LECTURE-7 & 8

Phase – 2: Requirement Modelling

Requirements validation and negotiation, Modeling requirements, Use Case Diagrams, Class Diagrams

Suleman Shahid

Based on slides from

Requirement Engineering

- Requirement elicitation and analysis
- Requirement Specification (Modelling)
- Requirement Validation
- Requirement Documentation

Requirement Validation & Negotiation

3

Principles of Requirement Validation

P1: Involvement of **stakeholders**
P2: **Separating** the identification and the correction
P3: Validation from different **views**
P4: Adequate **change** of documentation type
P5: Repeated validation

4

Requirements validation techniques

- Requirement reviews
  - Customer involvement (commenting)
  - Formal and informal reviews (inspection)
  - Walkthrough

- Perspective based Reading
- Using checklist for validation
- Validation using prototype

Requirement Negotiation - Conflict Identification

- Conflict of subject
  - "R131: The reaction time of the planned system shall not exceed one second".
- Conflict of interest
  - Subjectivity or objectivity of the goal, e.g., cost vs quality
- Conflict of value
  - Due to Cultural differences, personal ideas etc.
- Relationship conflict

- Conflict analysis
- Conflict resolution

Phase -2 Software Design

7

Communicating Design Ideas in Team Environment

- Representing architecture and behaviour of the system through models.
- A model is a description of the structure and meaning of a system

U M L

8

Unified Modelling Language (UML)

- A modeling language allows the specification, the visualization, and the documentation of the development of a software system
- The models are artifacts which clients and developers use to communicate
- UML 1.* is a modeling language
- UML 2.* is also a programming language
- Also known as Universal Modelling Language

UML Basics

The soul never thinks without an image
– Aristotle

UML

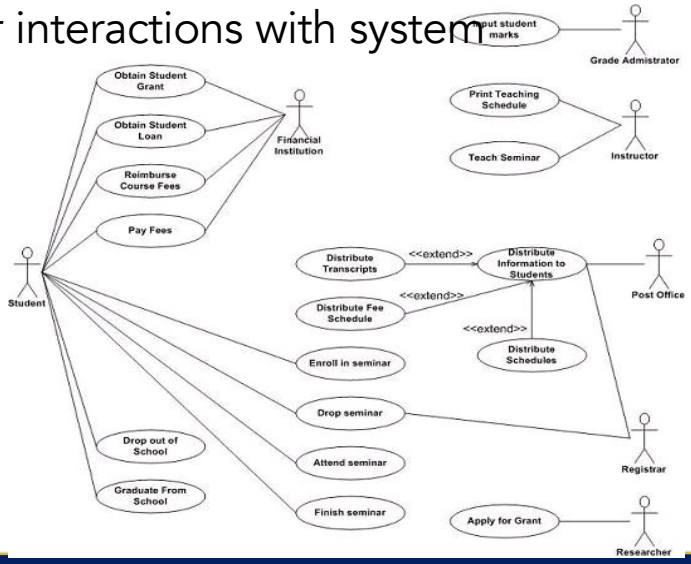
- Structure
  - What the system is made of
  - Components and relationships
  - Hierarchy, network, lattice etc.
- Behavioral
  - Interactions and communication
  - Flow of the system

Diagram Types

- Use Case Diagram: Modelling user interactions
- Class Diagram: Model the structure of the system and identifies relationship between different objects
- Collaboration Diagram (Sequence and communication): Model the behaviour of the system
- State Charts: Time/event dependent behavior of the system
- Activity Diagrams: Concurrent activities in the system
- Deployment Diagrams: Models the physical layout of the system

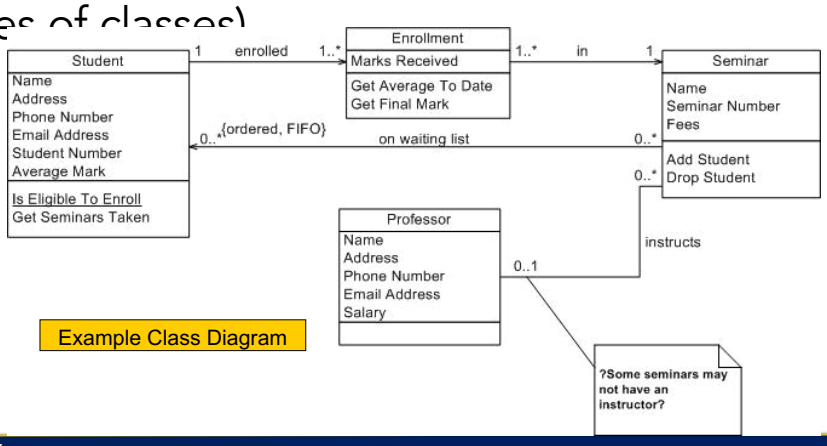
Use-case Diagram

- Shows a user interactions with system



Class Diagram

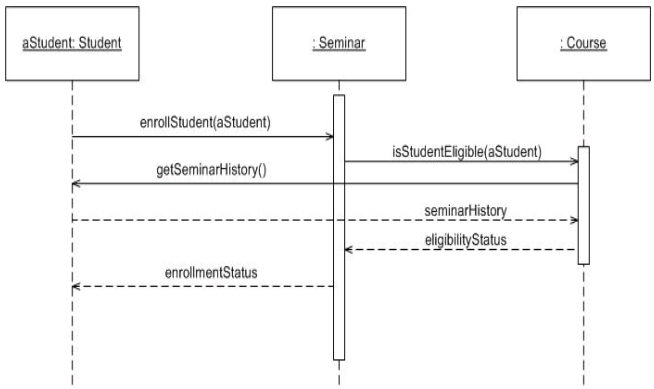
- A powerful tool for exploring architecture, functionality and relationships between objects in our system (i.e. instances of classes)



Example Class Diagram

Collaboration Diagrams – Sequence & Communication

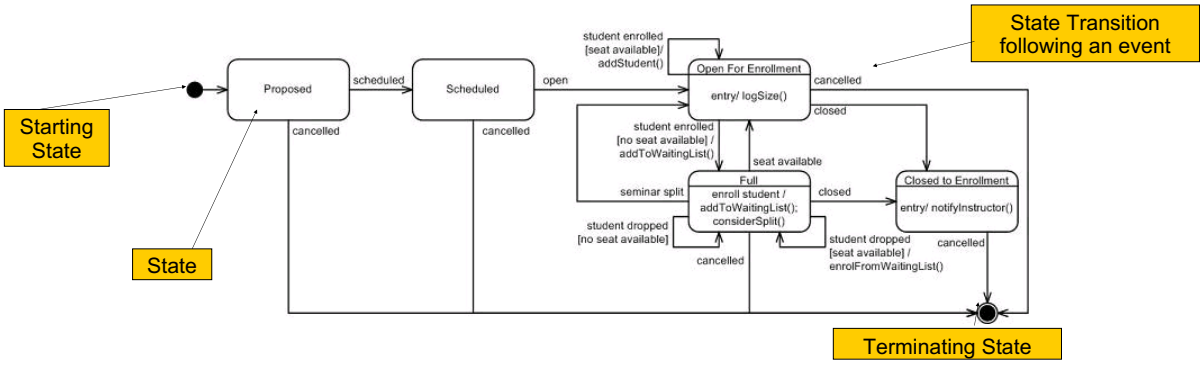
- Model the behaviour of the system in response to inputs from the external world.
- Model the interaction of a set of collaborating objects through a process of message passing to achieve the functionality expressed in one or more use cases.



Example Sequence Diagram

State Charts

- An extension of state transition diagrams.
- Model the time dependent behaviour of objects or systems in response to messages sent to it over a period of time.



Activity Diagrams

- Used during analysis to explore areas of parallelism or concurrency in the customer business model.
- Useful in areas of business systems modelling where many processes or activities within the business are carried out in parallel, e.g. simultaneously raising an invoice while at the same time producing a delivery note and shipping the goods.

Deployment Diagrams

- Deployment diagrams show how software will be deployed across a distribution of computers and networks
- A sketch of the system's physical architecture e.g. computers, disk drives, GUI's, databases, hardware interfaces, networks, programs running on system X and Y.

http://www.agilemodeling.com/essays/umlDiagrams.htm

Use-Case Diagrams

Modeling User Interactions

- Consider the process of automating a library
  - Stakeholders?
    - Administrator, Librarian, User
  - Requirements?
    - Borrow book, Check-in book, Reserve borrowed book, Add book, Add user, etc.

User Interaction – Librarian

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.

User Interaction – Administrator

Use-cases

- Adding new copies of a book to the library.
- Deleting old copies of a book from the library.
- Issuing 'Book Overdue' letters and fines.

Use-Case

- A process or procedure, describing a user's interaction with the system for a specified, identifiable purpose.
- Each Use-Case describes a step-by-step sequence of operations, iterations and events that document
  - The Interaction taking place.
  - The Measurable Benefits to the user interacting with the system.
  - The Effect of that Interaction on the system.

Use Cases

- Three key elements of use cases:
  - Environmental agents or actors that will cooperate with the system
  - Use case diagram shows relations of the actors and use cases
  - Use case scenarios are plans for realization of user goals or handling contingencies

Deriving a Use-Case from System Requirement

- RQ1: Users should be able to borrow books from the library.
  - UC: borrowing a book
- Define the objectives of the use-case, from the point of view of the user.
  - who is the user
  - what interaction takes place from their perspective
  - what benefits the user gets from that interaction?

Use-Case "Borrow Book"

- The borrower/member identifies himself or herself to the librarian using their membership card.
- The borrower/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 in his/her card indexing system and checks that the number of loaned books will be less than 6 (the maximum that can be loaned at any time to a library member).
- If acceptable, each book is then stamped with the appropriate return date (2 weeks from today).
- Each book has its identifying card removed from the inside cover.
- The Librarian updates the member's loan details by placing the identifying cards into that members record maintained by the library.
- Automating the process:
  - Bar codes to identify members and books
  - a database to replace the card indexing system and maybe even the librarian (either completely or in part).

Exercise

- Identify users that initiate the interaction, benefits from that interaction and effect on the system for the following use cases:
  - 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.

Example – A Cash Dispenser/ATM

- Who are the potential users?
- What are the use cases?
  - Request Cash
  - Request Balance
  - Request Statement
  - Request Cheque book

Use-Case "Request Cash"

- The user identifies him/herself to the system and requests a withdrawal for an amount of cash.
- A check is made, to make sure their account would remain in credit after the withdrawal, and if so, they are dispensed the cash and their account is debited accordingly.

Use-Case "Request Cash"

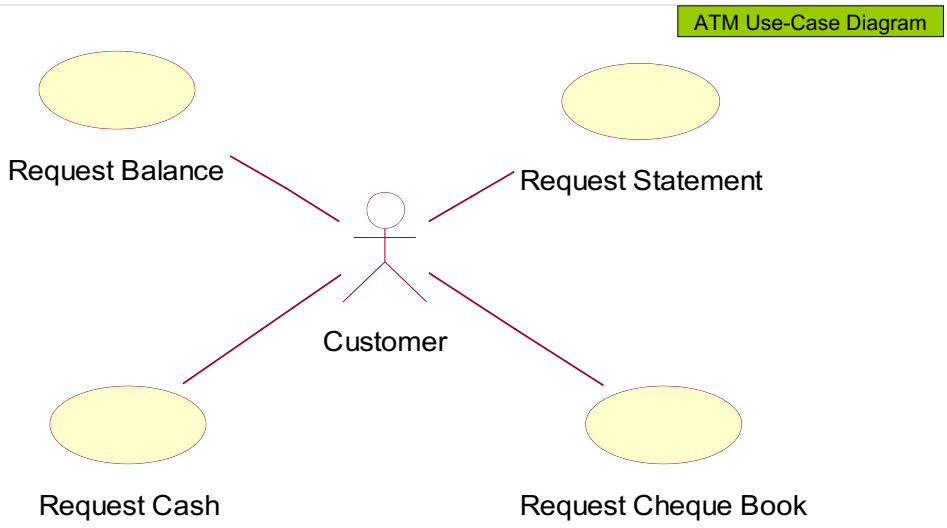
- The user inserts their ID card into the system.
- The system reads the magnetic strip from the card to identify user & account.
- The system contacts the banks central computer to request the PIN number for the card and their account details.
- The system prompts the user to enter their PIN.
- The user enters their PIN.
- 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 to make the cash withdrawal.
- 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-Case Diagram

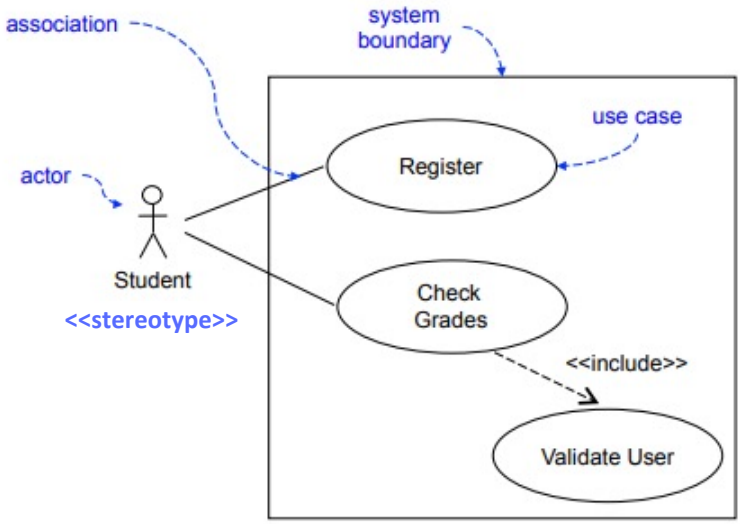
- A **stick figure** referred to as an '**actor**'
  - External entity outside the domain of the system
  - Initiates interaction with system to get measurable benefit
  - Primary actor, initiating actors are already identified in user stories
  - Secondary actor, participating actors identified as part of use case analysis

- A **named oval** representing each of the identified **use-cases**.
  - High level requirements or functionality
  - Initiated by actor

Use-Case Diagram

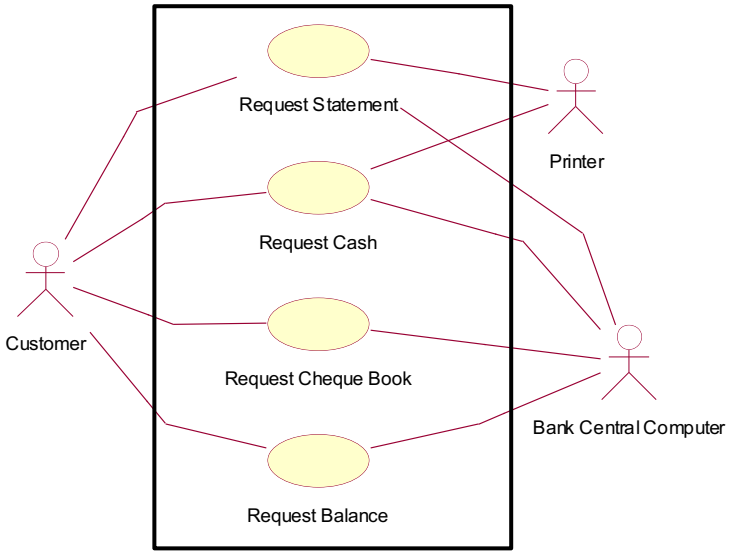


Terminologies

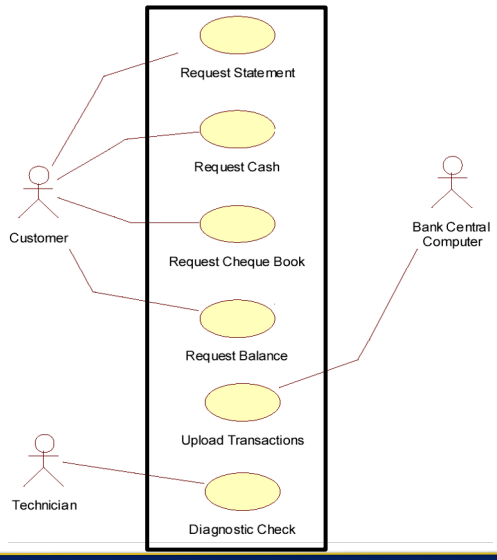


Use-Case Diagram

Secondary actors do not initiate a use-case.



What if Bank Central Computer Initiate a Use-case?



Use-Case Scenarios

- A scenario is a specific instance of a use case that is played out between actor and system at run time.
- For example, the "ATM Request Cash" use-case.
  - What could happen when a particular customer comes to make a cash withdrawal?
    - Scenario: User shall enter the correct PIN and have sufficient money in their account to make the requested withdrawal.
    - What if the customer incorrectly enters their PIN and the transaction is aborted.

ATM Request Cash use-case.

- Scenario are for every 'what-if' type question that can be posed during analysis.

- For example
  - 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 cards magnetic strip?
  - What if the cash dispenser is out of money?
  - What if the bank central computer is off-line?

Use-case Scenarios

- Scenarios are NOT Errors

- A scenario captures the many different possible interactions and outcomes that could occur when executing a specific use-case.

- A use-case binds together a set of scenarios that a user could face when interacting with the system for a specific goal, objective or aim.

Use-case Request Cash – Scenarios

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

Scenario 1: The users card is returned.
Scenario 2: The users card is returned.
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.

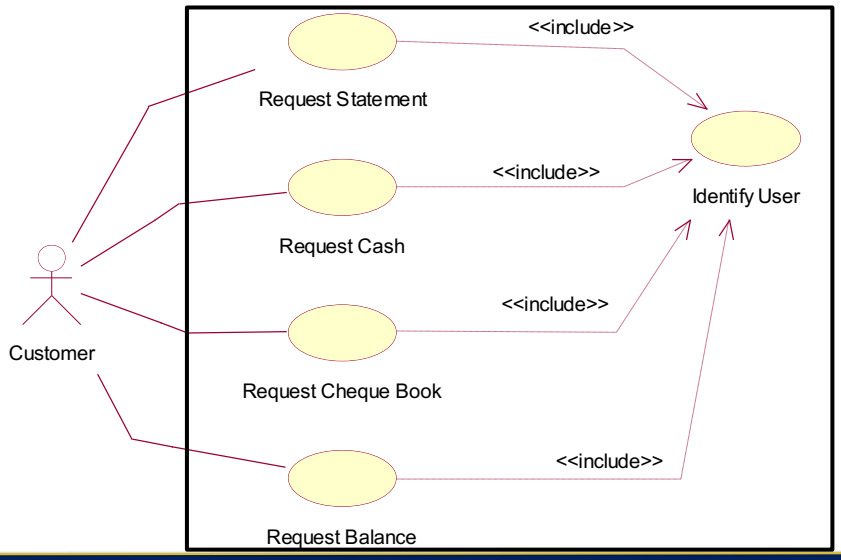
Use Case Relationships – Includes

- For commonality or replication between the steps involved in the execution of one or more use-cases. For example, consider ATM use cases
  - Request Cash
  - Request Balance
  - Request Statement
  - Request Cheque Book

User Identification
Insert their ID card
Enter their PIN
Verification

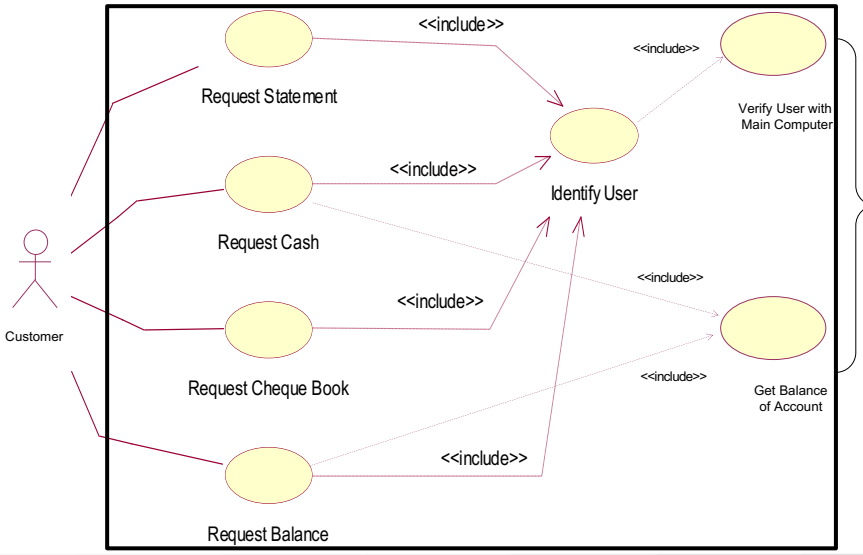
- Represent it with a mini-use-case called 'identify user' whose functionality is included as part of the other four use-cases.
- An 'includes' relationship (dashed line) indicates dependency. The arrow points to the use-case that will be included

Use-Cases – Incudes



41

Use-Cases – Incudes



Verify User with main computer

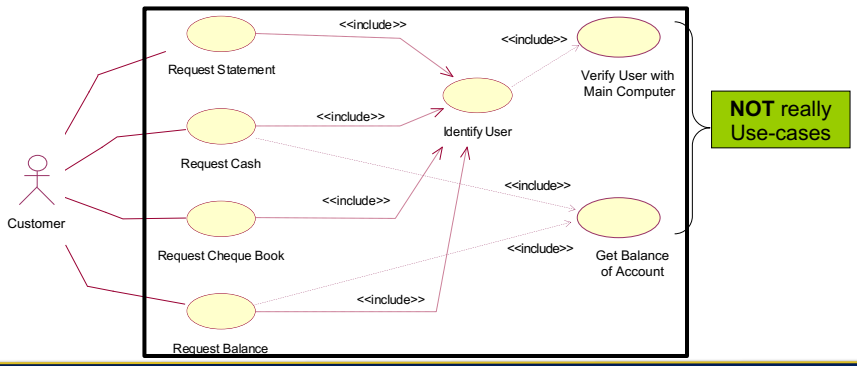
NOT really Use-cases

Get Balance of Account

42

Use-Cases – Incudes

- There is no user interaction in either use-case.
- There is no direct, immediate, measurable benefit to the user from the execution of that use-case.



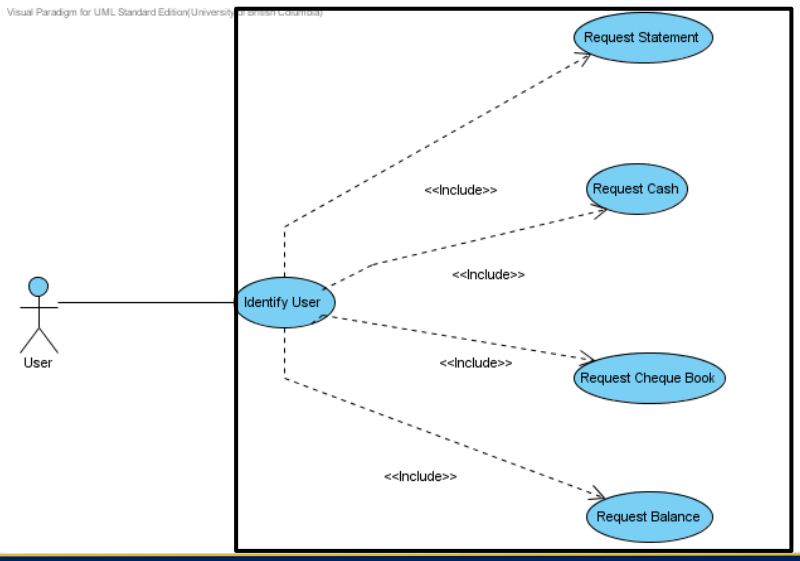
Documenting 'Includes' relationships in a Use-Case: Request Cash

- Start of Primary scenario/transaction
  - Include Identify User (a prerequisite or precondition for the execution of this use-case)
  - The user is prompted for the amount of the withdrawal. **note the assumption of success**
  - The user enters the amount of withdrawal.
  - The system checks the account balance with the banks central computer
  - If the user has insufficient funds <<Scenario 1>>
  - The cash is dispensed and the customer's account at the Bank Central Computer is debited with the withdrawal amount.
  - The card is returned to the user and a receipt issued.
- End-Of-Transaction

Documenting 'Includes' relationships in a Use-Case: Request Cash (Cont.)

- Scenario 1: 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, resume.
- Start of Included scenario/transaction
  - The user inserts their ID card into the system.
  - The system reads the magnetic strip from the card.
  - If identification fails << Scenario 1a>>
  - The system contacts the banks central computer to request the PIN number for the card and their account details.
  - If bank central computer cannot access users account <<Scenario 2>>
  - If PIN cannot be authenticated <<Scenario 3>>
- End-Of-Transaction
- Scenario 1a: 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.

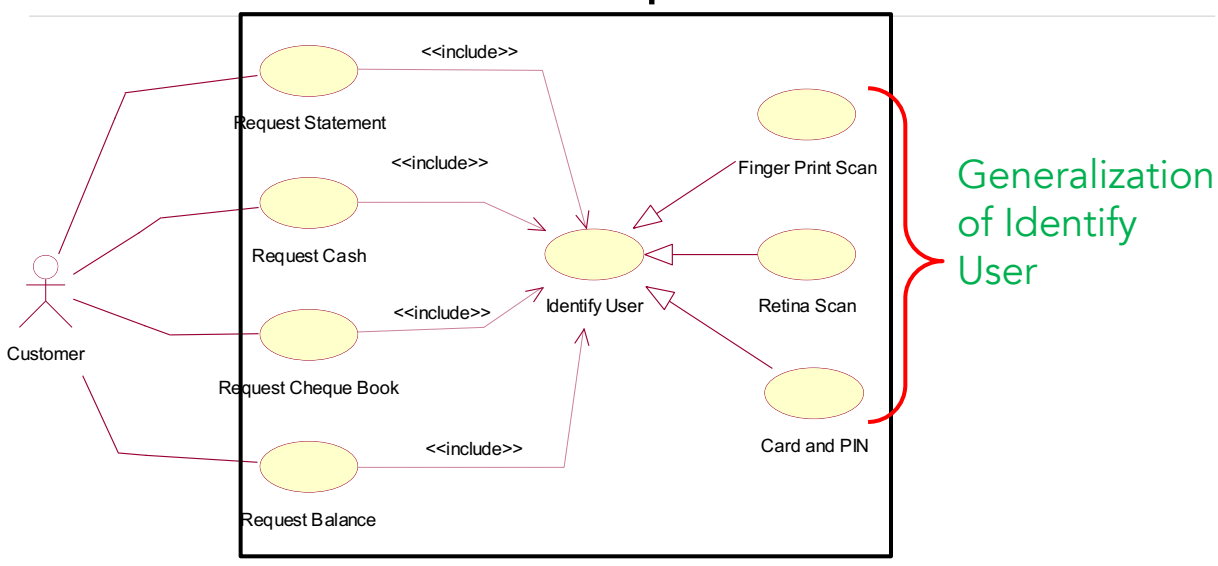
What's wrong with the following?



Use-Case Relationships – Generalization

- Generalization occurs when two of more use-cases attempt to achieve the same goal or objective but achieve it via different means.
- For example, identification in ATM
  - ID card with a magnetic strip were not the only way that you could identify yourself to a ATM.
  - Fingerprint
  - Retina scan
- Interaction with the user would vary but the outcome/benefits to the user would be the same

Use-Case Relationships – Generalization



Documenting Generalizations

- When documenting generalisation use cases the base or root use-case (i.e. Identify User) should be documented in very general terms

  Use-case: "Identify the user"
  obtain their account details from the bank central computer

- Derived use-cases can be documented with specific details.
- The various specific details of achieving that can be documented in the derived (real) use-cases.

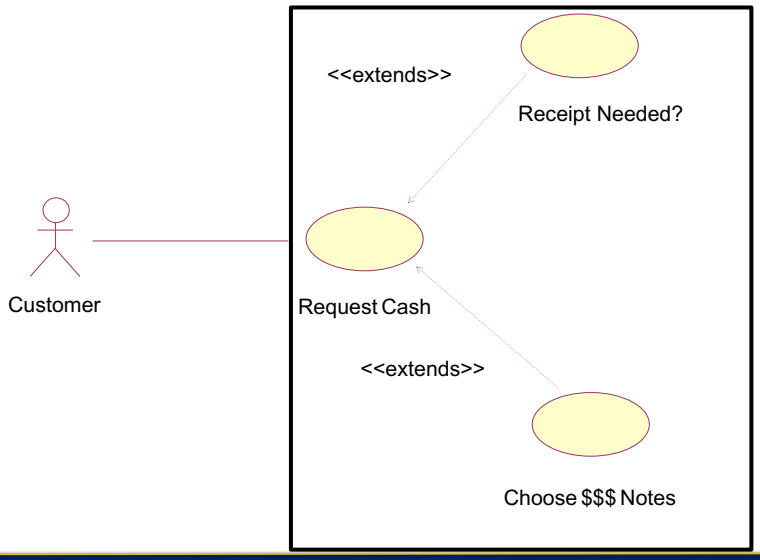
Use-Case Relationships – Extends

- Extends allows extensions to be made to a use-case.
  - Used to model optional behaviour, particularly interesting and important scenarios within a use-case.

- For example, in the ATM, the use-case 'Request Cash'
  - Different ways to get money

Documenting Extends relationship
Add mini-use cases and add an extends relationship to their parent or containing use-case

Use-Case Relationships – Extends



51

Documenting a Use-Case with Extends

Use-Case Request Cash

. . .

If(users wants to chose type of $ notes)

    Extends Choose $$$ Notes

If user chooses to have a receipt

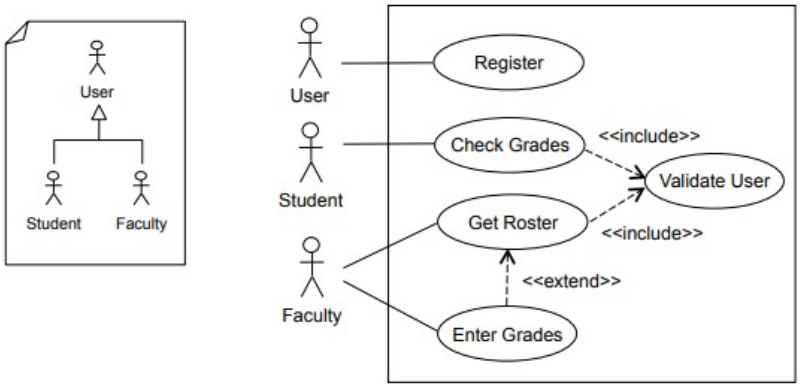
    Extends Print Receipt

. . .

The card is returned to the user and a receipt is issued.

52

Generalization of Actors



Summary

- Use Case Diagrams
  - Used to gather the requirements of a system.
  - Used to get an outside view of a system.
  - Identify the external and internal factors influencing the system.
  - Show the interaction among the requirements are actors.

Documenting Use-Cases

- Description
  - Functional flow of the system.
- Actors
- Preconditions, Postcondition
  - Must be true to your systems to be able to start/stop
- Goals

Use-Case Example

<table>
<tr><td colspan="2">Use Case: Check Grades</td></tr>
<tr><td colspan="2">Description: View the grades of a specific year and semester
Actors: Student
Precondition: The student is already registered
Main scenario:</td></tr>
<tr><td>User</td><td>System</td></tr>
<tr><td>3. The user enters the year and semester, e.g., Fall 2013.</td><td>1. The system carries out “Validate User”, e.g., for user “miner” with password “allAs”.
2. The system prompts for the year and semester.
4. The system displays the grades of the courses taken in the given semester, i.e., Fall 2013.</td></tr>
<tr><td colspan="2">Alternative:
The student enters “All” for the year and semester, and the system displays grades of all courses taken so far.
Exceptional:
The “Validate User” use case fails; the system repeats the validation use case.</td></tr>
</table>

Thank you!