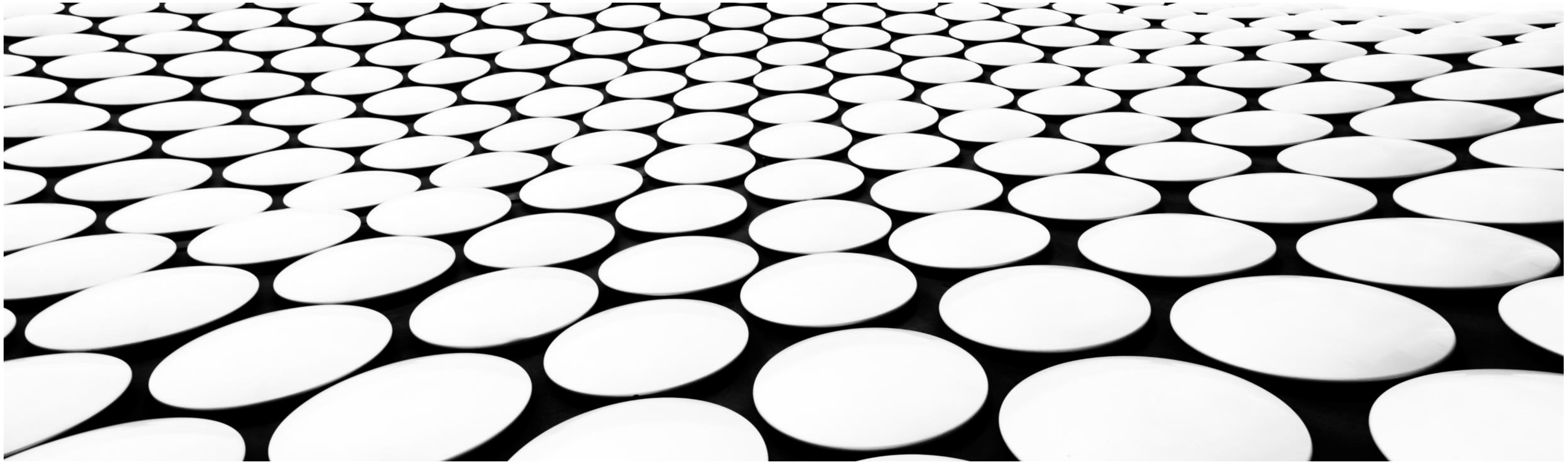
MODELING UML USE CASE DIAGRAMS AND CAPTURING USE CASE SCENARIOS

YESODA BHARGAVA



INTRODUCTION

- Use case diagram is a platform that can provide a common understanding for the end users, developers and domain experts.
- It is used to capture the basic functionality i.e. use cases, and the users of those available functionality, i.e. actors, from a given problem statement.
- In this experiment, we will learn how use cases and actors can be captured and how different use cases are related in a system.

LAB OBJECTIVE

- How to identify different actors and use cases from a given problem statement.
- How to associate use cases with different types of relationships.
- How to draw a use-case diagram.

WHAT IS A USE-CASE?

- A usage scenario for a piece of software.
- A potential scenario in which a system receives an external request (such as user input) and responds to it. [Wikipedia]
- A list of actions or event steps, defining the interactions between a role and a system, to achieve a goal.
- The actor can be a human, an external system, or time.
- Use case describes a way in which a real world actor interacts with the system.
- Basically how actors interact with the system in order to solve a problem.
- The method was originally formulated in 1986 for textual and visual modelling in software engineering.
- System functionalities written in an organized manner.

Use case diagrams aim to present a **graphical overview of the functionality provided by the system**. It consists of set of actions (referred to as use cases) that the concerned system can perform, one or more actors, and dependencies among them.

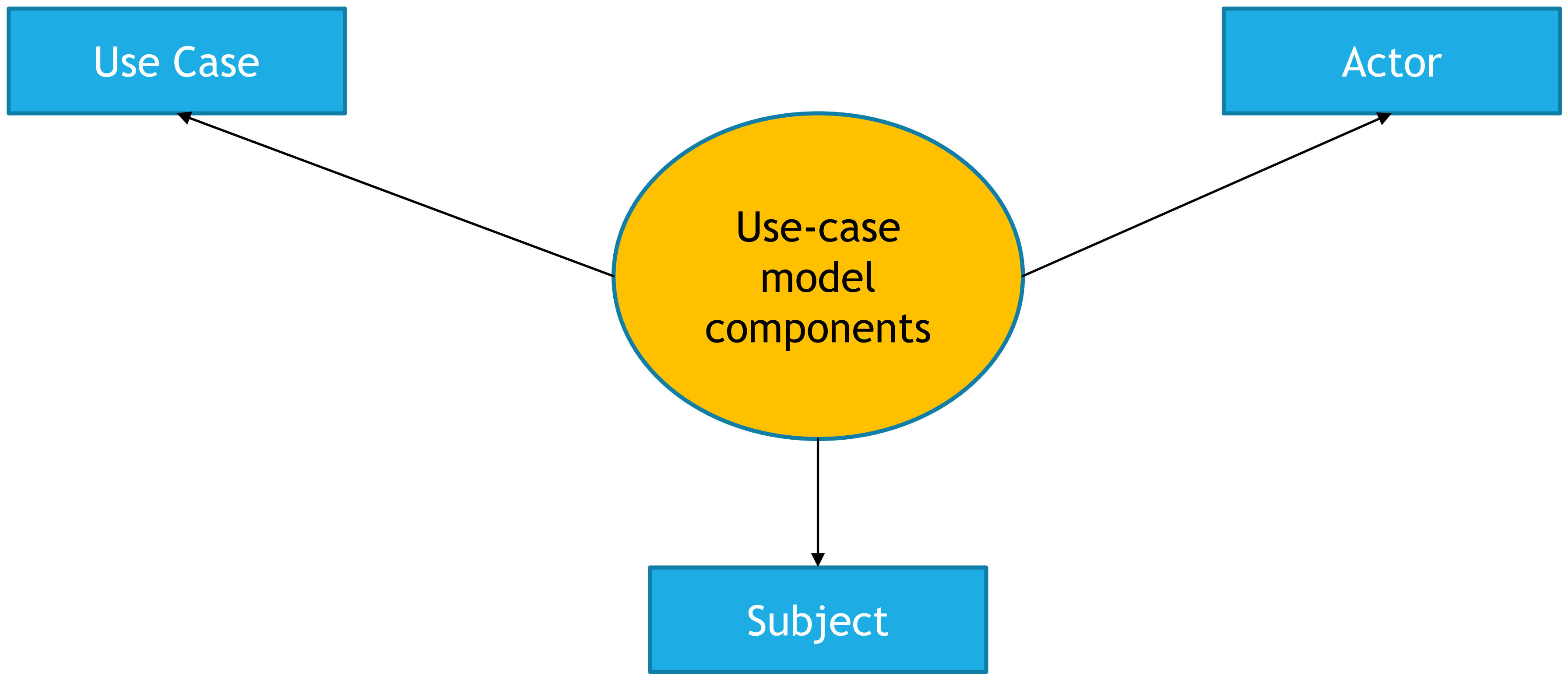
WHY BOTHER AT ALL?

- Use-cases provides the shortest summary of what the system will offer.
- Captures the dynamic aspect of a system.
- Overview of the role of each and every component in the system.
- Provides solutions and answers to many questions that might pop up if a project is started unplanned.
- Makes the communication about the system simpler.
- Helps in designing of the system and assists in iteration planning.

WHERE TO USE A USE-CASE DIAGRAM

- Use case diagrams help to gather system requirements and actors.
- Specify the events of a system and their flows.
- Use case diagrams are used for:
  - Requirement analysis and high level design
  - Model the context of a system

COMPONENTS OF BASIC USE-CASE MODEL



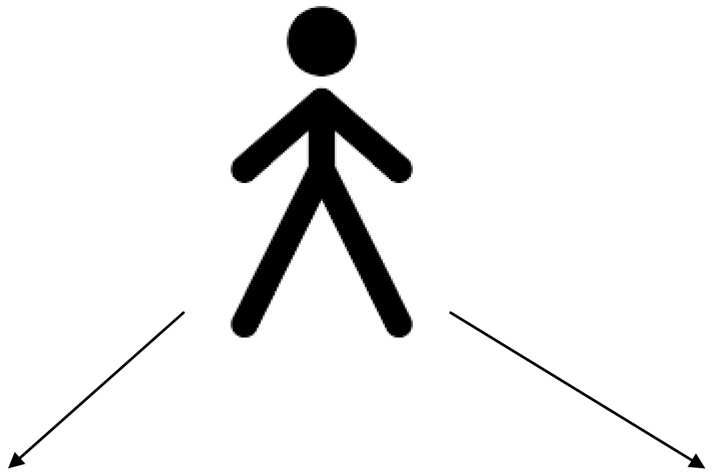
ACTOR

- An object or a set of objects, external to the system, which interacts with the system to get some meaningful work done.
- Actors could be humans, devices, or even other external systems.

ATM Machine

Who is the actor here?

Human



<u>Primary Actor</u>

The principal users of the system, who fulfil
Their goal by availing some service from the system.
E.g. In the ATM machine example, the human is the Primary actor here.

<u>Supporting Actor</u>

Render some kind of service to the system. E.g. Bank representatives who replenish the stock of cash.

LET US THINK OF ATM USE CASES

- Withdraw cash.
- Change PIN
- Check balance

A use case is simply a functionality provided by the system.

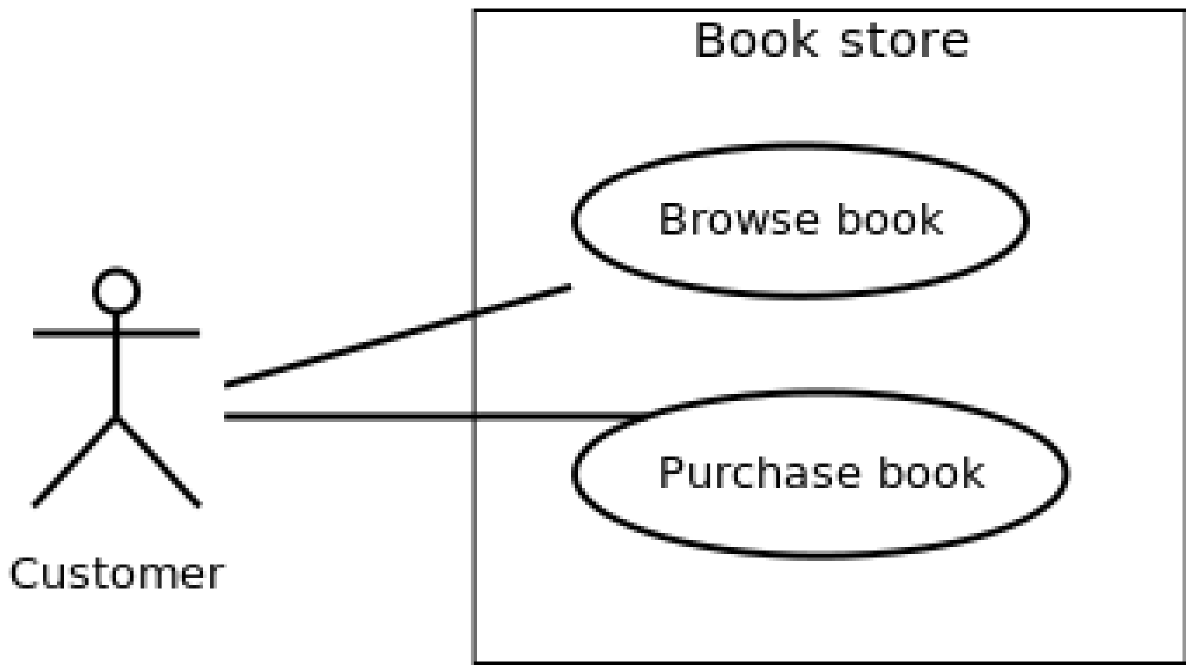
ATM
password
****
ATM
error
****

Use cases include both successful and unsuccessful scenarios of user interactions with the system.

SUBJECT

- Subject is the system under consideration.
- Use cases apply to a subject.
- ATM is a subject, having multiple use-cases and multiple actors to interact with.

GRAPHICAL REPRESENTATION



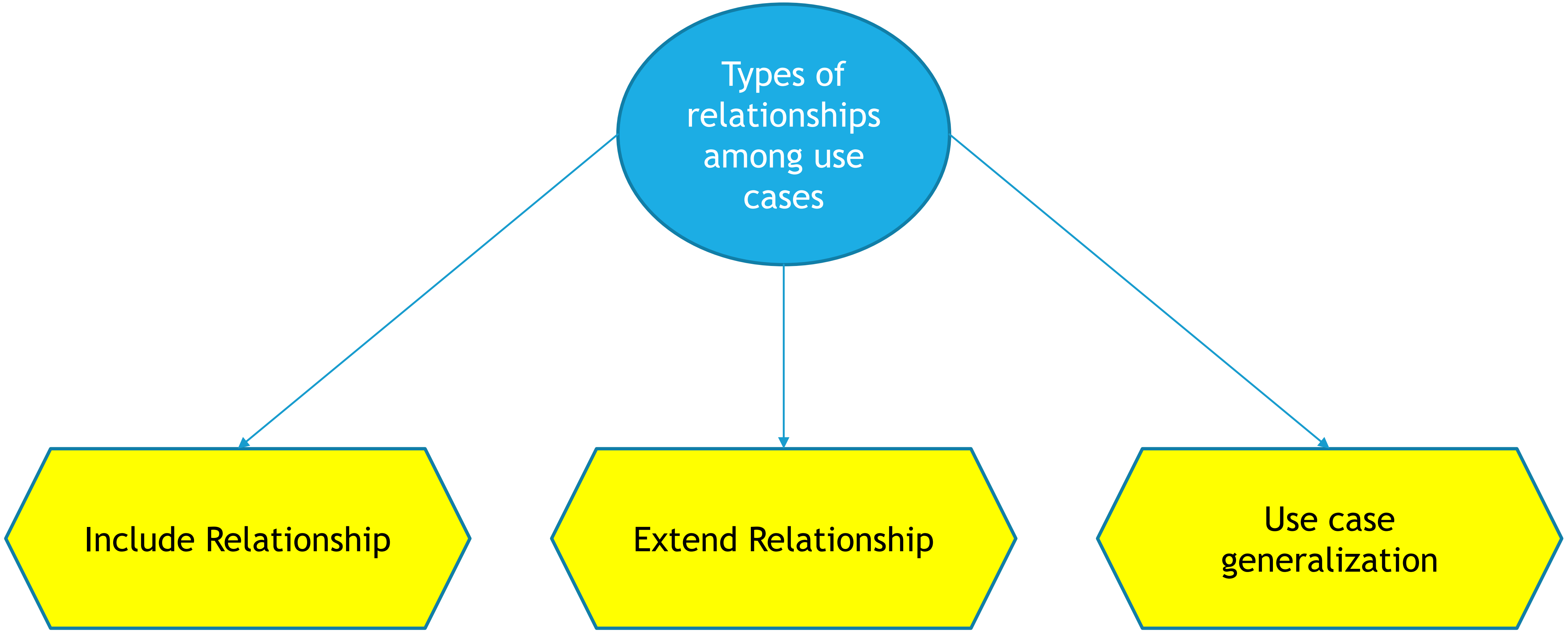
A use-case diagram for a book store.

- An actor is represented by a stick figure, and the name written below it.
- Use case depicted by an ellipse and name of the use-case written inside it.
- Subject is shown by drawing a rectangle.

ASSOCIATION BETWEEN ACTORS AND USE CASES

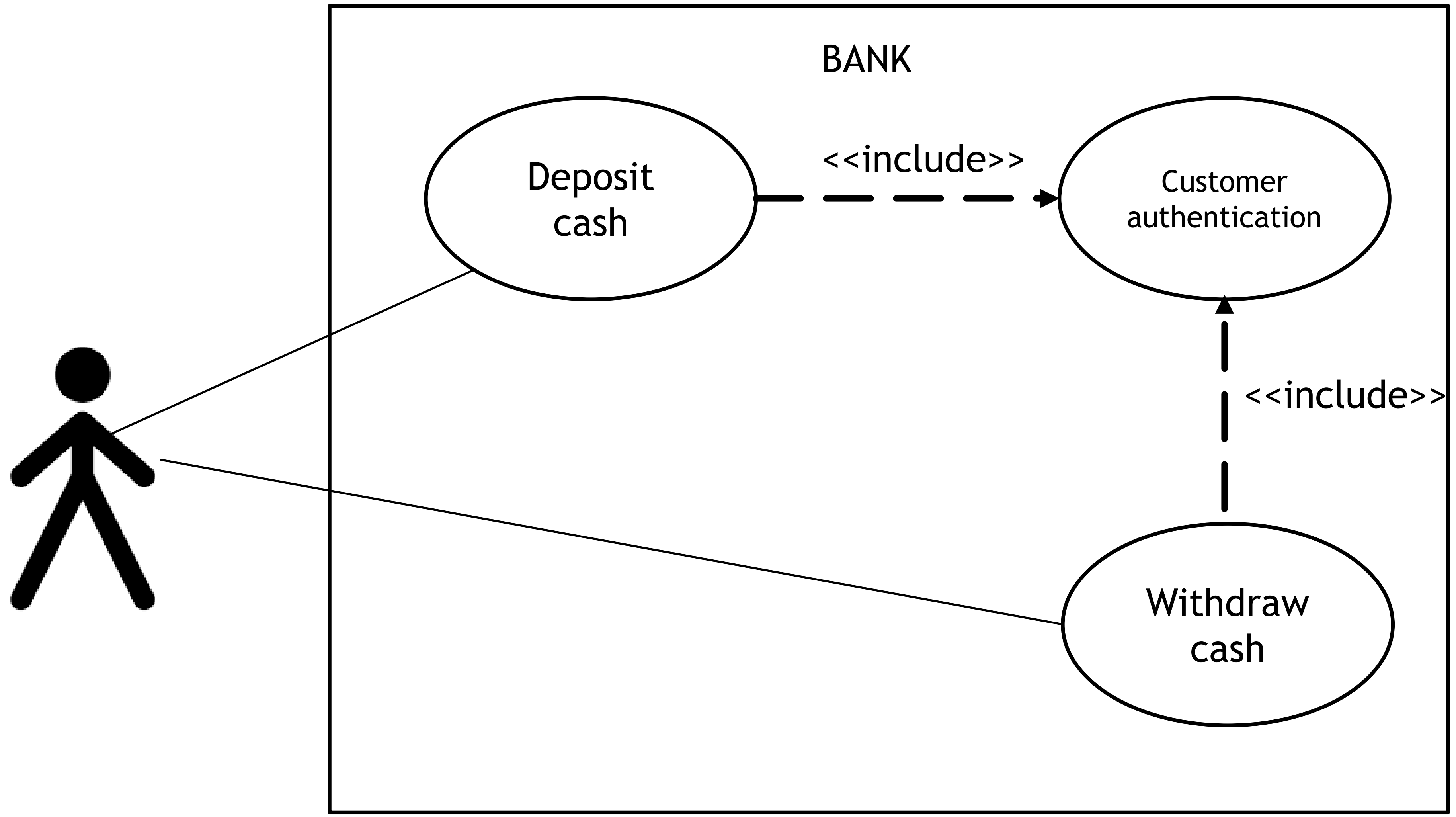
- A use-case is triggered by an actor.
- Actors and use cases connected through associations indicating that the two communicate through message passing.
- An actor must be associated with at least one use case.
- Similarly, a given use case must be associated with at least one actor.

USE CASE RELATIONSHIPS



INCLUDE RELATIONSHIP

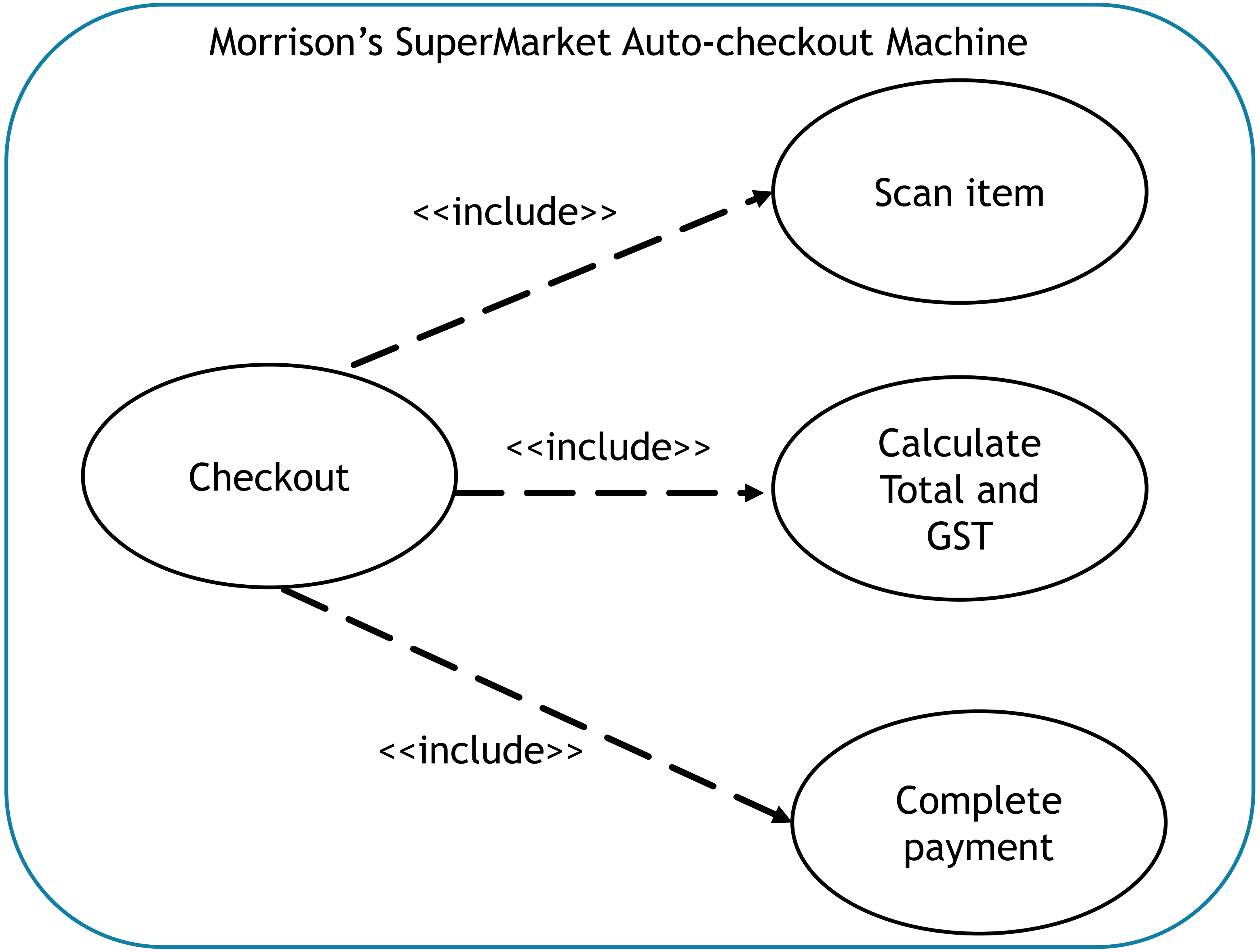
- Used to depict common behaviour that are shared by multiple use cases.
- Can be considered analogous to writing functions in a program in order to avoid repetition of writing the same code.
- Such function can be called from different points within the program.



Include relationship between use case. *Customer authentication* use case which is included by *withdraw cash*, and *deposit cash* use cases. Arrow goes from including use case to the included use case.

MORE ON USE CASE INCLUDE

- Use case include is a **directed relationship** between **two use cases** used to show that behaviour of the included use case (*Customer authentication*) is inserted into the behaviour of the including (the base) use case.



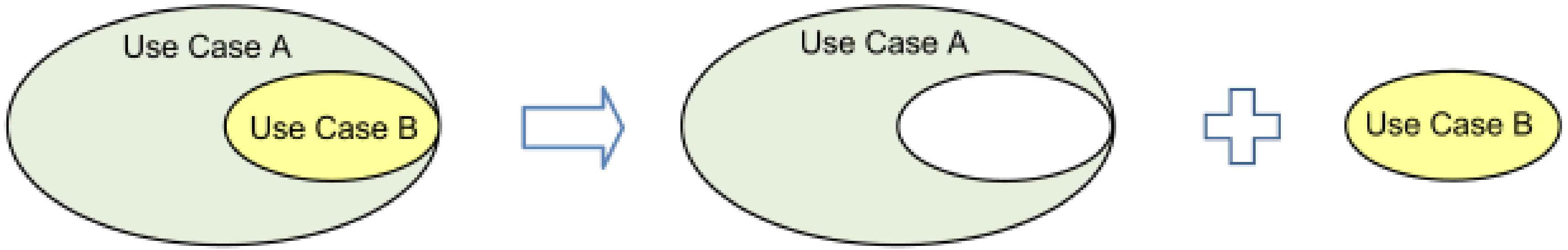
Include relationship between use cases Is shown by a dashed arrow with an open-arrowhead from the including (base) use case to the included (common part) use case.

Included use cases should be complete, for including use case to be complete.

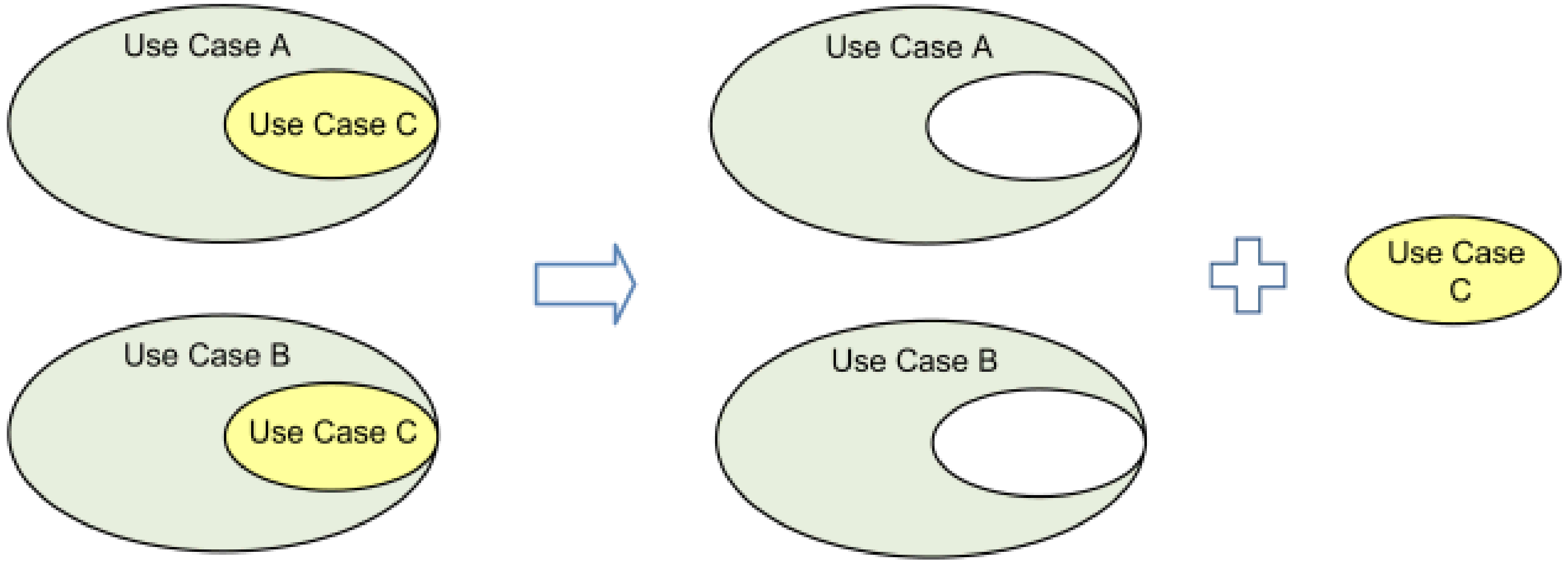
Checkout use case is incomplete Without the other three.

The included use cases are required, not optional.

- A large use case could have some behaviors which might be detached into distinct smaller use cases to be included back into the base use case using the UML **include** relationship. The purpose of this action is modularization of behaviors, making them more manageable.



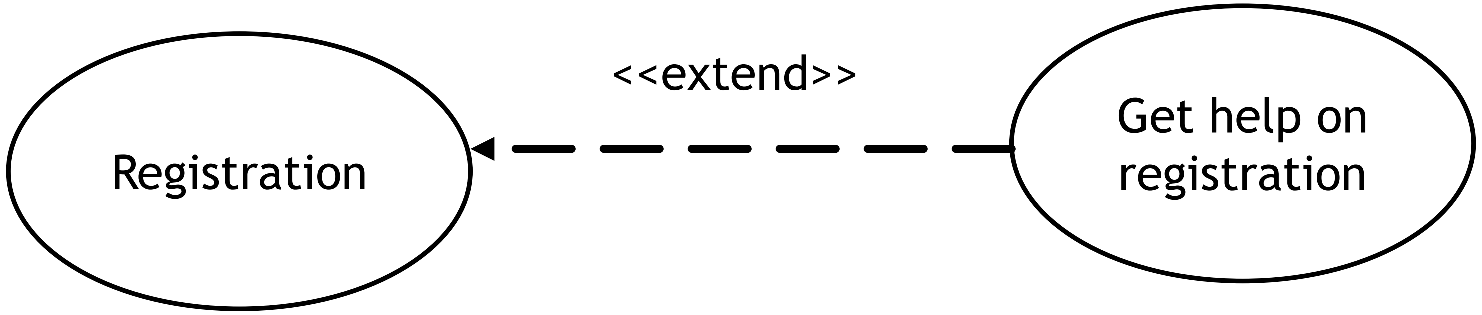
When two or more use cases have some **common behavior**, this common part could be extracted into a separate use case to be included back by the use cases with the UML **include** relationship.



EXTEND RELATIONSHIP

- Use case extensions used to depict any variation to an existing use case.
- The “extends” relationship shows the alternative options under the specific use case.
- The arrow goes from extending use case to the extended use case.

$$\begin{tikzcd}
\text{Extended use case} & \text{Extending use case} \arrow[l, dashed, "\text{<<extend>>}"']
\end{tikzcd}$$

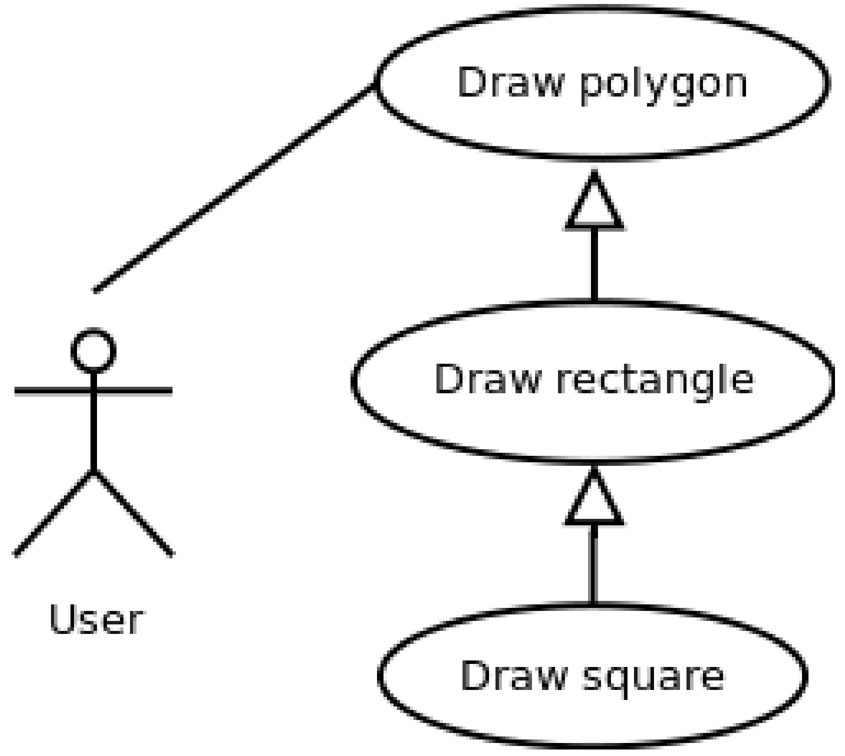


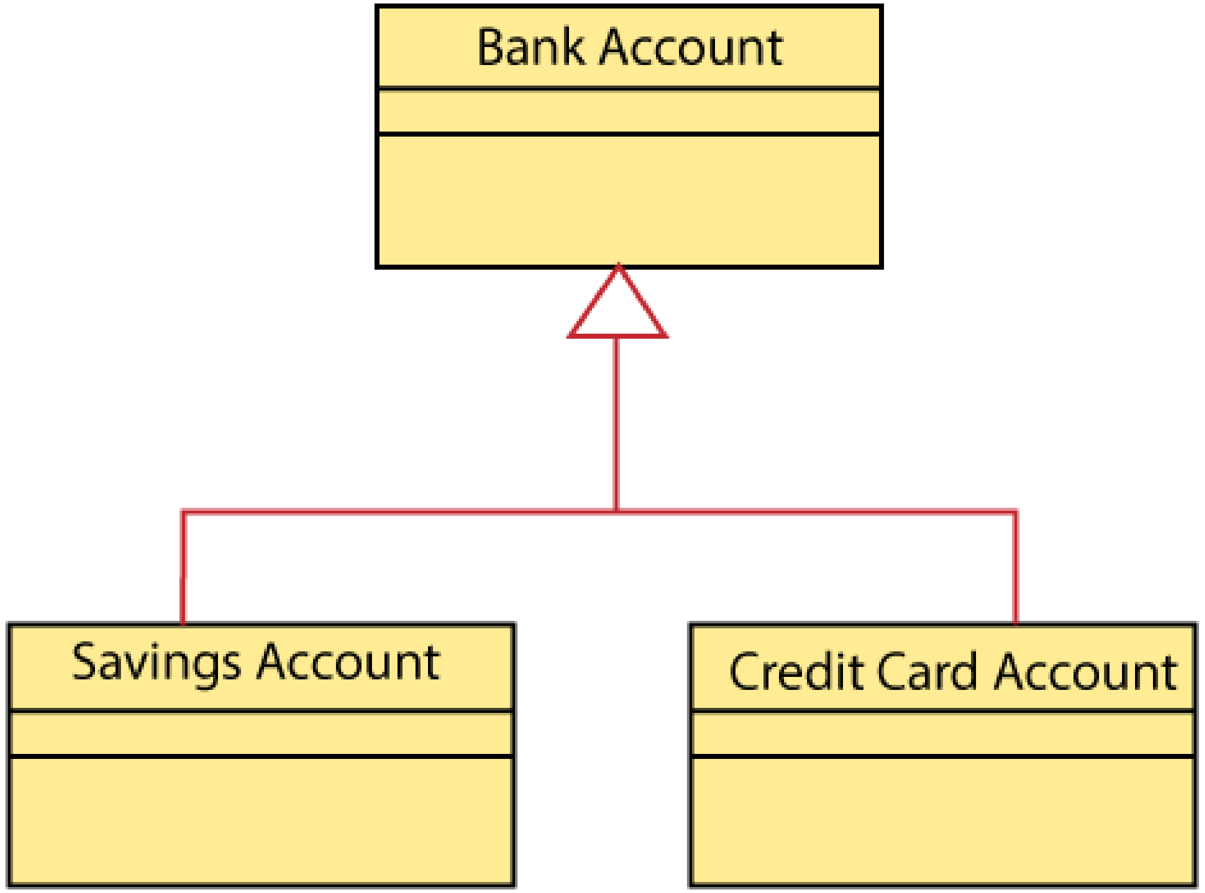
MORE EXPLANATION ONE EXTENDED RELATIONSHIP

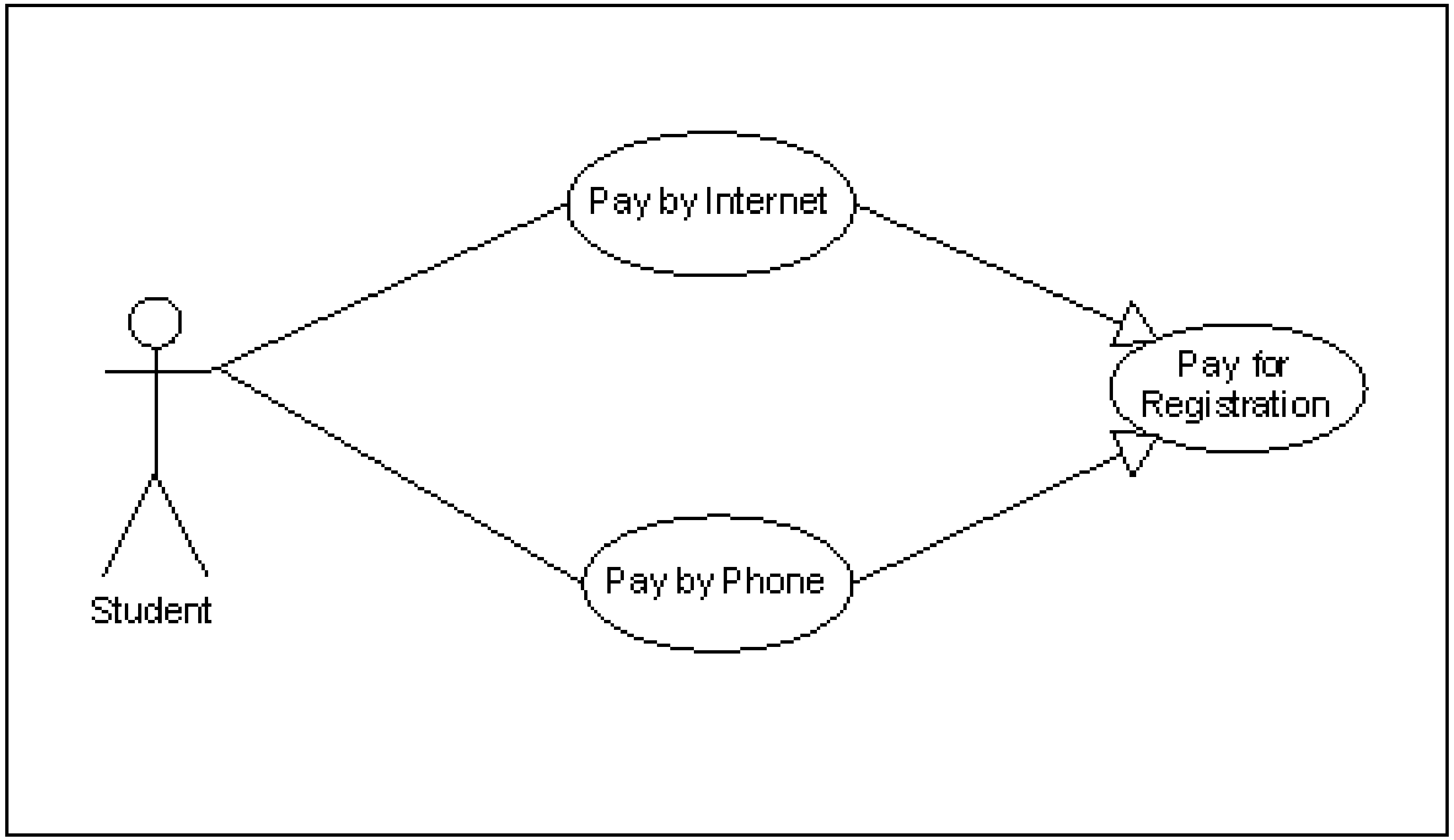
- Extend is a **directed relationship** that specifies how and when the behaviour defined in optional extending use case can be inserted into the behaviour defined in the extended use case.
- **Extended use case is independent of the extending use case and is meaningful on its own.**
- Extending use case typically defines optional behaviour that is not necessarily meaningful by itself.

GENERALIZATION RELATIONSHIP

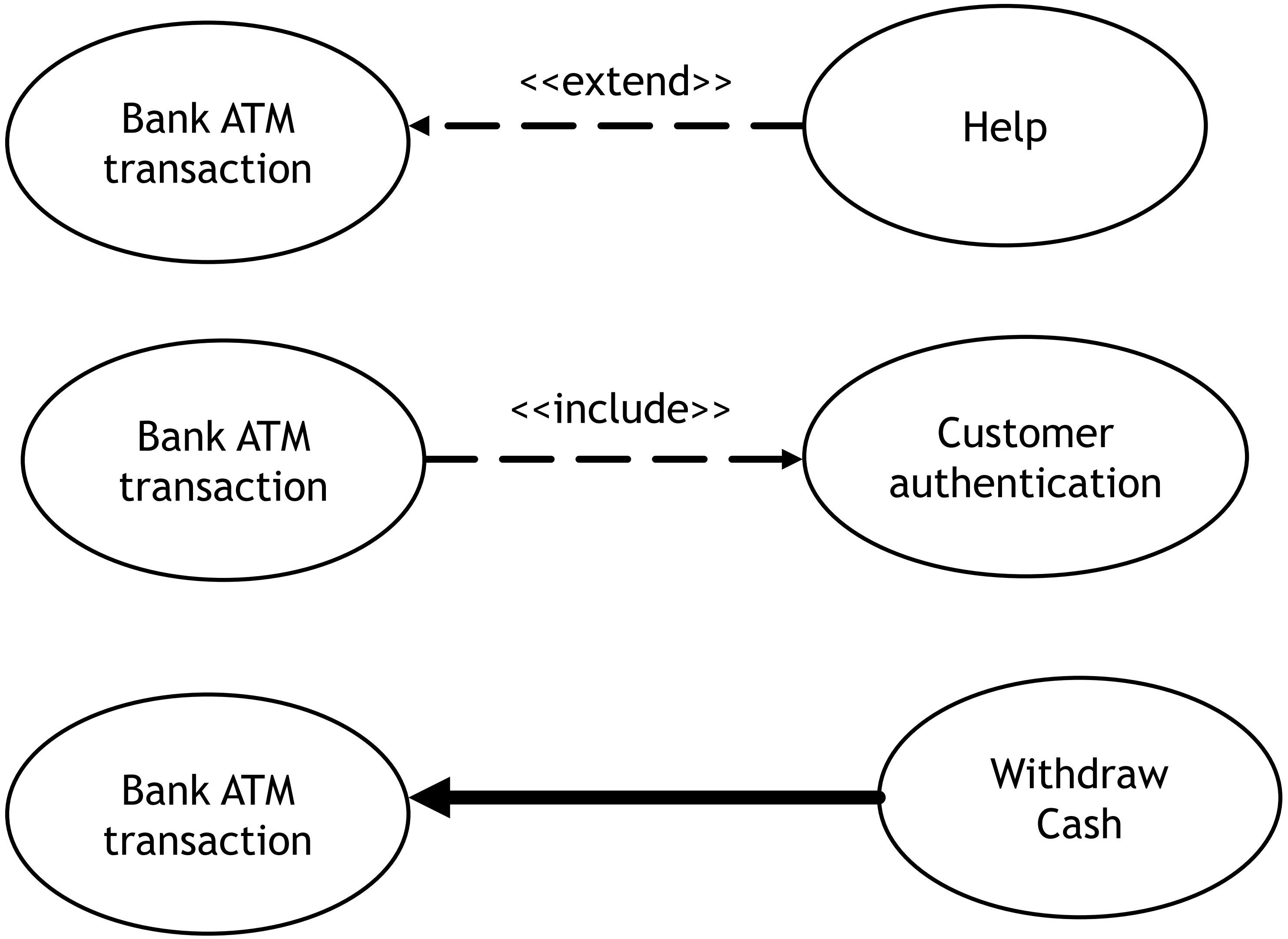
- Used to represent inheritance between use cases.
- A derived use case specializes some functionality it has already inherited from the base user case.
- One element is a specialization of another general component. It may be substituted for it.




Bank Account
Savings Account
Credit Card Account


Pay by Internet
Pay for Registration
Pay by Phone
Student

COMPARISON OF USE-CASE RELATIONSHIPS



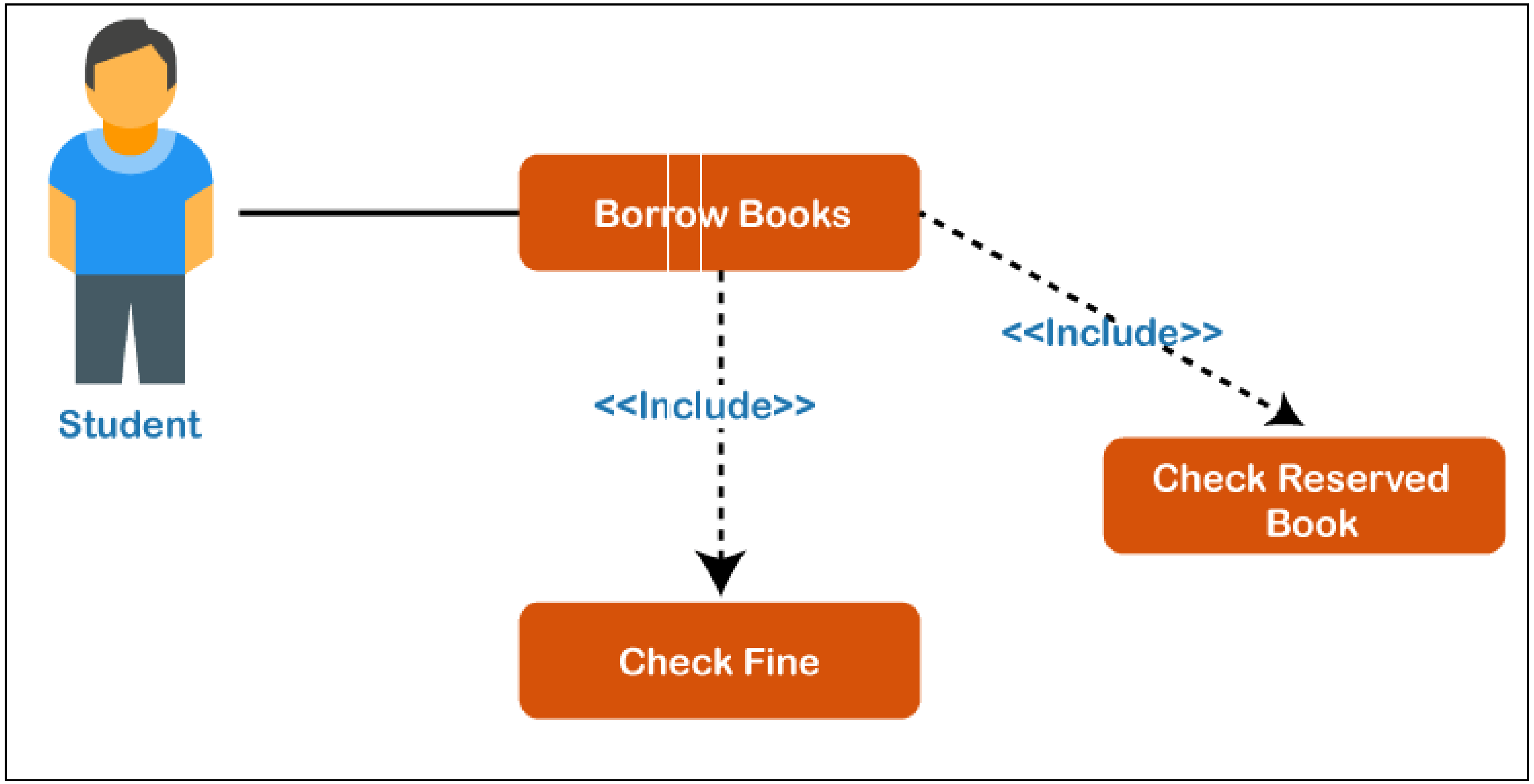
Extend Relationship

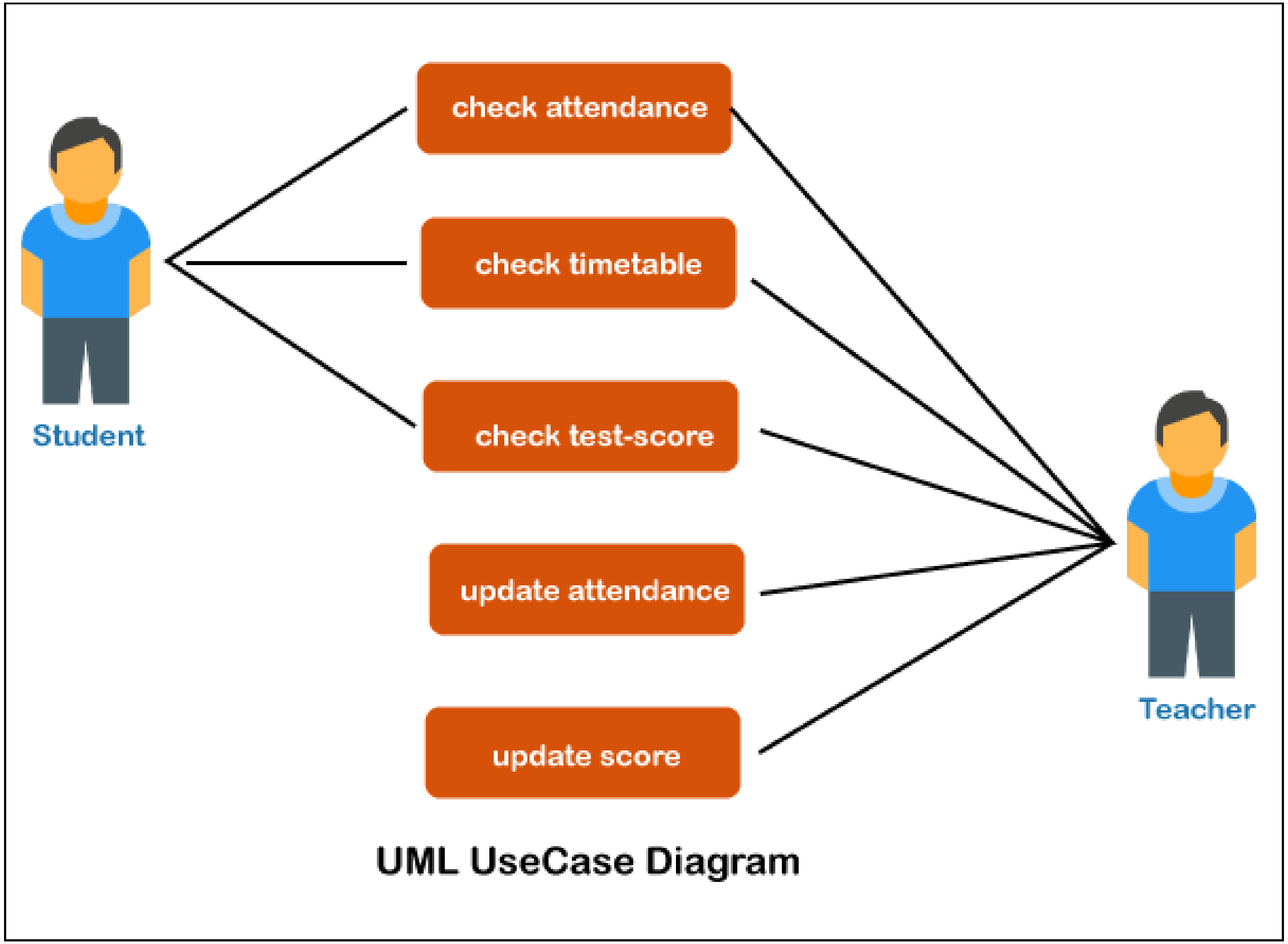
Include Relationship

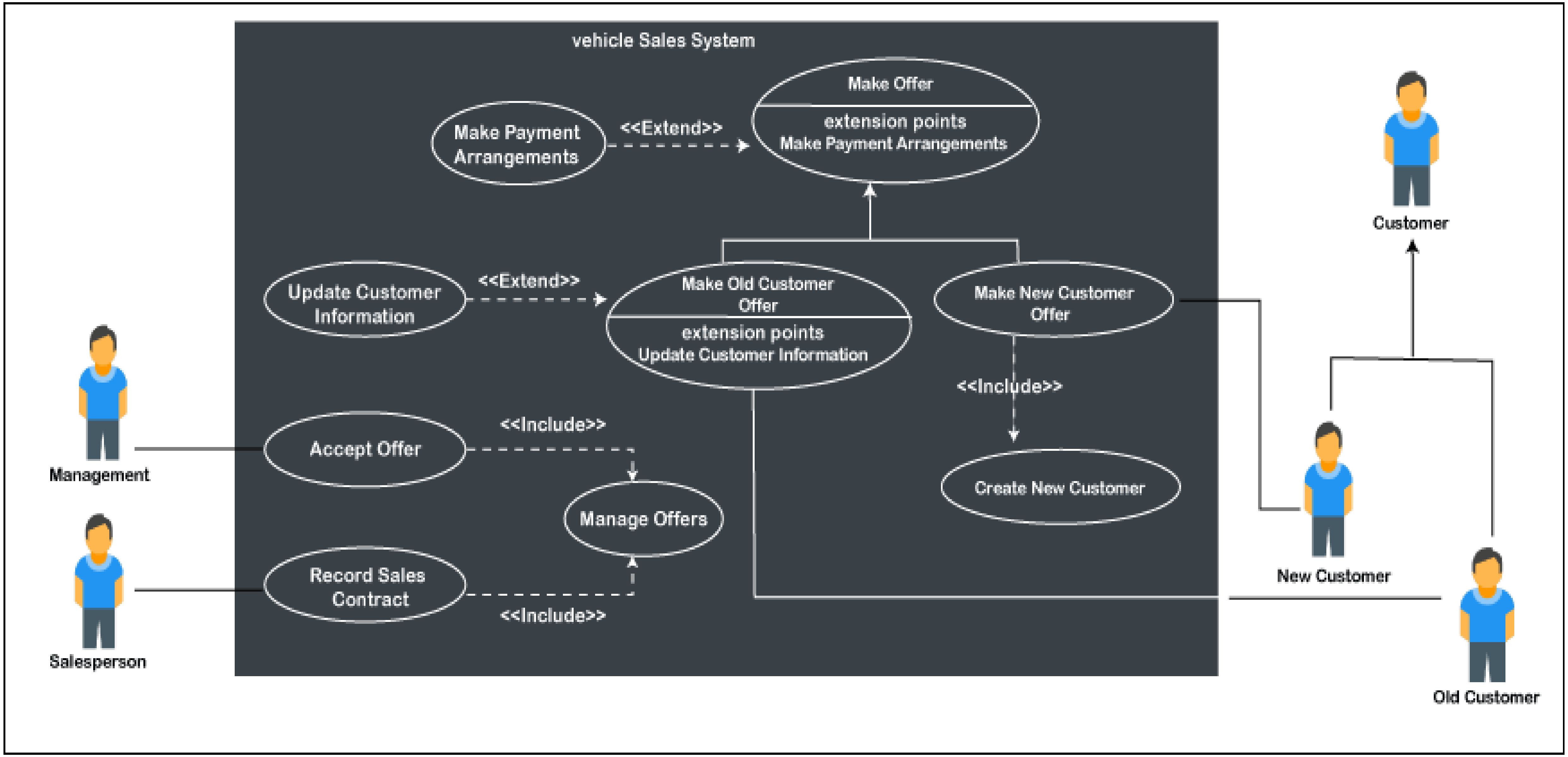
Generalization

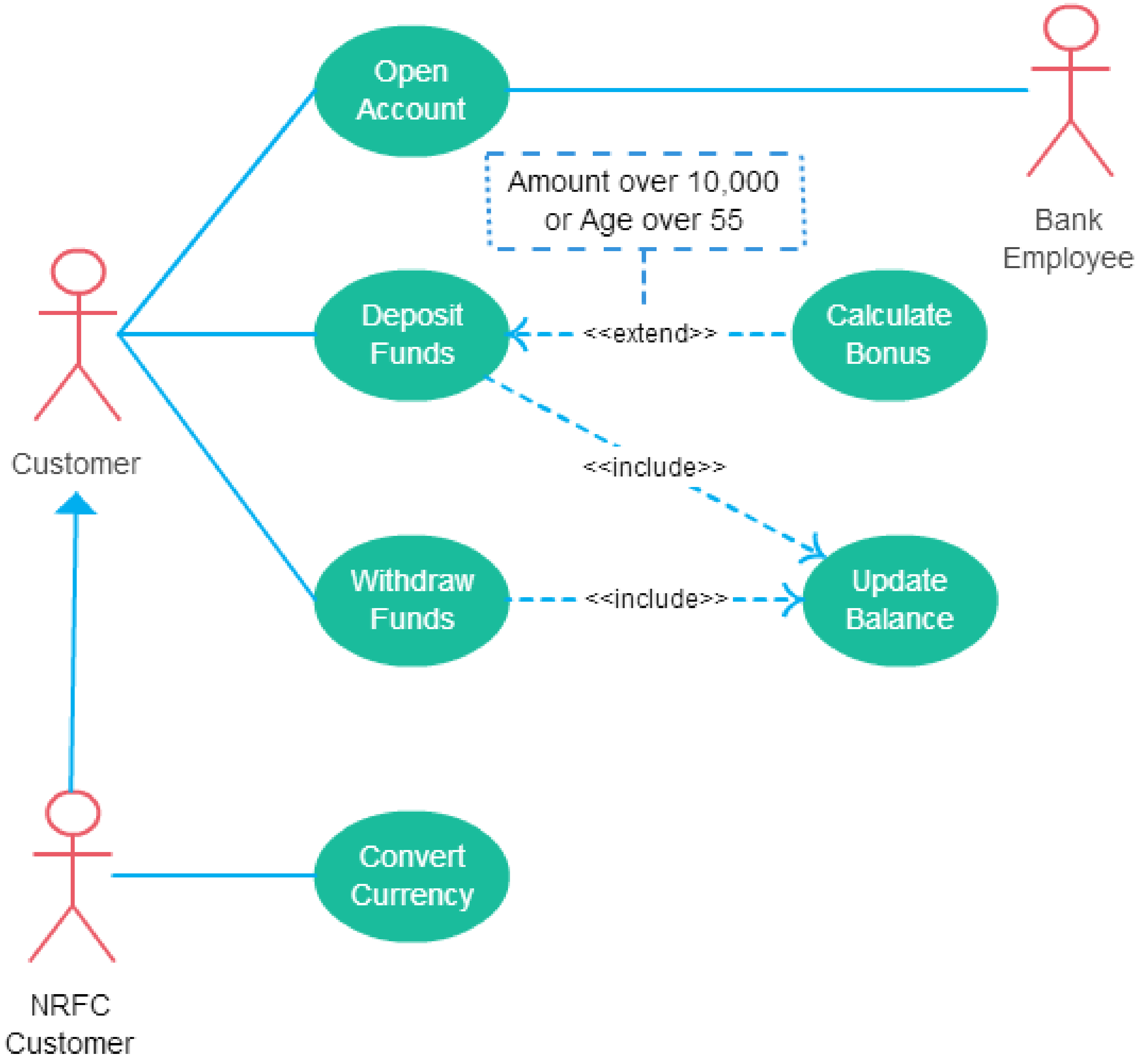
UML USE CASE DIAGRAM EXAMPLES

- Use case diagrams are both behaviour diagrams, because they describe the behaviour of the system, and they are also structure diagrams for they show the structure of the system.


Student
Borrow Books
<<Include>>
<<Include>>
Check Reserved Book
Check Fine


check attendance
check timetable
check test-score
update attendance
update score
Student
Teacher
UML UseCase Diagram


vehicle Sales System
Make Offer
extension points
Make Payment Arrangements
Make Payment Arrangements
<<Extend>>
Update Customer Information
<<Extend>>
Make Old Customer Offer
extension points
Update Customer Information
Make New Customer Offer
<<Include>>
Create New Customer
Accept Offer
<<Include>>
Manage Offers
Record Sales Contract
<<Include>>
Management
Salesperson
Customer
New Customer
Old Customer


Open Account
Bank Employee
Amount over 10,000 or Age over 55
Customer
Deposit Funds
<<extend>>
Calculate Bonus
<<include>>
Withdraw Funds
<<include>>
Update Balance
NRFC Customer
Convert Currency

HOW TO DRAM UML USE CASE DIAGRAM

DEFINE SUBJECT

- Subject is a business, software system, subsystem, component, device, etc. that you are designing or trying to understand how it is working.
- Properly define the kind of system, and what is its scope or boundary.
- Give it a proper name.
- Examples:
  - Department Store
  - Airport
  - Automated Teller Machine (ATM)
  - Point of Sale (POS) Terminal
  - Online Travel Reservation System
  - Electronic Prescription Service

DEFINE ACTORS

- Actors should have names according to the role they play in relation to our system. Examples of actor names (user roles):
  - Supplier
  - Passenger
  - Receptionist
  - Web Client
  - Bank
  - Payment Authorization System (external system example of actor)
  - Customer
  - Student
  - Analyst

IDENTIFYING ACTORS

- Given a problem statement, you can identify actors by asking following questions:
- Who gets the most benefits from the system?
- Who keeps the system working?
- What other software/hardware does the system interact with?
- Any interaction (interface) between the concerned system ad any other system.

DEFINE USE CASES

- Think of your functional requirements.
- Remember the review comments.
- Top-level use cases should describe complete unit of functionality provided to the actor. Examples of use case names:
- Hire Employee
- Transfer Funds
- Find Book
- Make Travel Reservations
- Schedule Patient Appointment

IDENTIFYING USE CASES

- Post primary and secondary actors identification, find out their goals.
- What functionality can these actors obtain from the system.
- Use case name should start with a verb like, “*Check balance*”, “*Gather use data*”.
- The name of the use-case must be descriptive. E.g. Do not just say “*print*”, rather say, “*print invoice*”.

LAB ACTIVITY

- Create a google presentation in your folders for SE.
- Create UML Use case Diagram for your respective projects.
- We will use another lab session for this activity as well, as I review your USE CASE diagrams in the class.
- This diagram needs to be finished because it will be a part of your SRS document.