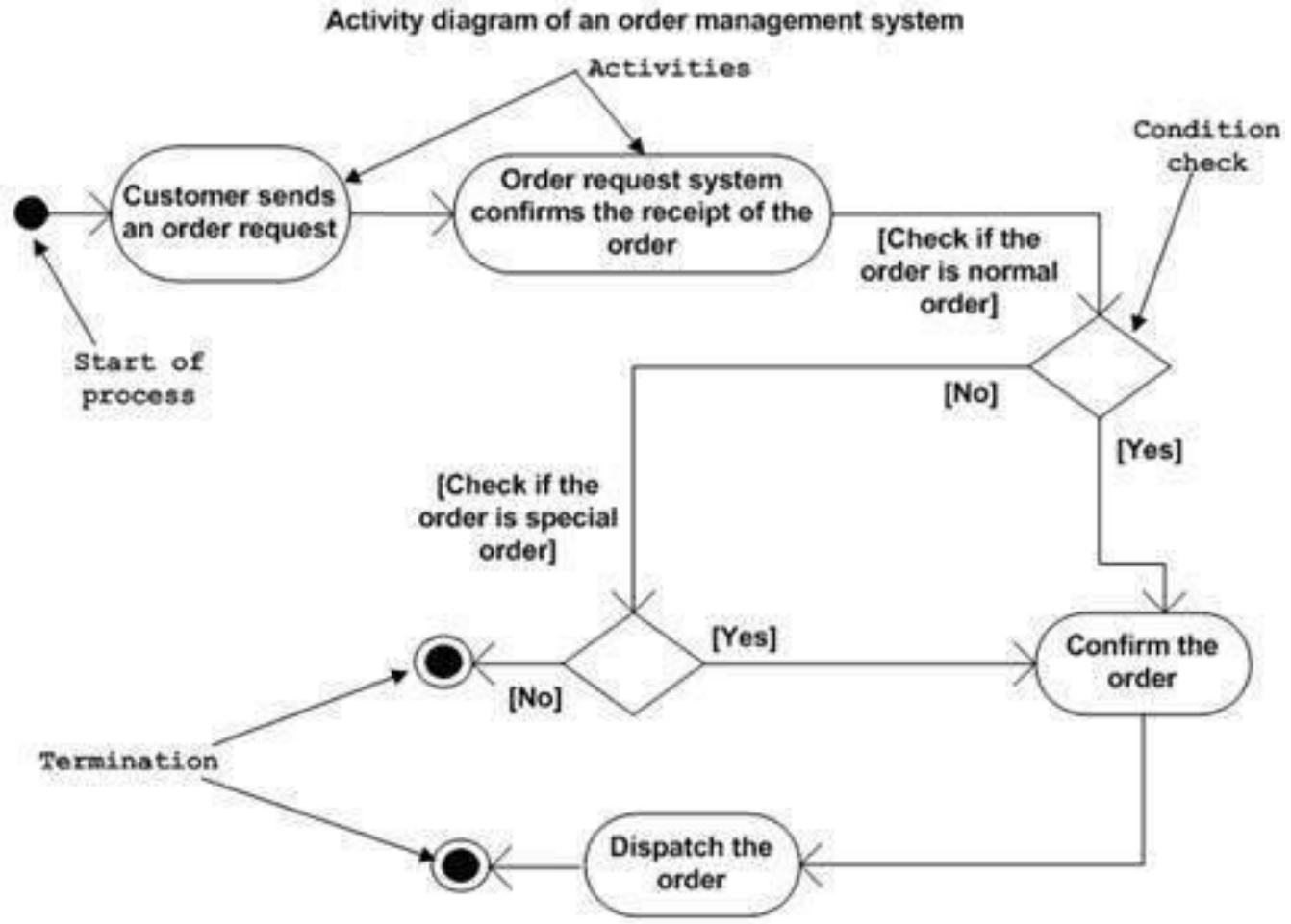
BRAC UNIVERSITY
Inspiring Excellence

- Example: Parcel shipping
  - The process of shipping a parcel is non-trivial; there are many parties involved (customer, driver,. . .) and many steps.
  - The process can be captured by a Use Case diagram, but activity diagrams are great example of "a picture being worth a thousand words".
  - Object nodes are useful for illustrating what is moving around.

Activity Diagram Parcel shipping



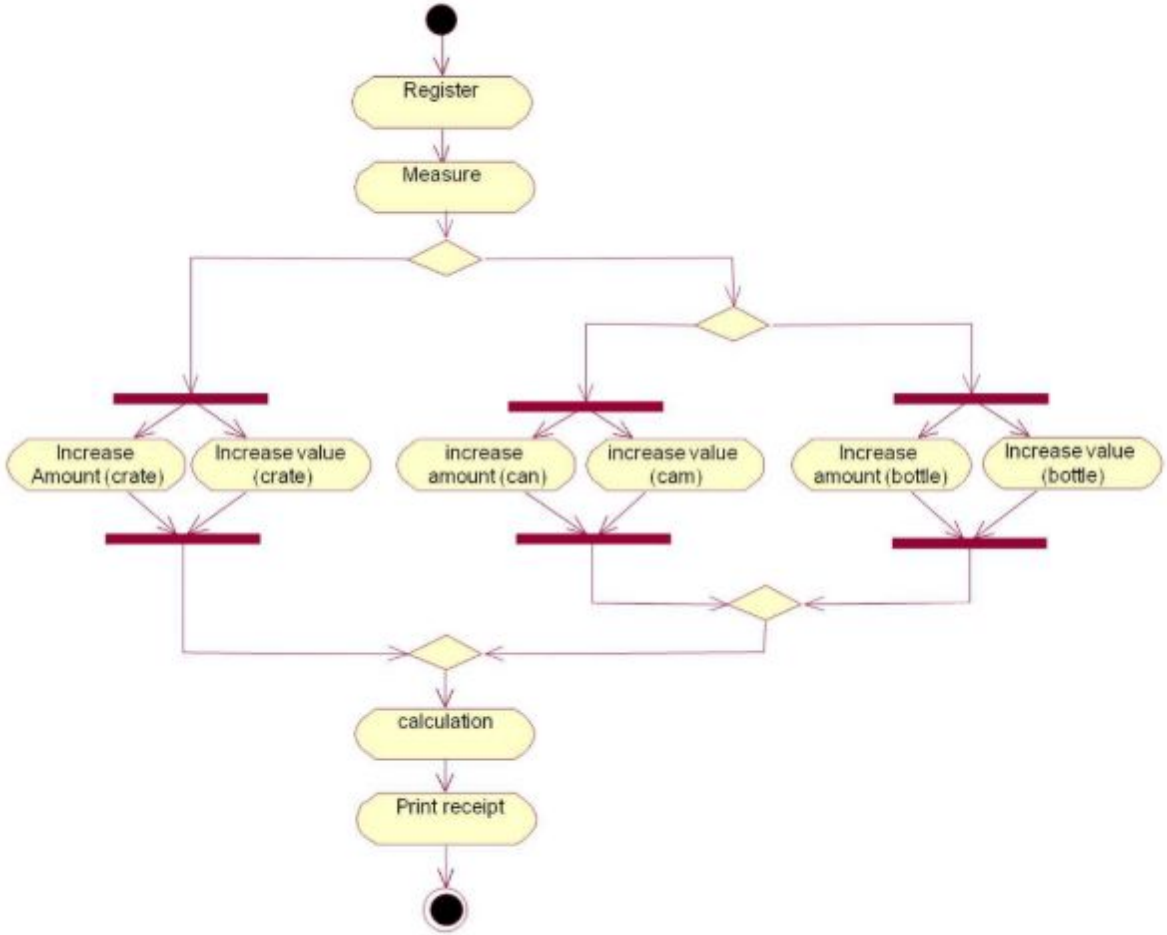
Activity Diagram of order management system



Guideline for Activity Modelling

- The technique proves most valuable for very complex processes, usually involving many parties.
- On a first overview "level 0" diagram, keep all the actions at a very high level of abstraction, so that the diagram is short. Then expand details in sub-diagrams at the "level 1" level,. . . etc.
- Try to make the level of abstraction of action nodes roughly equal within a diagram (Very different levels of abstraction might be a node labelled "Deliver Order" and a node labelled "Calculate Tax").

More Examples: Recycling Activity Diagrams



Swimlanes

- Swimlanes (or activity partitions) indicate where activities take place.
- Swimlanes can also be used to identify areas at the technology level where activities are carried out
- Swimlanes allow the partition an activity diagram so that parts of it appear in the swimlane relevant to that element in the partition

Swimlanes

- Partitions may be constructed on the basis of:
  - the class and actor doing the activity
  - Partitioning by class and actor can help to identify new associations that have not been documented in the class model
  - the use case the activity belongs to
  - Partitioning by use cases can help document how use cases interact

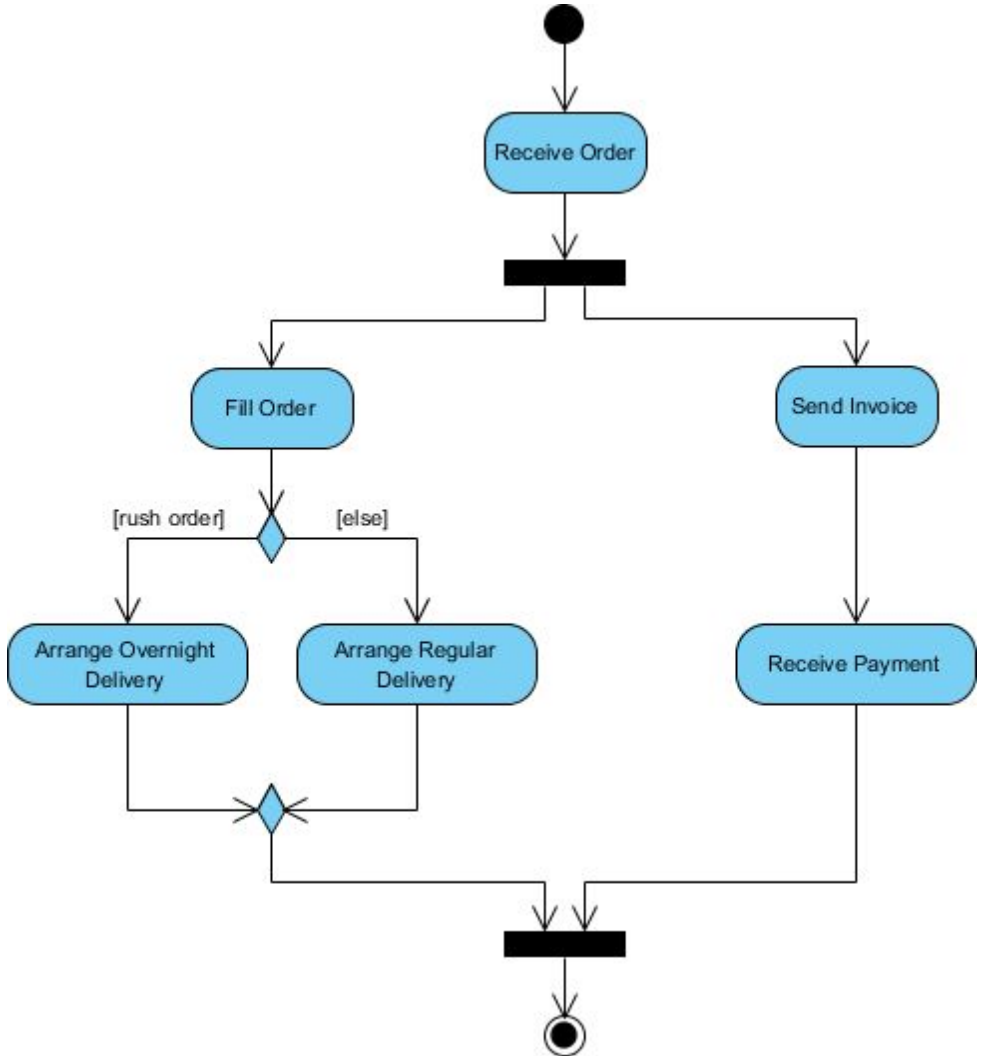
Example Activity

Process Order - Problem Description

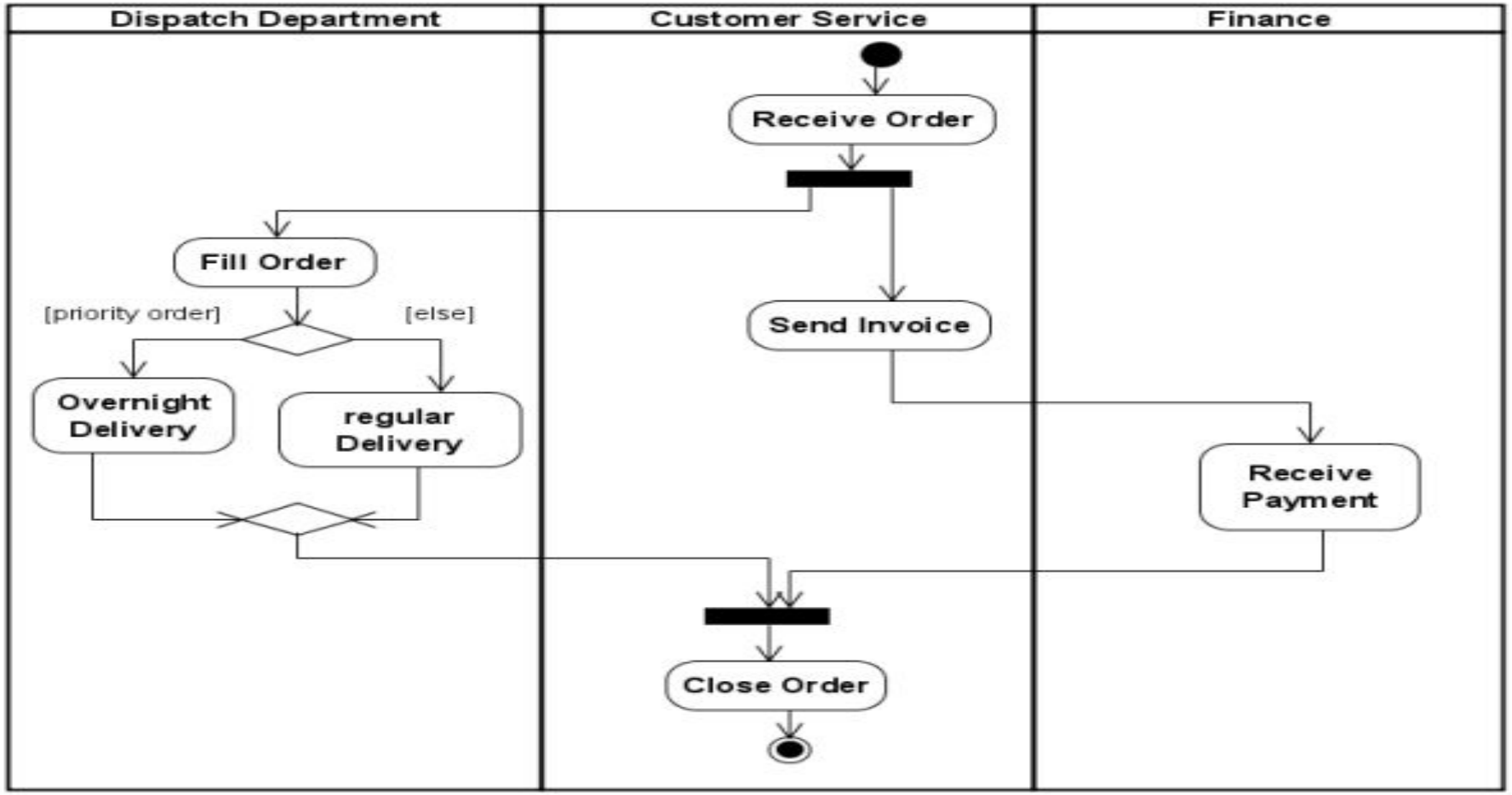
Once the order is received, the activities split into two parallel sets of activities. One side fills and sends the order while the other handles the billing.

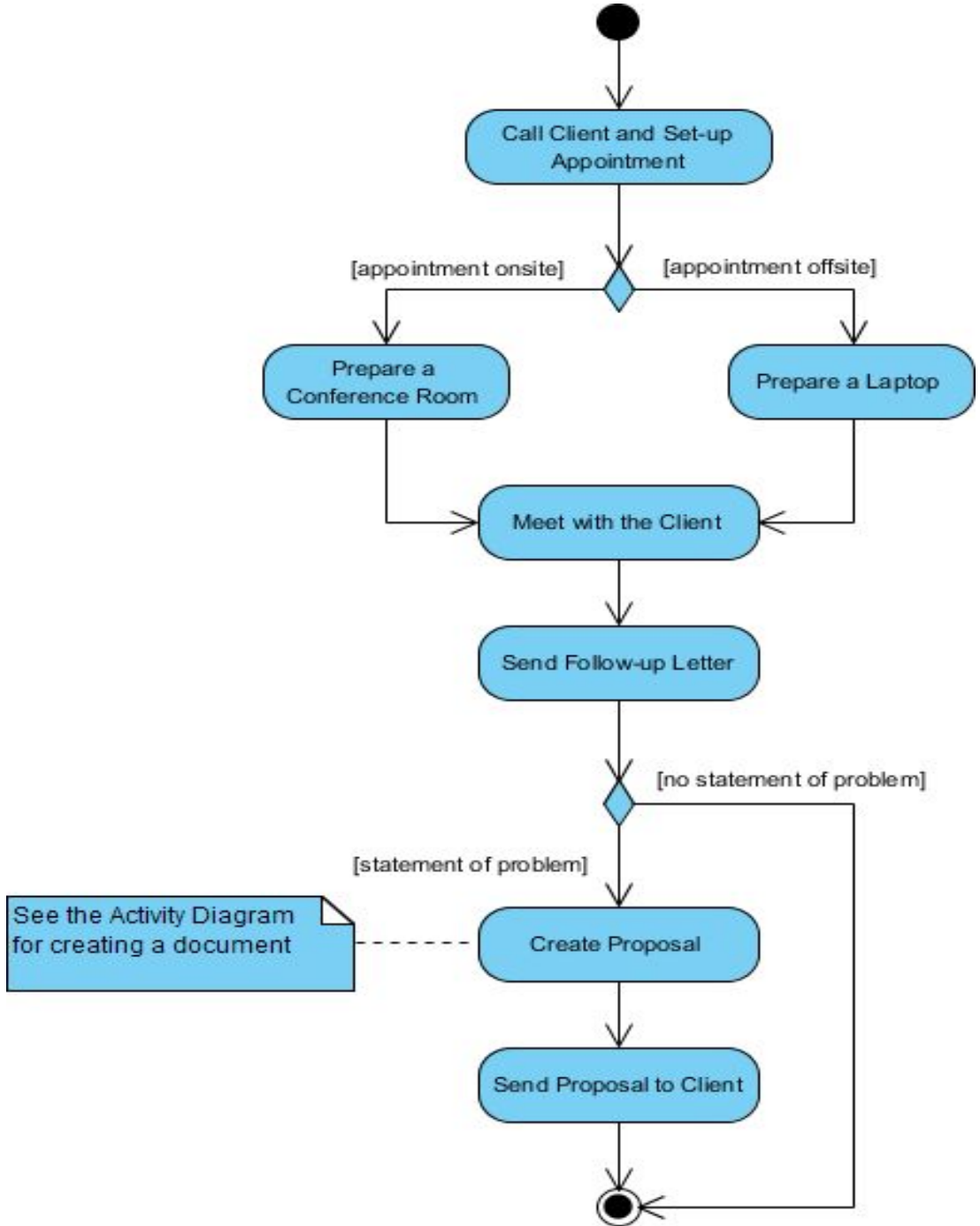
On the Fill Order side, the method of delivery is decided conditionally. Depending on the condition either the Overnight Delivery activity or the Regular Delivery activity is performed.

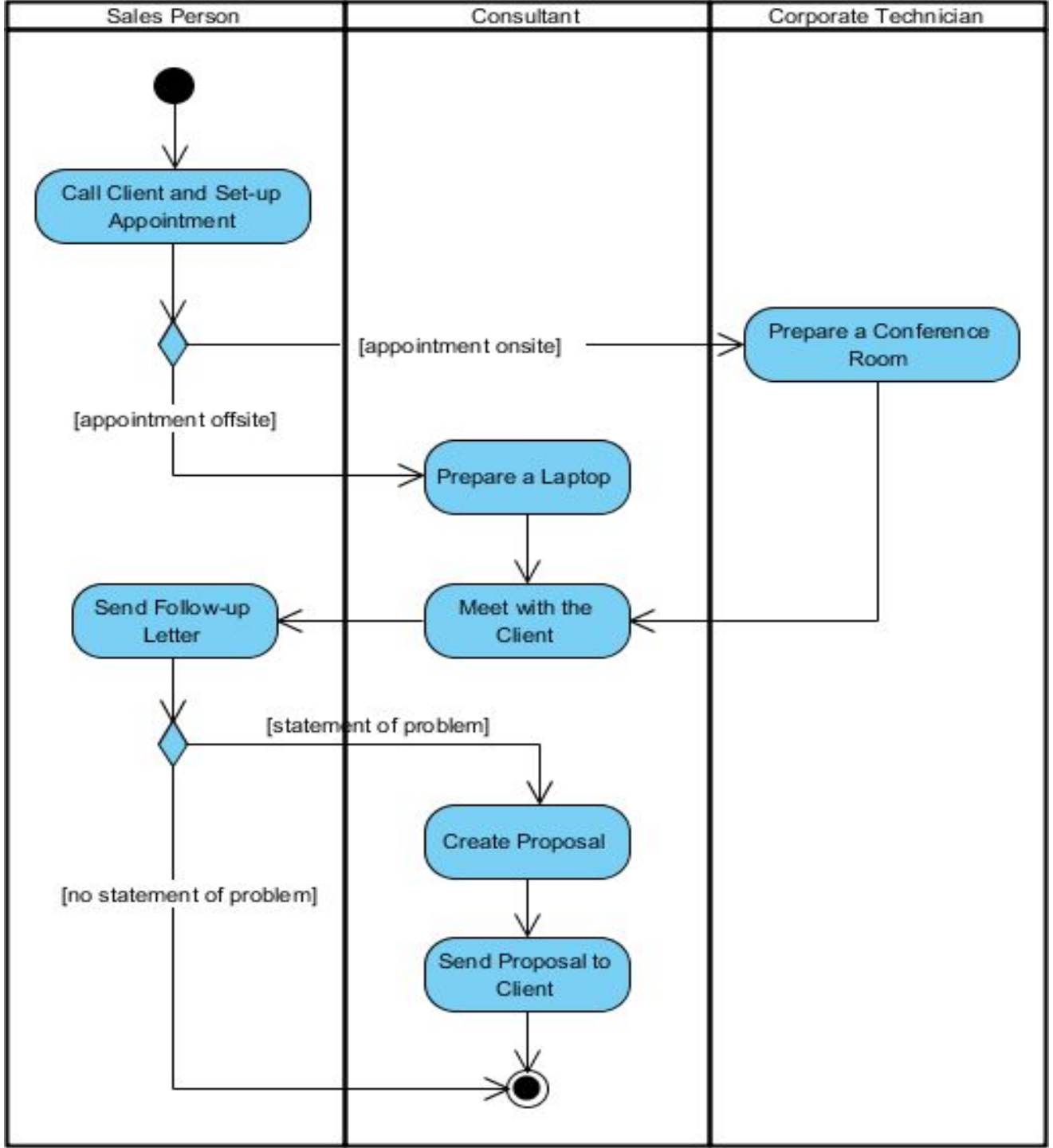
Finally the parallel activities combine to close the order.



Example Swimlanes



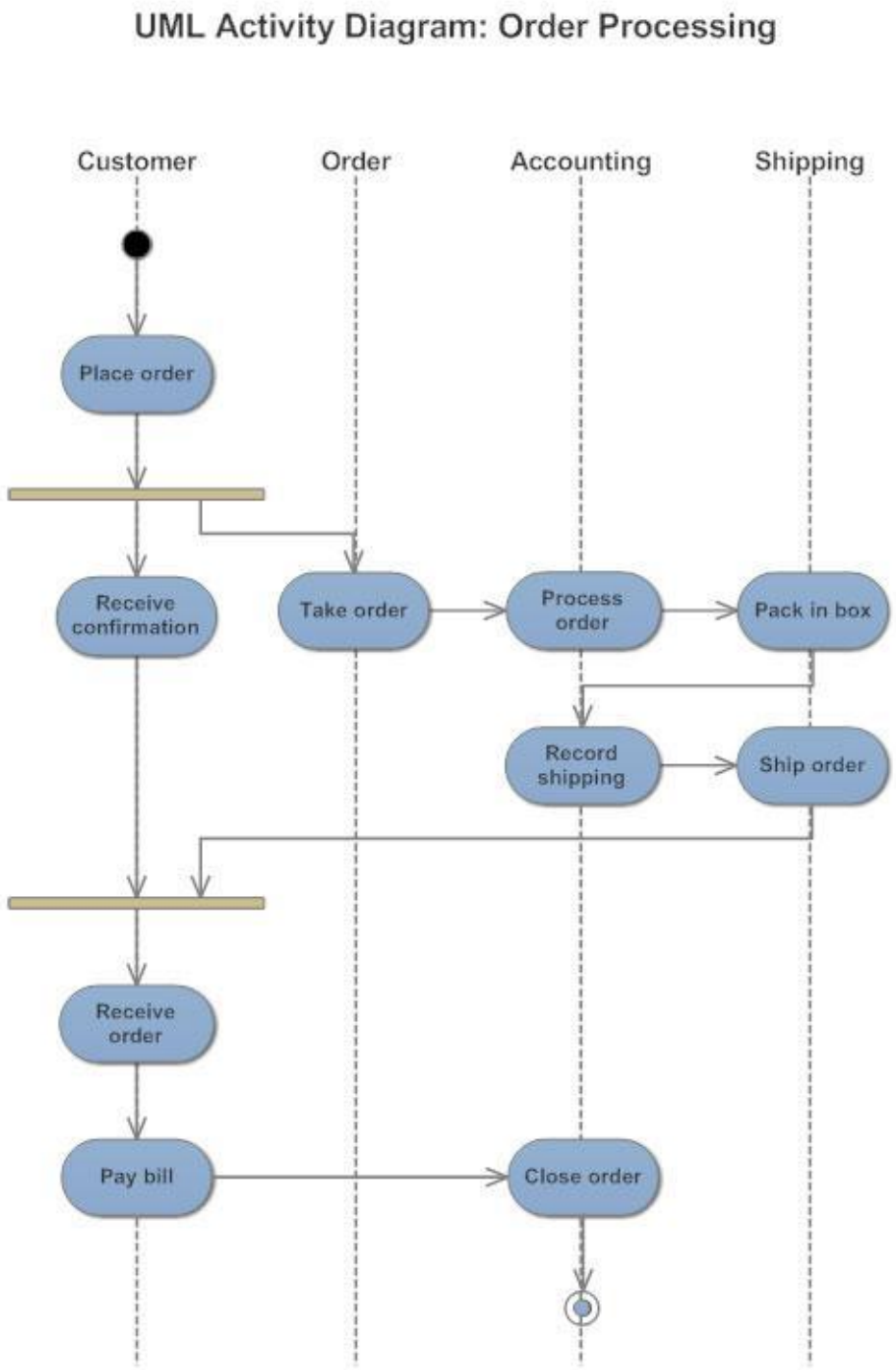

Call Client and Set-up Appointment
[appointment onsite]
[appointment offsite]
Prepare a Conference Room
Prepare a Laptop
Meet with the Client
Send Follow-up Letter
[no statement of problem]
[statement of problem]
See the Activity Diagram for creating a document
Create Proposal
Send Proposal to Client



BRAC UNIVERSITY

Inspiring Excellence

BRAC UNIVERSITY
Inspiring Excellence



Class work

- ATM system of BRAC bank
  - Customer
  - ATM
  - Bank server

BRAC UNIVERSITY
Inspiring Excellence