Lecture 9: Use Case Diagrams

Learning Goals

- Define and distinguish between **User Stories** and **Use Cases**
- Describe a set of use-cases and **use-case scenarios** given an application
- Give examples of use-cases for a given system
- Draw a **Use Case Diagram** for a system
- Differentiate between **include**, **generalize**, and **extend** relationships

User Stories and Use Cases

- Capture the **functional requirements** of a system
- Describe interactions between **actors** and the system

Actor: a person or thing that has a "goal" in the system
(e.g. User, Organization, Software System)
Not necessarily a single individual, so sometimes referred to as **roles**

Goal: a purpose, or something to be achieved through interaction

User Stories take the viewpoint of an actor and list its goals

Use Cases describe all the actors and their goals in detail

User Stories

User Story: *"As a <role>, I want to be able to <some goal> so that <some reason>"*

http://dilbert.com/

Often used in the planning stage, each "story" describing one feature of the software.

Example User Stories – Video Rental Library

1. User stories

1.1 Most popular blu-ray discs sold

As a **customer** I want to **see the most popular blu-ray discs sold** so that **I can order one or many of them**

1.1.1 Sort by price

As a **customer** I want to **sort the most sold blu-ray discs by price** so that **I can see the less expensive ones first**

1.1.2 Sort by popularity

As a **customer** I want to **sort the most sold blu-ray discs by popularity** so that **I can see the most popular ones first**

Use Cases

Use Cases explore each User Story, analyze the **requirements** and how they can be **achieved** in terms of **interactions** between the actor and system.

- What **data** goes in/out (the interaction)
- What **effect** this has on the system
- What the **benefit** to the user is (the objective)

Describes the sequence of interactions to accomplish a specified, identifiable task, from a high-level perspective, and is written in **plain language**.

Use Cases: Library System

Library Use cases

- Checking out a book for loan.
- Checking in a returned book.
- Checking if a book is in stock and where to find it.
- Reserving a book that is currently out on loan.
- Dealing with payment of overdue fines.
- Adding new members to the library.
- Deleting old members from the library.
- Dealing with changes of members details e.g. name address etc.

Use Cases: Library System

- Checking out a book for loan.

What needs to take place for this to occur?

Benefit:
- Member goes home with a book

Interaction:
- Member identifies him/herself with id, indicates which book
- If user is allowed and book is available, lend book for a specified period
- Update records of loan and availability

Effects:
- Member's loan record updated to reflect loaned books
- Library's loan record updated to show book availability

Use Cases: Library System

Use-Case: Borrow Book

- The member identifies him/herself to the librarian using membership identification card.
- The member presents one or more books to the Librarian.
- The Librarian checks the books to make sure they can be loaned.
- The Librarian checks the membership card to make sure it is valid.
- The Librarian looks up the member's records and checks for any due fines and that the number of loaned books will be less than 6 (the maximum).
- The Librarian refuses to loan books to members with overdue fines.
- If acceptable, each book is stamped with the appropriate return date (2 weeks from today).
- The Librarian updates the member's loan details by entering it into the loan library.

Use Cases: ATM

ATM Use cases

- Withdraw Cash
- Request Balance
- Request Statement
- Request Cheques

By Richard001 (Own work) [Public domain], via Wikimedia Commons

- Withdraw Cash

Benefit: - User goes home with cash money

Interaction:

- User identifies him/herself
- Requests amount
- System checks balance and limits
- If valid, dispenses cash, debits account
- Asks user if would like receipt

Effects: - User's account balance updated

- ATM's available cash updated

Use Cases: ATM

Use-Case: Withdraw Cash

- The user inserts their ID card into the system.
- The system reads the card's chip to identify the user & account.
- The system prompts the user to enter their PIN.
- The user enters their PIN.
- The system contacts the banks central computer to verify the PIN account details.
- If PIN is authenticated the user is prompted for the amount of the withdrawal. If not, the card is returned to the user with an appropriate failed identification message.
- The system prompts for the amount of the cash withdrawal.
- The user enters the amount of the cash withdrawal.
- The system checks with the banks central computer to ensure that the user has sufficient funds.
- If there are sufficient funds, the cash is dispensed and the customer's account at the Bank Central Computer is debited accordingly, otherwise an appropriate "insufficient funds" message is displayed
- The card is returned to the user and a receipt is printed.

Use Cases

What to write and how much detail is necessary?

There are no explicit rules or syntax. It must capture everything the system should do, otherwise the feature may be missed.

Capture all the **interactions** you expect will be played out, the **benefits** to the user or actor, and the **effect** it has on the system.

Should focus on the "**what's**", not the "how's", i.e. should not go into the underlying details required to get the functionality to work.

Use Case Scenarios

Specific instance of a use case that is played out.

What *could* happen? What do we do in that case?

ATM: What if the users PIN is incorrectly entered?
What if the user has insufficient funds in their account?
What if the cash dispenser cannot read the?
What if the cash dispenser is out of money?
What if the bank central computer is off-line?

Each scenario addresses one of **path of interation**, but are all part of the same Use Case. They are **not errors**, since they may reflect important logic or business operation.

Use Case Scenarios

Often documented using **structured pseudo-code**, but keep it **simple**, **unambiguous** and **clear** to others.

Start of Primary scenario/transaction

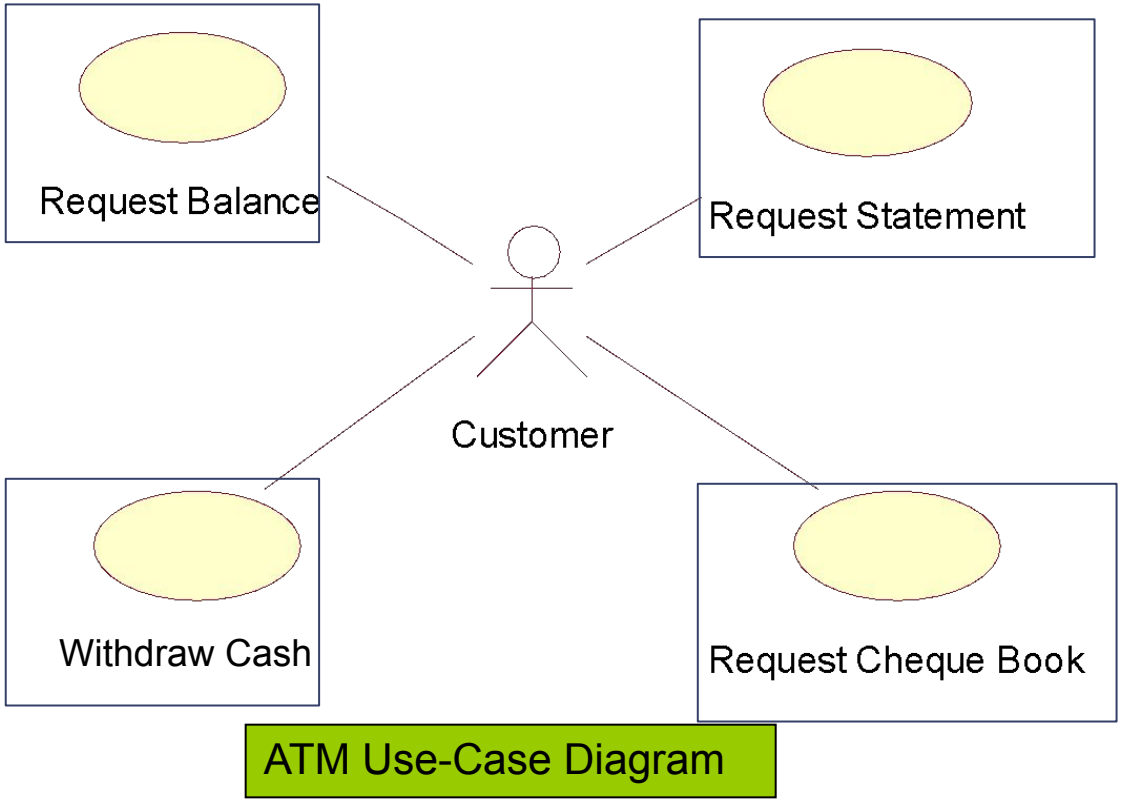
1. The user inserts their ID card into the system.
2. The system reads the magnetic strip from the card.
3. If the system cannot read the card then **<<Scenario 1>>**
4. The system contacts the banks central computer to request the PIN number for the card and their account details.
5. If bank central computer cannot access users account then **<<Scenario 2>>**
6. The system prompts the user for their PIN.
7. The user enters their PIN.
8. If PIN cannot be authenticated **<<Scenario 3>>**
9. The user is prompted for the amount of the withdrawal.
10. The user enters the amount of withdrawal.
11. The system checks with the banks central computer
12. If the user has insufficient funds **<<Scenario 4>>**
13. The cash is dispensed and the customer's account at the Bank Central Computer is debited with the withdrawal amount.
14. The card is returned to the user and a receipt issued.

End-Of-Transaction

Scenario 1: The users card is returned. End-of-Transaction
Scenario 2: The users card is returned. End-of-Transaction
Scenario 3: The user is given two more attempts to enter a correct PIN. If this fails the card is kept and the transaction ends. Otherwise resume primary scenario.
Scenario 4: The user is given the opportunity to enter a lesser amount or cancel the transaction. If cancel is chosen, the card is returned and the transaction ends. If the lesser amount is acceptable then resume primary scenario.

Use Case Diagrams

Very simple diagram involving 4 elements: “actors”, “use-cases”, “system” and “association”

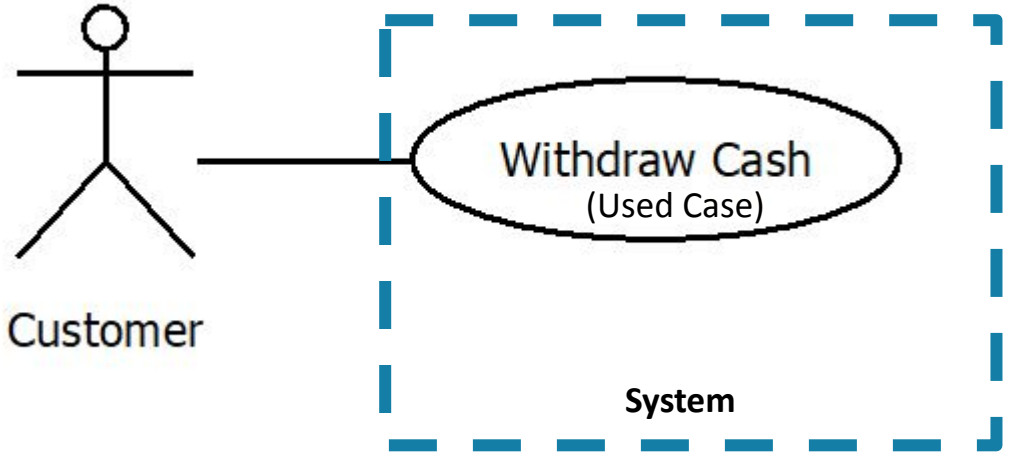


ATM Use-Case Diagram

Use Case Diagrams

Each oval represents a unique use-case, with a high-level label. All use-cases must be **initiated** by an actor.

Use cases are documented elsewhere, with a detailed description of the interactions.



Use Case Diagrams

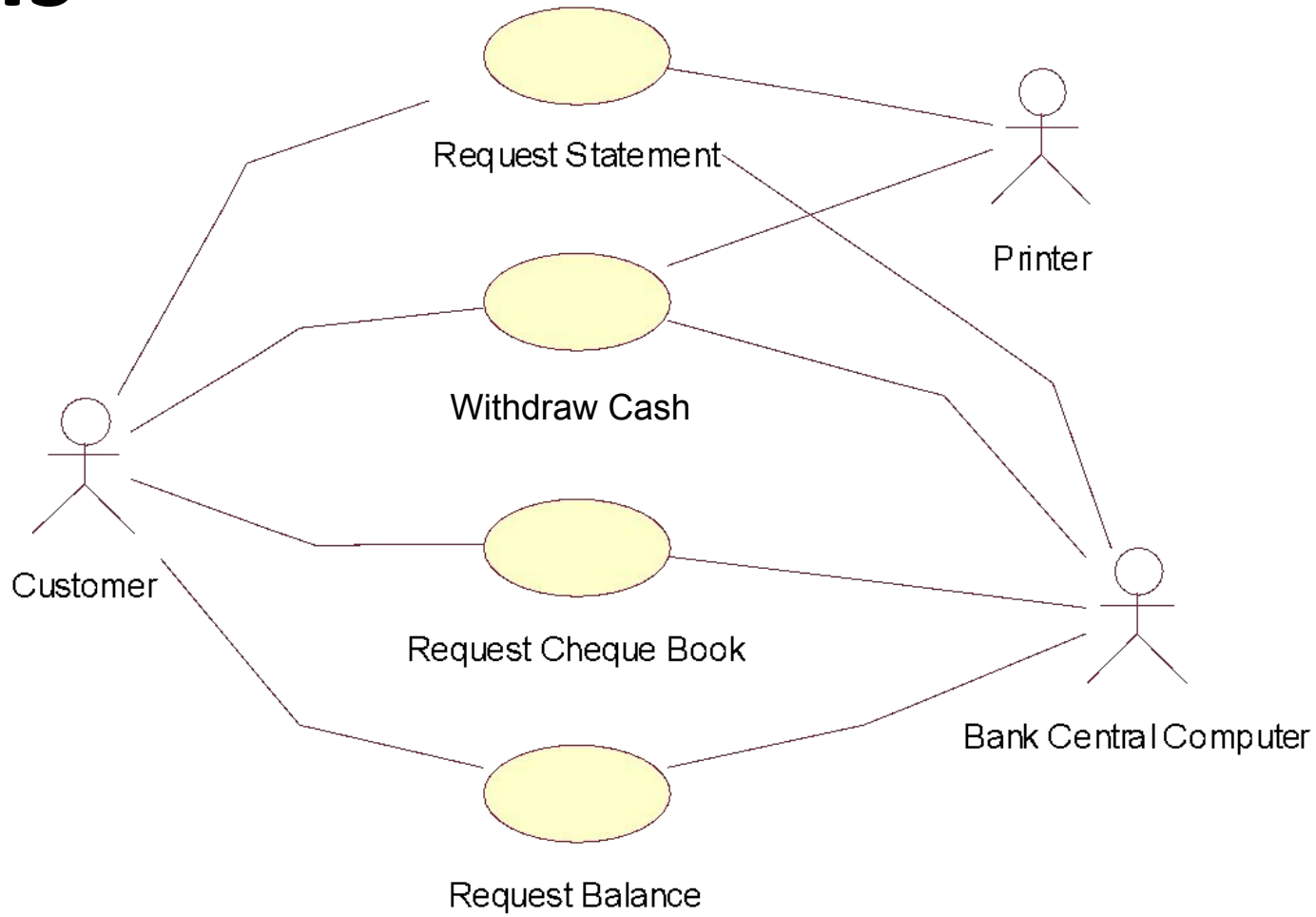
An actor is an **external entity** we are modelling. They could be users, other systems, hardware, etc. They initiate one or more use-cases.

Not all people (or entities) involved in execution is an actor. Actors ***must*** **initiate** a use-case and gets some measureable **benefit** from the interaction, not just someone/something that happens to be involved.

Use Case Diagrams

We *may* show other actors that are involved if it is **important to the use-case**.

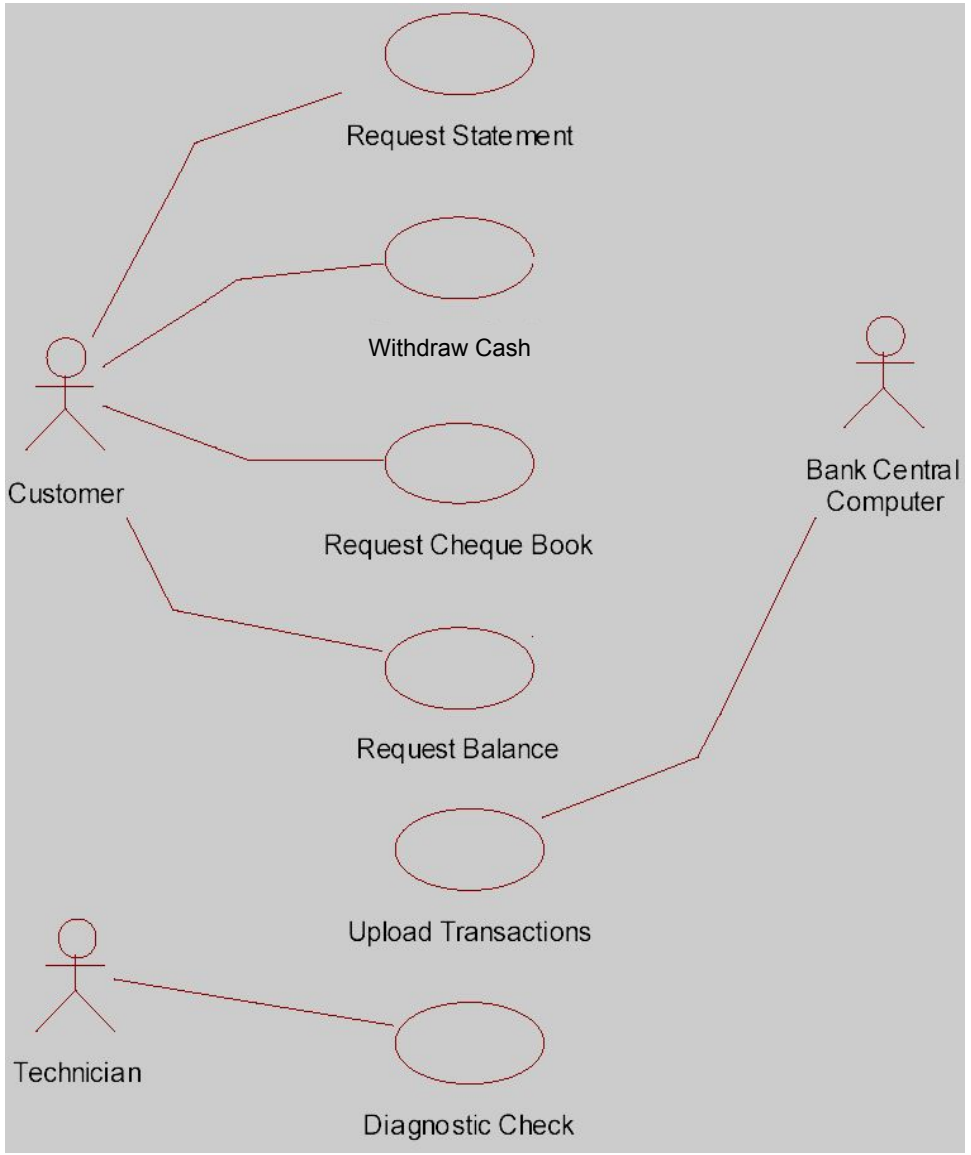
In UML, we refer to “actors” who do not initiate a use-case as a **secondary actor**. The initiator is the **primary actor**.



Use Case Diagrams

There may be multiple primary actors in your system.

e.g. the Bank's Central Computer requests all transactions be uploaded, or a technician runs a diagnostics check

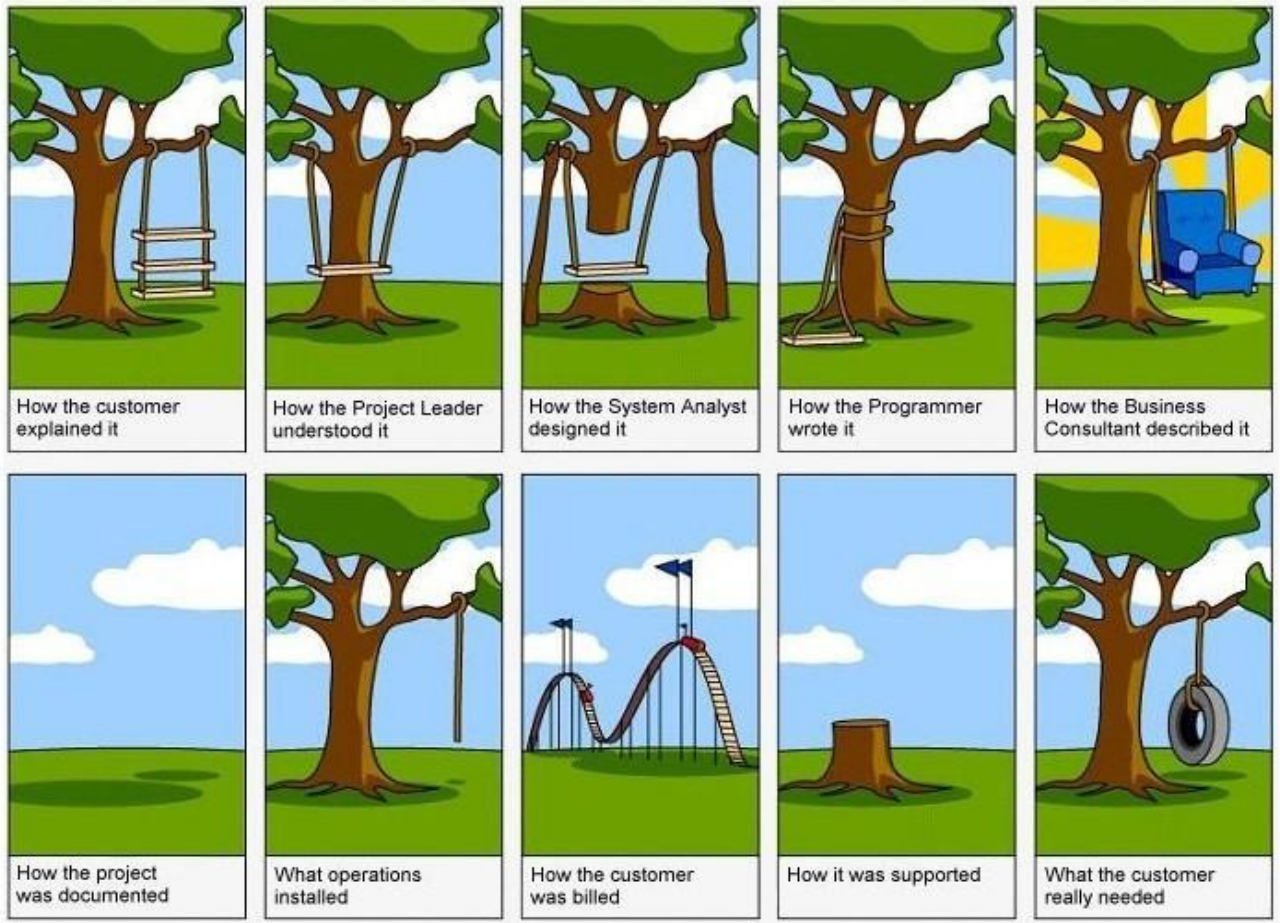


Use Case Diagrams

Why bother with such simple diagrams?

- They show the **big-picture** without getting bogged down by the details of design and implementation
- Focuses on what needs to be done, useful for organizing tasks
- Useful for **communicating** with customer, relating use-cases to major objectives
- Helps **reduce missed** or **misunderstood** features or functionality during analysis
- Developers can immediately assess **required functionality**, assess risks and identify any potential **challenges**

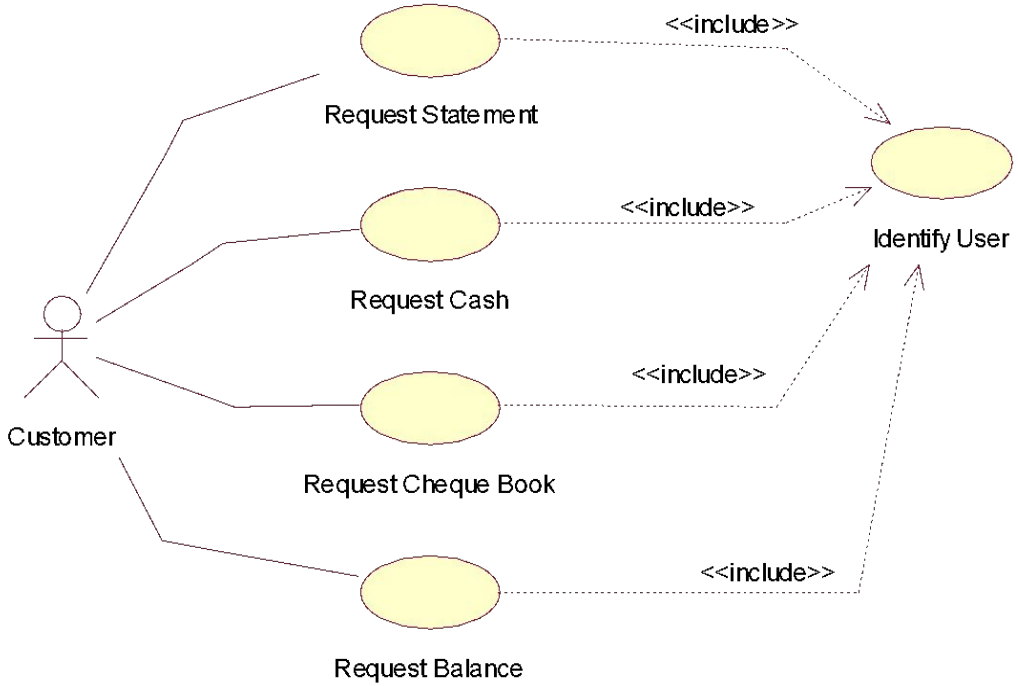
Use Case Diagrams



Use Case Diagrams: Includes

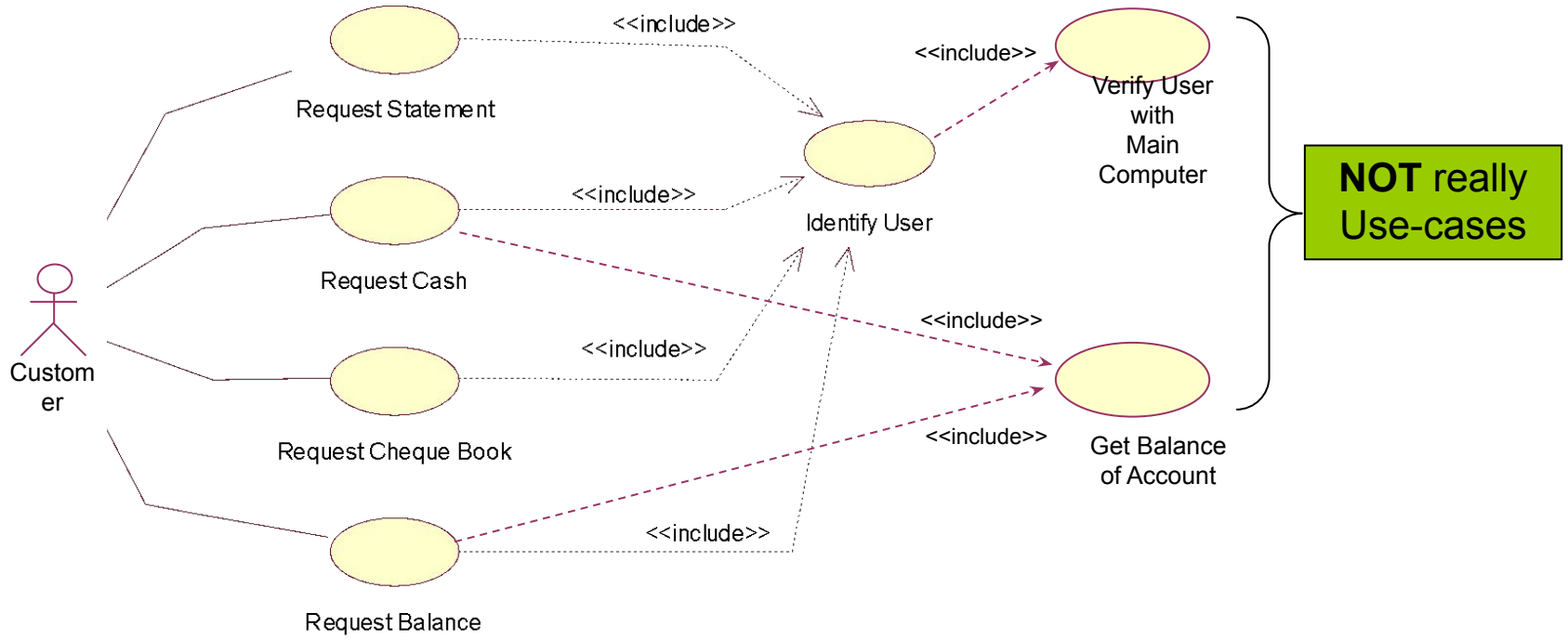
Sometimes **commonality** exists between steps, e.g. each operation at an ATM requires an identification step using a bank card and pin.

Rather than duplicate the common interaction within each use-case, we extract into a “mini use-case” which can be **included** in the others.



THIS IS NOT BREAKING DOWN INTO FUNCTIONS! The included mini use-case should still involve some **interaction** between actor and system, with **benefits** and **effects**.

Use Case Diagrams: Includes



If a use-case does not contain any user-interaction OR, does not lead to any direct, measurable benefit for the user, then it is NOT a use-case, it is simply functionality.

Documenting Includes

Start of Primary scenario/transaction

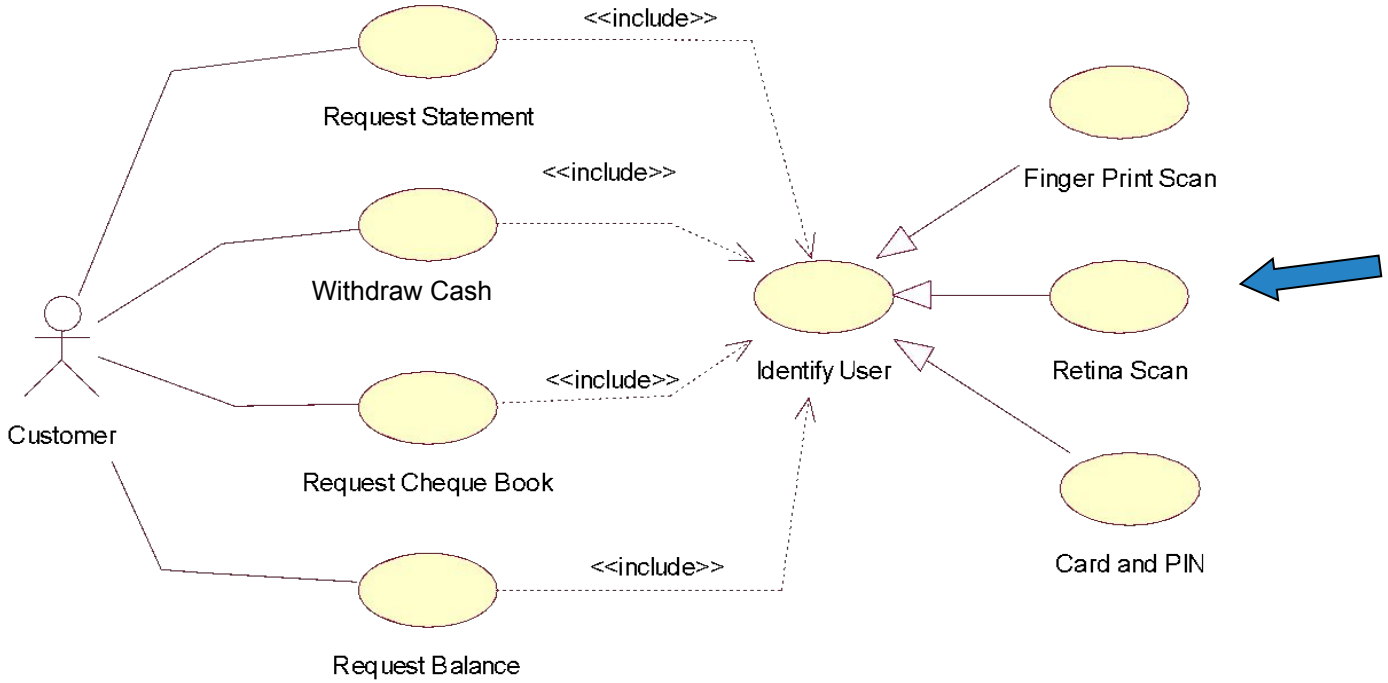
1. Include ***Identify User*** ***(a prerequisite or precondition for the execution of this use-case)***
2. If identification fails **<< Scenario 1>>**
3. The system contacts the banks central computer to request the PIN number for the card and their account details.
4. If bank central computer cannot access users account **<<Scenario 2>>**
5. If PIN cannot be authenticated **<<Scenario 3>>**
6. The user is prompted for the amount of the withdrawal.
7. The user enters the amount of withdrawal.
8. The system checks the account balance with the banks central computer
9. If the user has insufficient funds **<<Scenario 4>>**
10. The cash is dispensed and the customer's account at the Bank Central Computer is debited with the withdrawal amount.
11. The card is returned to the user and a receipt issued.

End-Of-Transaction

Scenario 1: The users card is returned. End of Transaction
Scenario 2: The users card is returned. End of Transaction
Scenario 3: The user is given two more attempts to enter a correct PIN.
If this fails the card is kept and the transaction ends.
Otherwise resume primary scenario.
Scenario 4: The user is given the opportunity to enter a lesser amount or cancel the transaction.
If cancel is chosen, the card is returned and the transaction ends.
If the lesser amount is acceptable then resume primary scenario.

Use Case Diagrams: Generalzation

When two or more use-cases achieve the **same goal**. e.g. Unlocking an iphone: passcode, OR fingerprint, OR face recognition.



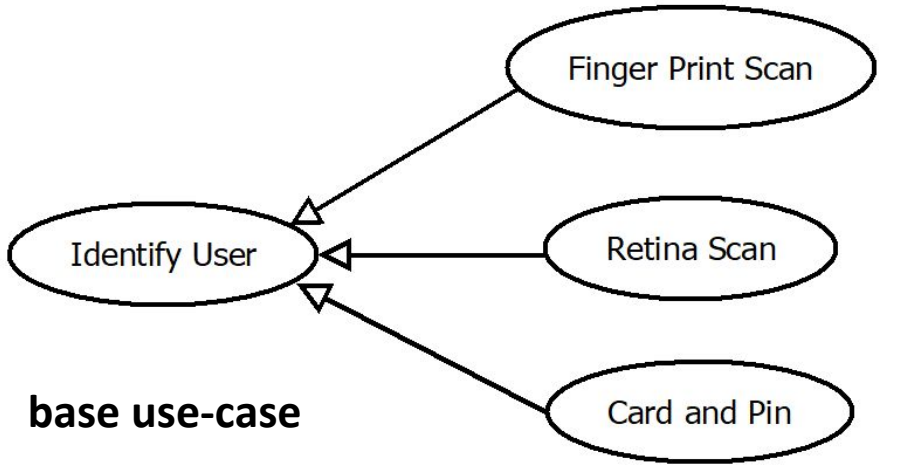
'Finger Print Scan', 'Retina Scan' and 'Card and Pin' are all 'kinds of' use case that identify the user

The outcome/benefits are the same, but interaction is different.

Use Case Diagrams: Generalization

The base (or root) use-case should be documented in very **general** terms.

> "*Identify the user*"
> *obtain their account details from the bank central computer*



Generalization is about **isolating common objectives** and expressing that commonality in the use-base case.

The details are then described in the derived use-cases.

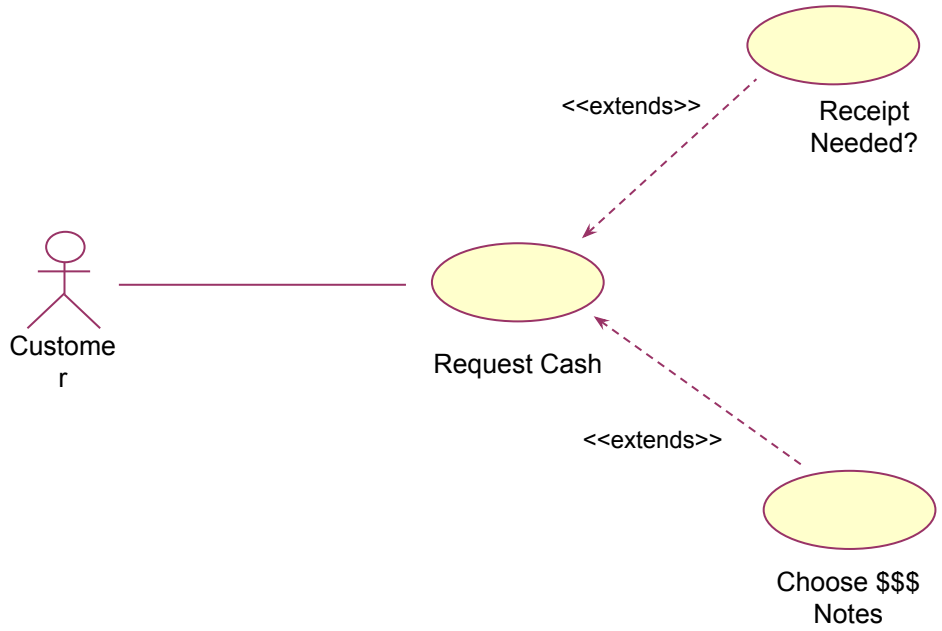
Use Case Diagrams: Extends

Allow for extensions of use-cases. Model **optional behaviour** involving user-or actor-**interaction**.

Use-Case Request Cash

1. . . .
2. . . .
3. If(users wants to chose type of $ notes)
4. Extend use case : Choose $$$ Notes
5. If user chooses to have a receipt
6. Extend use case : Print Receipt
7. . . .
8. . . .
9. The card is returned to the user and a receipt is issued.

End-Of-Transaction



Use Case Diagrams: Summary

User stories are used to develop and analyze the list of **functional requirements**

Use-Cases describe the **details** of the **interactions**, can include multiple **scenarios** leading to different interaction procedures. Descriptions do not have a specific syntax, but must capture **requirements** in a way that analysists and customers can **understand**.

Use-Cases can **include** other use-cases, can sometimes be **generalized** – having multiple derived use-cases, and can **extend** other use-cases adding optional interactions.

Note the arrow types in the diagrams!! Both the **style** and **direction**.