Software Requirements & Use cases

Lecture 03 Requirements & Use Cases

Course Instructor: **SYED SAQLAIN HASSAN**

Course Code: SE321

Spring 2023

Inception Phase in Unified Process

- Inception is to:
  - **Envision the product scope, create the vision, and develop a business case**.
    - *Do the stakeholders have basic agreement on the vision of the project, and is it worth investing in serious investigation?*
- Inception is a very brief phase (3-4 weeks)
- The intent of inception is to establish some initial common vision:
  - for the objectives of the project
  - determine if it is feasible and
  - decide if it is worth serious investigation in elaboration

- Some artifacts that start in Inception

Artifact	Comment
Vision and Business Case	Describes the high-level goals and constraints, the business case, and provides an executive summary.
Use-Case Model	Describes the functional requirements, and related non-functional requirements.
Supplementary Specification	Describes other requirements.
Glossary	Key domain terminology.
Risk List & Risk Management Plan	Describes the business, technical, resource, schedule risks, and ideas for their mitigation or response.
Prototypes and proof-of-concepts	To clarify the vision, and validate technical ideas.

Requirements

- Requirements are the capabilities & conditions that the system (project or product) must provide and meet.
- Requirement issues are the leading cause of project failure.
  - Even if you do a perfect job of building the wrong thing, its no good!
- *Unified process realizes that change is constant, so plans for change instead of setting an impossible goal.*

- The key activities and resulting artefacts are :
  - List candidate requirements
    - Artefact: **Feature list**
  - Understand system context
    - Artefact: Business or domain model
  - Capture functional requirements
    - Artefact: Use-case model
  - Capture non-functional requirements
    - Artefact: Supplementary requirements or individual use cases (if specific)

Types of Requirements

- In the UP, requirements are categorized according to the FURPS+ model

Functional	• features, capabilities, security.
Usability	• human factors, help, documentation.
Reliability	• frequency of failure, recoverability, predictability.
Performance	• response times, throughput, accuracy, availability, resource usage.
Supportability	• adaptability, maintainability, internationalization, configurability.

Supplementary Requirements

- The "+" in FURPS+ indicates supplementary and sub-factors, such as:

Implementation	• resource limitations, languages and tools, hardware, …
Interface	• constraints imposed by interfacing with external systems.
Operations	• system management in its operational setting.
Packaging	• for example, a physical box.
Legal	• licensing and so forth

Usecase

Identifying & Writing Usecase!!

Use case

- A use case is a written description of how users will perform tasks on the proposed software.
  - from a user’s point of view, a system’s behavior as it responds to a request.
- Each use case is represented as a sequence of simple steps, beginning with a user's goal and ending when that goal is fulfilled.
- *A use case is a series of related interactions between a user and a system that enables the user to achieve a goal.*

Benefits of Use Cases

- Use cases add value because they help explain how the system should behave and in the process, they also help brainstorm what could go wrong.
- They provide a list of goals and this list can be used to establish the cost and complexity of the system.
- Project teams can negotiate which functions become requirements and are built.

- What Use Cases Include
  - Who is using the software
  - What the user want to do
    - The user's goal
  - The steps the user takes to accomplish a particular task
  - How the software should respond to an action
- What Use Cases Do NOT Include
  - Implementation-specific language
  - Details about the user interfaces or screens.

- Primary purpose of Use Case is to capture the required system behavior from the perspective of the end user, to achieve one or more goals.

- Different users benefit in different ways, of course:
  - Customers use them to describe, or at least to approve, the description of the system's behavior.
  - Potential users use them to understand the system's behavior.
  - Architects use them to identify architecturally significant functionality.
  - Developers use them to understand the required system behavior so they can identify classes from the Use Cases' flow of events.
  - Testers use them as a basis for identifying a subset of the required Test Cases.
  - Managers use them to plan and assess the work for each iteration.
  - Technical writers use them to understand the sequence of system behavior that they need to describe in the documentation.

An Example

- A Human Resources software is used to keep track of employee information at company XYZ.

- Use case : Add an employee
  - User selects "Employees > Add New" from the menu.
  - System displays "Add Employee" screen.
  - User enters employee information into the screen (First name, Last name, Phone, Email, Job title, Department).
  - User clicks 'Save' button.
  - System saves the employee information into database, and displays "Success" message.

Identifying &writing Usecases

Identifying Use Cases

- A use case describes the system's behavior as it responds to a series of related requests from an actor.
- The basic procedure to identify use cases are :
  - Step 1: Choose the System boundary
  - Step 2: Identify the primary actors
  - Step 3: Identify Actor (User) Goals
  - Step 4: Define use cases

Step1: Choosing the System Boundary

- Everything outside of the system is outside the system boundary.
  - Cashier, Payment authorization services etc. are outside the POS system (SuD)
- Once the actors are identified, the boundary becomes clearer.

Step2: Identify Actors

- Actors are external to the system and interacts with the system.
  - Actors may be other systems
  - Actor may an Individual, as an actor is a role.
    - Buyer, seller
- Users:
  - Direct users
  - Users responsible to operate and maintain it
  - External systems used by the system
  - External systems that interact with the system

Types of Actors

- Primary Actor
  - Has goals to be fulfilled by system
    - e.g. roles that people play: the cashier, customer.
- Supporting Actor
  - Provides service to the system
  - Often a computer system (or organization/person)
- Offstage Actor
  - Interested in the behavior, but no contribution
  - not primary or supporting
    - e.g, a government tax agency

Why identify?

Primary actors

To find user goals, which drive the use cases.

Supporting actors

To clarify external interfaces and protocols.

Offstage actors

To ensure that all necessary interests are identified and satisfied.

Step3: The Actor-Goal List

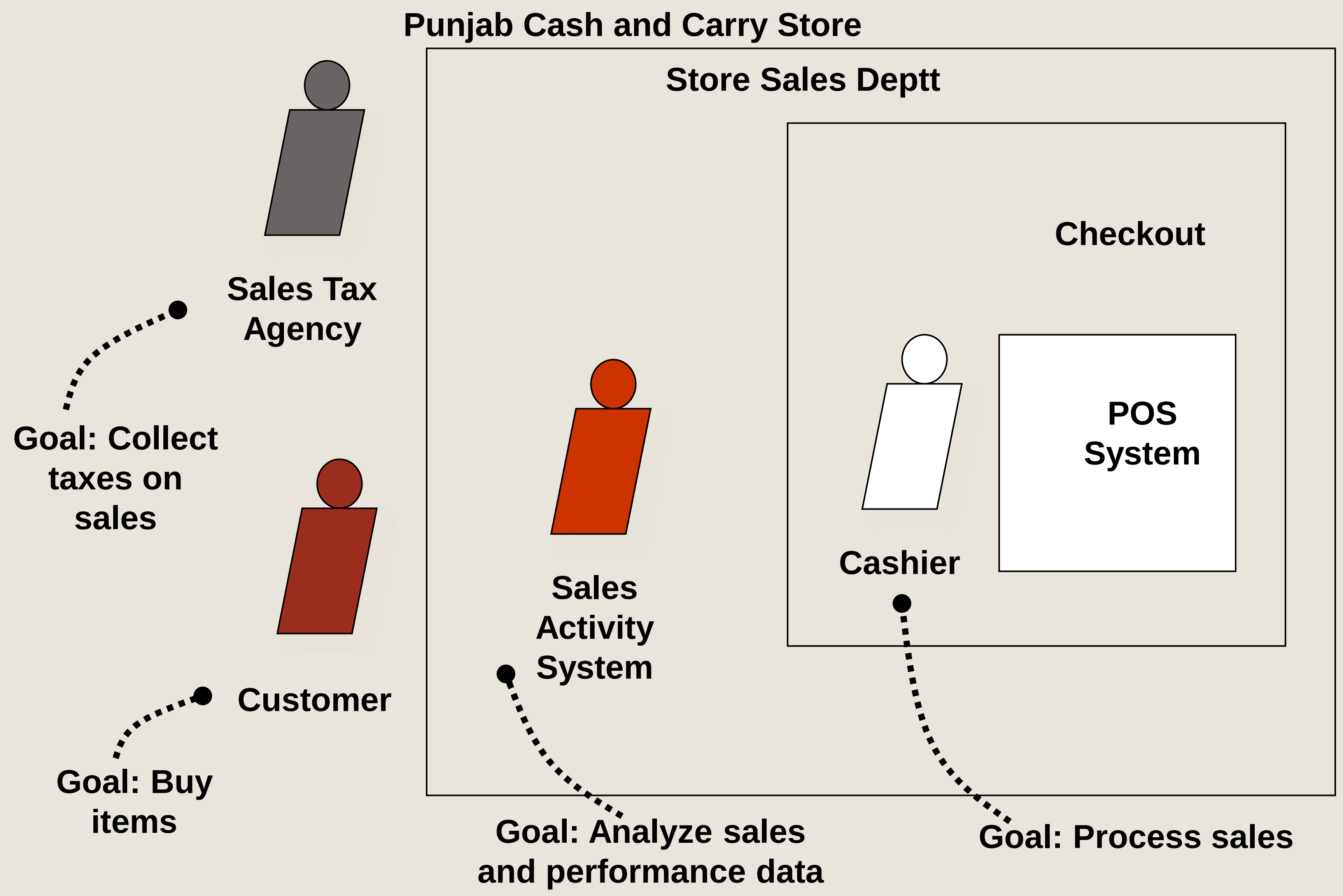
- Record the primary actors and their user goals in an actor-goal list.
  - Vision document has a section for this

Actor	**Goal**
Cashier	1. Process sales 2. Process rentals 3. Handle returns 4. Cash in 5. Cash out …
Manager	1. Start up 2. Shut down
System Administrator	1. Add users 2. Modify users 3. Delete users 4. Manage security 5. Manage system tables
Sales Activity System	1. Analyze sales and performance data

- Primary Actor and User Goals depend on System Boundary
- Use case Initiator or System user

- Questions to ask
  - *Why is the cashier, and not the customer, the primary actor in the use case Process Sale?*
  - *Why is the customer not listed in the actor-goal list?*

The answer depends on the system boundary of the SuD


Punjab Cash and Carry Store
Store Sales Deptt
Checkout
Sales Tax Agency
Goal: Collect taxes on sales
Customer
Goal: Buy items
Sales Activity System
Goal: Analyze sales and performance data
Cashier
POS System
Goal: Process sales

Actors and Goals via Event Analysis

- Another approach to aid in finding actors, goals, and use cases is to identify external events.
  - Identify the external events that a system must respond to.
  - Relate the events to actors and use cases.

External Event	From Actor	Goal
Enter sale line item	Cashier	Process a sale
Enter payment	Cashier or Customer	Process a sale
…………….		

Step4: Define Use Cases

- Tasks can be grouped at many levels of granularity from one or a few small steps to enterprise level activities
- For requirements analysis for a computer application, focus on use cases at the level of elementary business processes (EBPs).
  - EBP is a term from the business process engineering field:
- A task performed by one person in one place at one time, in response to a business event, which adds measurable business value and leaves the data in a consistent state. e.g., Approve Credit or Place Order.

- An EBP-level use case usually is composed of several steps, not just one or two.
  - It is usually useful to create separate "sub" use cases representing subtasks within a base use case.
    - e.g. Paying by credit
- Exception to this is to group CRUD (Create, Read, Update, Delete) operations into one use case.
  - Manage Users
- All systems have a Start Up and Shut Down use case (perhaps trivial and low level).
  - But sometimes, important.
    - an avionics system

- Which of these is a valid use case?
    - Negotiate a Supplier Contract
    - Handle Returns
    - Log In
  - All of these are use cases at different levels, depending on the system boundary, actors, and goals.
- A common use case mistake is defining many use cases at too low a level; that is, as a single step, sub-function, or subtask within an EBP.
  - Use cases can exist that fail the EBP test; many potentially exist at a lower level.

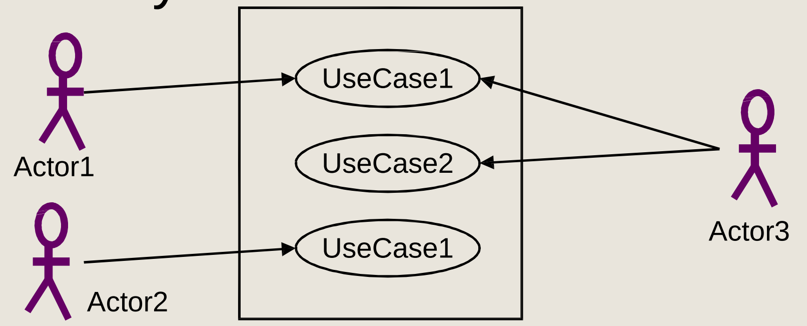
Use-Case Modeling

- User-centered development
  - A process of systems development based on understanding the needs of the stakeholders and the reasons why the system should be developed.

System concepts for Use-Case Modeling

Use-case diagram

- Depicts the interactions between the system and external systems and users.
- Graphically describes who will use the system and in what ways the user expects to interact with the system



System concepts for Use-Case Modeling

- **Use Cases**

  Needs three things

  Title – What is the goal? Short phrase with an active verb.

  Actor – Who desires it?

  Scenario – How is it accomplished?

System concepts for Use-Case Modeling

Use-case narrative

- A textual description of the business event and how the user will interact with the system to accomplish the task.
- Use active voice.

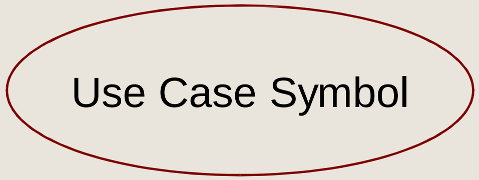
  eg. System validates Member-ID

System concepts for Use-Case Modeling

- Use Cases
  - A behaviorally related sequence of steps both automated and manual for the purpose of completing a single business task.
  - Describe system functions from the perspective of external users in a manner they understand.

System concepts for Use-Case Modeling

- Use Cases
  - Represented graphically by a horizontal ellipse with the name of the use case appearing above, below, or inside the ellipse.



System concepts for Use-Case Modeling

The collection of Use Cases for a system constitute all the defined ways the system may be used.

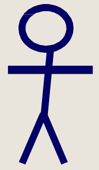
How to capture Use Cases?

By talking to typical users and discussing things that they might want to do with the system.

eg. Borrowing of books , Returning of books etc. (Library System)

System concepts for Use-Case Modeling

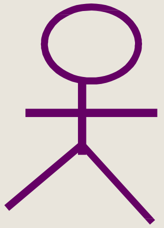
- Actors
  - Anyone or anything that needs to interact with the system to exchange information.
  - Represented graphically as a stick figure labeled with the name of the role the actor plays.


Actor Name

System concepts for Use-Case Modeling

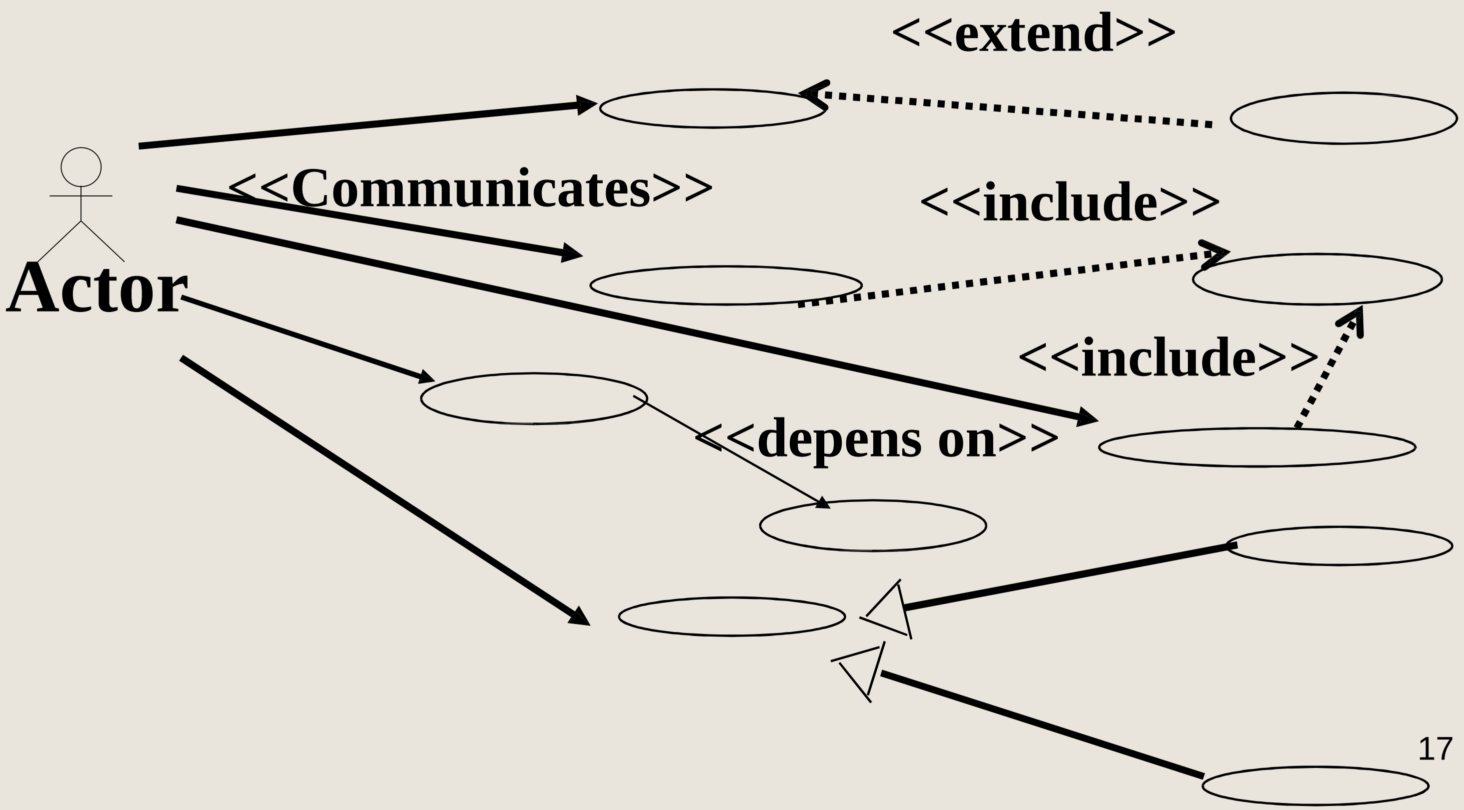
- Actors
  - An Actor may
    - Only input information to the system.
    - Only receive information from the system.
    - Input and receive information to and from the system.
  - Typically they are found in the problem statement and by conversations with customers and domain experts.

  eg. *Librarian* in a Library System



Use Case Diagram

Introduced by Jacobson (1994) to visualize use cases.



System concepts for Use-Case Modeling

- Relationships
  - Depicted as a line between two symbols on the use-case diagram
  - Meaning of the relationship differs depending on how the lines are drawn and type of symbols they connect.

System concepts for Use-Case Modeling

- Relationships
  - Associations (also called <<Communicates>>)
    - A relationship between an actor and a use case in which an interaction occurs between them
    - Modeled as a solid line connecting the actor and the use case
    - May be bidirectional or unidirectional

System concepts for Use-Case Modeling

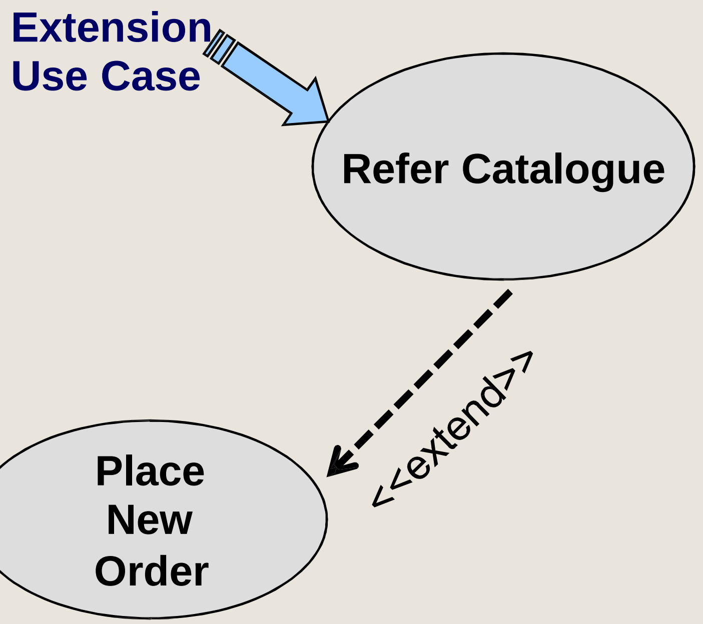
- Relationships
  - Extend

<<Extend>>

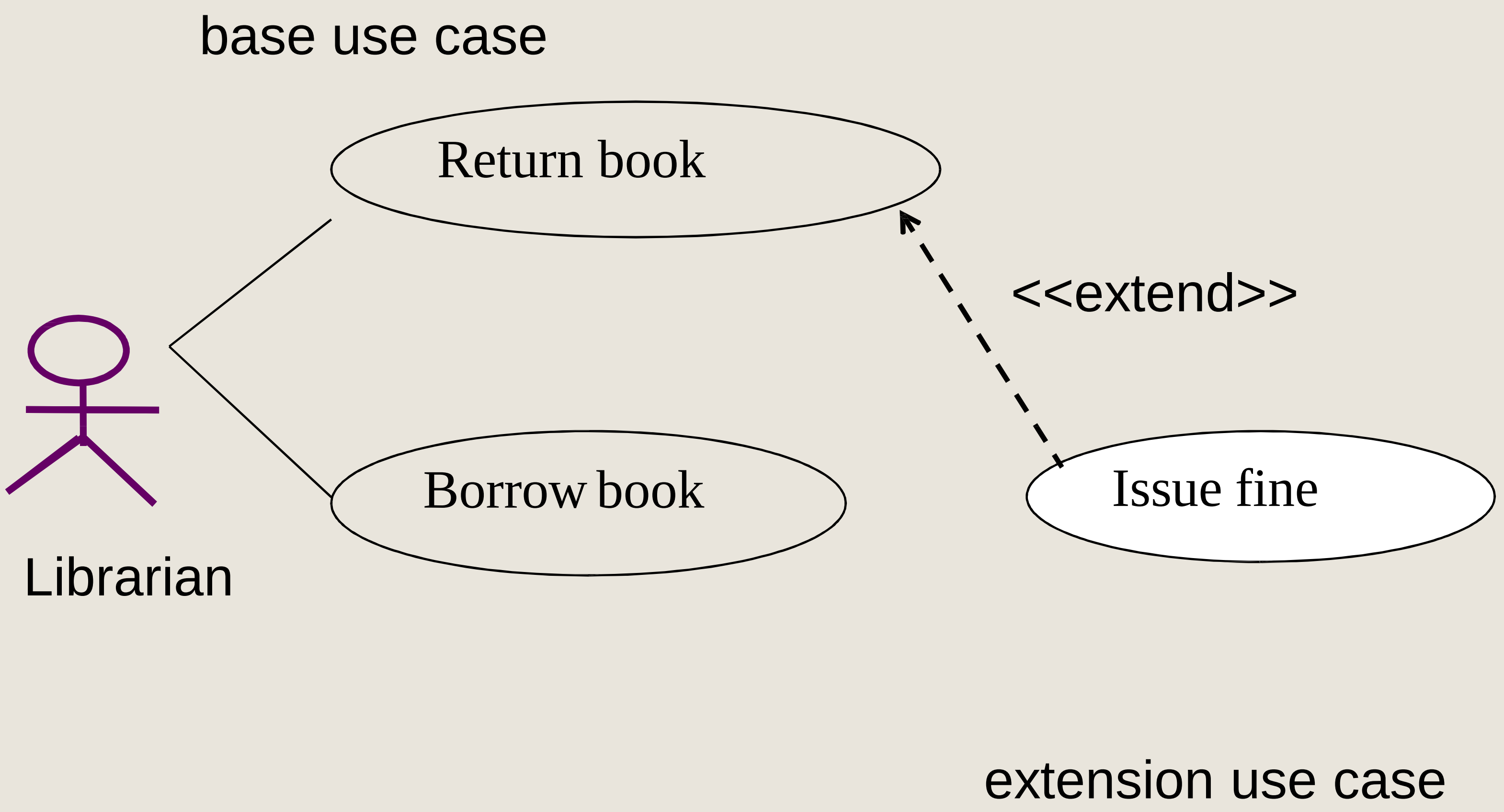
- A use case consisting of steps extracted from a more complex use case in order to simplify the original case and thus extend its functionality.
- The extension use case extends the functionality of the original use case.
- Shows optional behavior of a Use Case
- Depicted as an arrow headed line (either solid/dashed)

System concepts for Use-Case Modeling

- Relationships
  - Extend



System concepts for Use-Case Modeling



System concepts for Use-Case Modeling

- Relationships

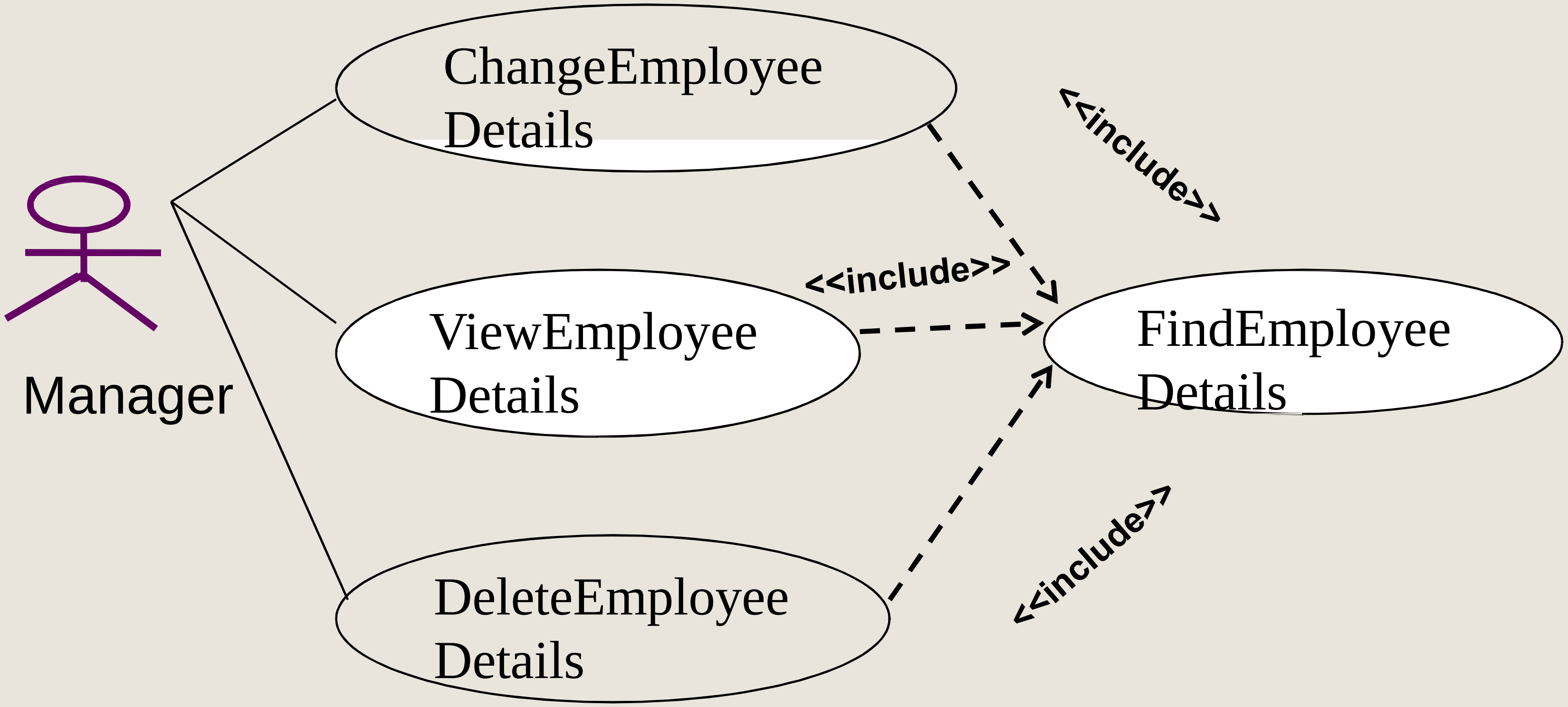
  <<include>>

  - Uses (or Include)
    - The base use case explicitly incorporates the behavior of another use case.
    - The relationship between the abstract use case and use case that uses it.

Abstract use case: a use case that reduces redundancy among two / more other use cases by combining the common steps found in those cases.

System concepts for Use-Case Modeling

- Relationships
  - Uses (or include)
    - Another use case uses or includes the abstract use case.

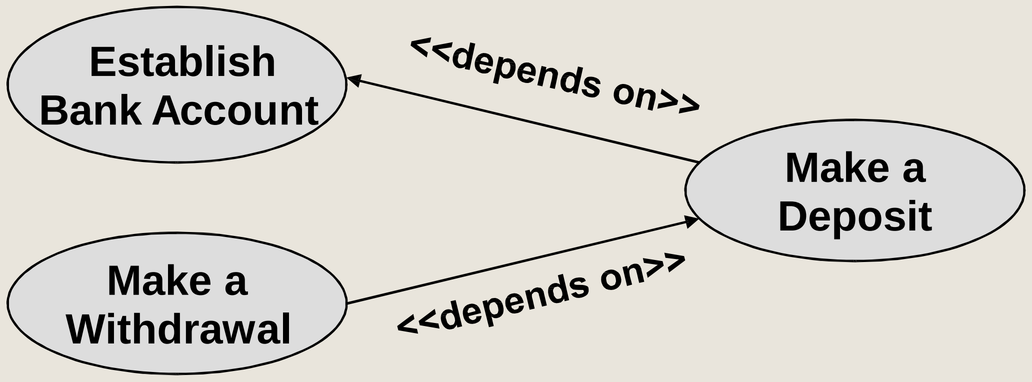


System concepts for Use-Case Modeling

- Relationships **<<depends on>>**
  - Depends on
    - A relationship between use cases indicating that one use case cannot be performed until another use case has been performed.
      - e. g. banking business – use case ‘ Make a Withdrawal’ cannot be performed until the use case ‘Make a Deposit’ has been executed.

System concepts for Use-Case Modeling

- Relationships
  - Depends on

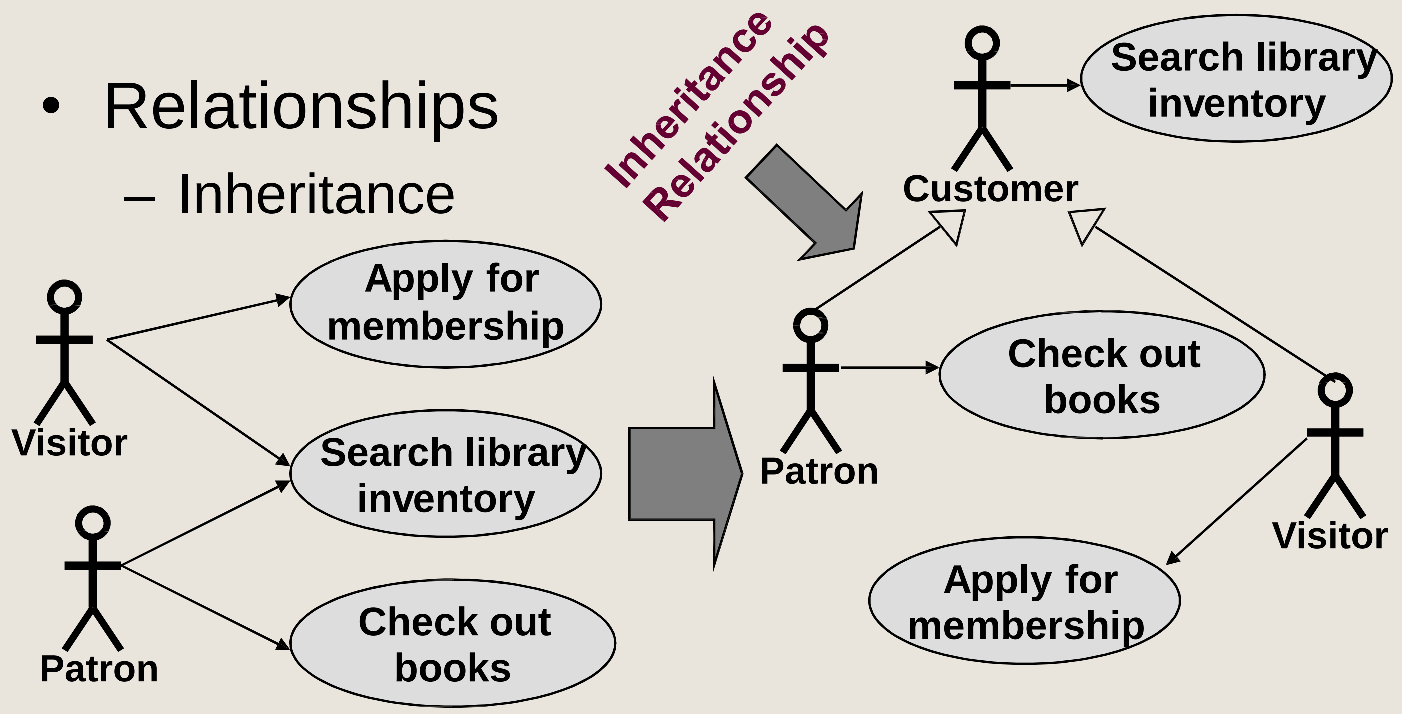


System concepts for Use-Case Modeling

- Relationships
  - Inheritance
    - A relationship between actors created to simplify the drawing when an abstract actor inherits the role of multiple real actors

System concepts for Use-Case Modeling

- Relationships
  - Inheritance



System concepts for Use-Case Modeling

- System Boundary
  - A Box drawn around the Use Cases to denote the boundary of the system being modeled.
  - Refer to as the **Subject** in UML 2.

The Process of Requirements Use-Case Modeling

- **Step 1: Identify business actors**
  - Concentrate on how the system will be used and not how it will be built.
  - Helps to refine and further define the scope and boundaries of the system.
  - Determine the completeness of the system requirements.
  - Actors should be named with a noun or noun phase.

The Process of Requirements Use-Case Modeling

- Where do you look for potential actors?
  - A context diagram that identifies the scope and boundaries of the system
  - Existing system documentation and user manuals.
  - Minutes of project meetings and workshops.
  - Existing requirements documents

In UML 2, actors may also represents other Subjects- gives you a way to link different Use Case models.

The Process of Requirements Use-Case Modeling

- When looking for actors, ask the following questions:
  - Who / What provides inputs to the system?
  - Who / What receives outputs from the system?
  - Are interfaces required to other systems?
  - Are there any events that are automatically triggered at a predetermined time?
  - Who will support and maintain information in the system?

The Process of Requirements Use-Case Modeling

- When looking for actors, ask the following questions:
  - Who is interested in a certain requirement?
  - Who will benefit from the use of the system?
  - Who will supply the system with this information, use this information, and remove this information?
  - Does one person play several different roles?
  - Does the system use an external resource?
  - Do several people play the same roles?

The Process of Requirements Use-Case Modeling

- **Step 2: Identify Business Requirements Use Cases**
  - An information system may consist of dozens of use cases.

    Use Case- A specification of sequence of actions

The Process of Requirements Use-Case Modeling

- **Step 2: Identify Business Requirements Use Cases**
  - Business requirements use case capture the interactions between a user and the system free of technology and implementation details
  - Use case describes how a real-world actor interacts with the system

The Process of Requirements Use-Case Modeling

- When looking for use-cases, ask the following questions:
  - What are the main tasks of the actor?
  - What information does the actor need from the system?
  - What information does the actor provide to the system?
  - Does the system need to inform the actor of any changes or events that have occurred?
  - Does the actor need to inform the system of any changes or events that have occurred?

System concepts for Use-Case Modeling

- When looking for use-cases, ask the following questions:
  - Will any actor create, store, change, remove, or read this information?
  - What use cases will create, store, change, remove, or read this information?
  - What use cases will support and maintain the system?
  - Can all functional requirements be performed by the use cases?
- Use cases are named with a verb phrase specifying the goal of an actor.
  Eg. Member Registration

The Process of Requirements Use-Case Modeling cont…

- **Step 3: Construct Use-Case Model Diagram**
  - Once use-cases and actors have been identified, a use-case model diagram can be used to show the system scope and boundaries.
  - If you have several sub systems, you may draw several use case model diagrams.

  Eg. Library System, Registration System, Examination System

The Process of Requirements Use-Case Modeling

- **Step 4: Document Business Requirements Use-case Narratives**
  - When preparing narratives,
    - First document them at a high level (can get aquick understanding of the events and magnitude ofthe system)
    - Then expand each use case separately to a fully documented business requirement narrative.

The Process of Requirements Use-Case Modeling

e.g. High Level Version of a use-case narrative

Author (s): ----------------------- Date:----------------------

Version: ------------------

Use-Case Name:	Use-Case Type Business Requirements:
Use-Case ID:	
Priority:	
Source:	
Primary Business Actor:	☐
Other Participating Actors:	
Other Interested Stakeholders:	
Description:	

Importance of the Use Case – typically high , medium , low

The Process of Requirements Use-Case Modeling

e.g. High Level Version of a use-case narrative

Author (s): ----------------------

Date:----------------------
Version: -----------------

Use-Case Name:	Use-Case Type Business Requirements:
Use-Case ID:	
Priority:	
Source:	
Primary Business Actor:	☐
Other Participating Actors:	
Other Interested Stakeholders:	
Description:	

Entity that triggers the creation of the Use Case. Eg. Document

The Process of Requirements Use-Case Modeling

e.g. High Level Version of a use-case narrative

Author (s): ----------------------

Date:----------------------
Version: -----------------

Use-Case Name:	**Use-Case Type** **Business Requirements:**
Use-Case ID:	
Priority:	
Source:	
Primary Business Actor:	☐
Other Participating Actors:	
Other Interested Stakeholders:	
Description:	

Who benefits from the use case

The Process of Requirements Use-Case Modeling

e.g. High Level Version of a use-case narrative

Author (s): ----------------------- Date:----------------------

Version: ------------------

Use-Case Name:	Use-Case Type Business Requirements: ☐
Use-Case ID:	
Priority:	
Source:	
Primary Business Actor:	
Other Participating Actors:	
Other Interested Stakeholders:	
Description:	

Facilitating Actors

The Process of Requirements Use-Case Modeling

e.g. High Level Version of a use-case narrative

Author (s): ----------------------

Date:---------------------
Version: -----------------

Use-Case Name:	**<u>Use-Case Type</u>**
Use-Case ID:	**Business Requirements:** ☐
Priority:	
Source:	
Primary Business Actor:	
Other Participating Actors:	
Other Interested Stakeholders:	
Description:	

General understanding of problem domain and scope

In brief

The Process of Requirements
Use-Case Modeling

e.g. Expanded Version of a use-case narrative

More details such as
- Preconditions
- Trigger
- Typical Course of Events
- Alternate Courses
- Post conditions

etc. are included.
(for more information Refer pg258 of Ref 1)

Typically another Use Case that must be previously executed.

The Process of Requirements Use-Case Modeling

e.g. Expanded Version of a use-case narrative

More details such as

- **Preconditions**
- **Trigger**
- **Typical Course of Events**
- **Alternate Courses**
- **Post conditions**

etc. are included.

Time receiving a cheque.

The Process of Requirements
Use-Case Modeling

e.g. Expanded Version of a use-case narrative

More details such as

- **Preconditions**
- **Trigger**
- **Typical Course of Events**
- **Alternate Courses**
- **Post conditions**

etc. are included.

eg. Borrowing : checkMember, checkOverdue, CheckOverLimit, checkCopyBorrowable , Confirm Borrowing

The Process of Requirements Use-Case Modeling

e.g. Expanded Version of a use-case narrative

More details such as
- **Preconditions**
- **Trigger**
- **Typical Course of Events**
- **Alternate Courses**
- **Post conditions**

etc. are included.

Errors, Confirm Messages

The Process of Requirements Use-Case Modeling

e.g. Expanded Version of a use-case narrative

More details such as

- **Preconditions**
- **Trigger**
- **Typical Course of Events**
- **Alternate Courses**
- **Post conditions**

etc. are included.

Receipt Delivered to the Customer

Use-Cases and Project Management

- Use-case model can be used to drive the entire system development effort.
- Once the use-case model is complete the project manager or system analyst uses the model to plan the build cycles of the project.
- To determine the importance of the use-cases, the project manager or system analyst will complete **a use-case ranking and evaluation matrix** and construct a **use case dependency diagram** with inputs from the stake holders and the development team.

Use-Cases and Project Management

- **Ranking and Evaluating Use Cases**
  - Most important use-cases should be developed first
  - ***Use-case ranking and priority matrix*** is a tool used to evaluate use cases and determine their priority

Use-Cases and Project Management

- ***Use-case ranking and priority matrix***
  - Completed with input from the stakeholders and the development team
  - Adapted from Craig Larman’s work
  - Evaluates use-cases on a scale of 1-5 against six criteria

    1. **Significant impact on the architectural design**
    2. **Easy to implement but contains significant functionality**
    3. **Includes risky, time-critical, or complex functions**
    4. **Involves significant research or new or risky technology**
    5. **Includes primary functions**
    6. **Will increase revenue or decrease cost**

Use-Cases and Project Management

Criteria

Use-case ranking and priority matrix : e.g.

Use Case Name	Ranking Criteria, 1 to 5						Total Score	Priority	Build Cycle
	1	2	3	4	5	6			
Submit subscription order	5	5	5	4	5	5	29	High	1
Place new order	4	4	5	4	5	5	27	High	2
Make product inquiry	1	1	1	1	1	1	6	Low	3
Establish new member subscription program	4	5	5	3	5	5	27	High	1
Revise order	2	2	3	3	4	4	18	Medium	2

Use-Cases and Project Management

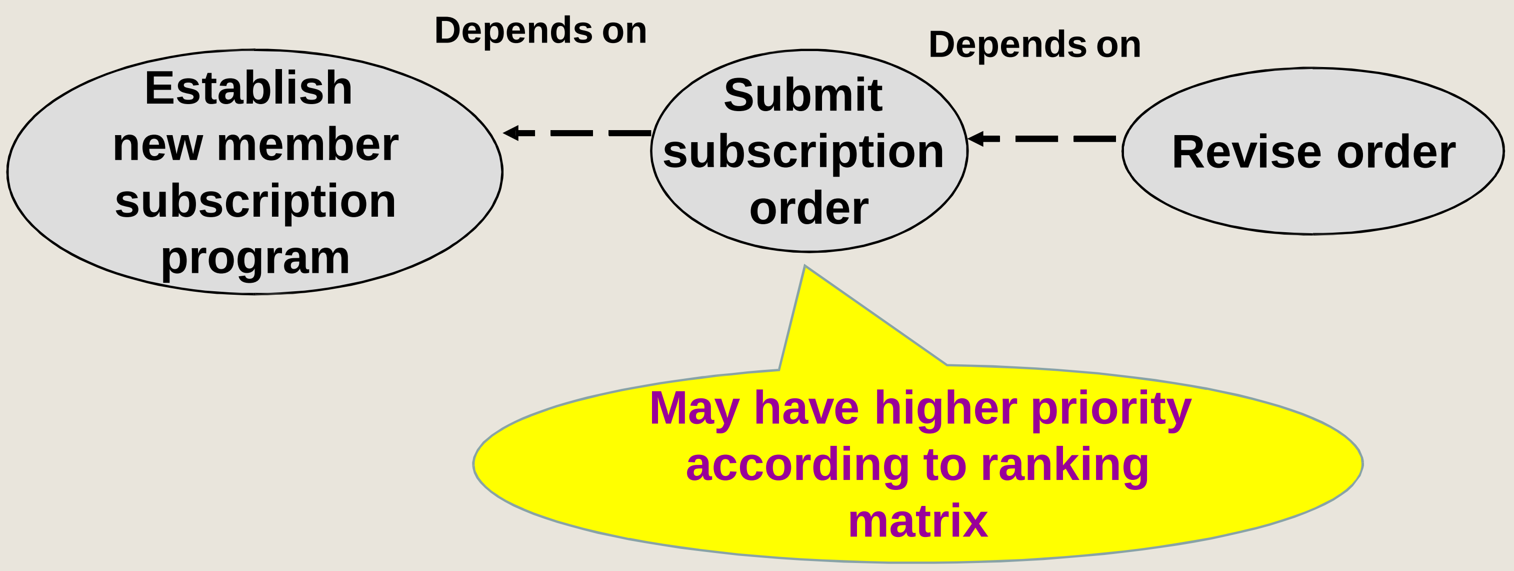
- **Identifying Use-Case dependencies**
  - Some use-cases may depend on other use-cases, with one use case leaving the system in a state that is a precondition for another use case.
  - Use-case dependency diagram is used to model such dependencies.

Use-Cases and Project Management

- **Use-case dependency diagram**
  - A graphical depiction of the dependencies among use-cases
  - Benefits
    - Enhances the understanding of system functionality
    - Helps to identify missing use-cases
    - Facilitate project management by depicting which use-cases are more critical and thus need to have a higher priority

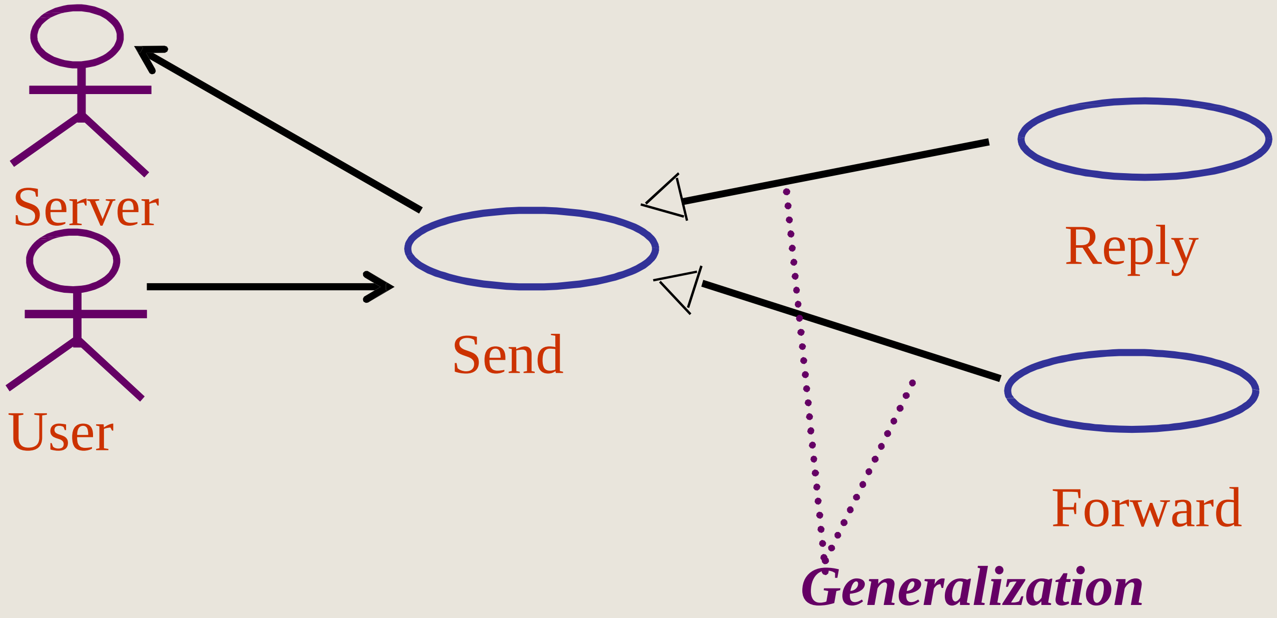
Use-Cases and Project Management

- Use-case dependency diagram: e.g.

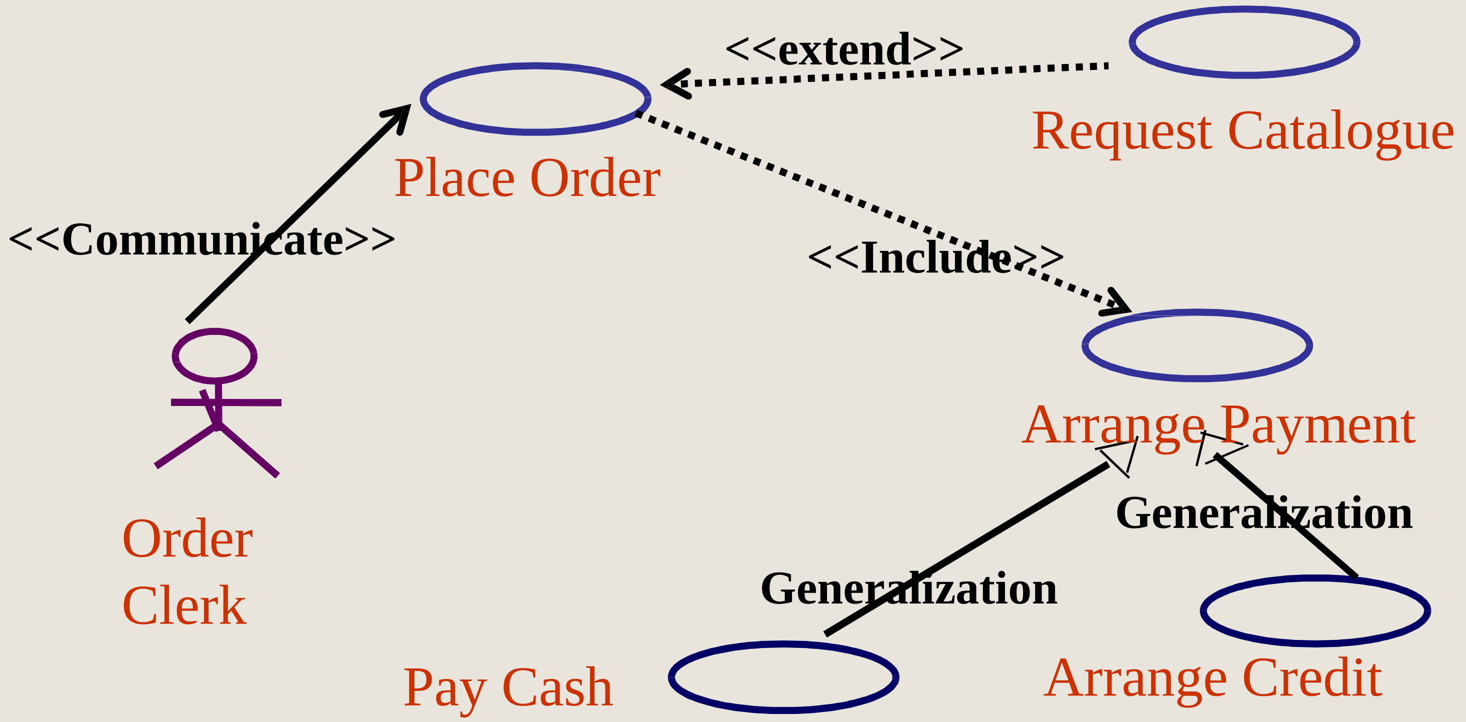

Depends on
Depends on
Establish new member subscription program
Submit subscription order
Revise order
May have higher priority according to ranking matrix

Eg. USE CASE Diagram

<u>User sending an</u>
<u>E-mail message</u>



Use Case diagram for an Order Processing Application.



General activities in performing Object Oriented Analysis

- Modeling the functions of the system
- Finding and Identifying the Business Objects
- Organizing the Objects and Identifying the Relationships

Modeling the functions of the System

- Document functional system requirements using business requirements use cases
  - Use Cases documented to contain only general information about the business event.
  - Goal was to quickly document all of the business events (use cases) in order to define and validate requirements.
- Construct the Analysis Use Case Model

Construct the Analysis Use Case Model

- Evolve Requirements Use Case model into Analysis Use Case Model by performing the following steps:
  - Identify, define and document new actors
  - Identify, define and document new Use Cases
  - Identify any reuse possibilities
  - Refine the Use case model diagram
  - Document System analysis use case narratives

Construct the Analysis Use Case Model

- Identify, define and document new actors
  - eg. Consider Use Case *Place New Order*

    Customer can place the order using Internet (identified initially)

    Customer also can submit orders by e-mail (Identified later)

    Service associates *input the order details sent through e-mail*

Construct the Analysis Use Case Model

- Identify, define and document new Use Cases
  - eg. New actor ***Service associates*** *need to a interact with the system – need to add a new Use Case*
- Identify any reuse possibilities
  - Two users can share common steps like in the place order example
  - Can extract the common steps into a separate Use Case. <<***include***>>

Construct the Analysis Use Case Model

– Refine the Use case model diagram (If necessary)
  - With the discovery of Actors and/or Use Cases , Use Case Model Diagram needs to be updated.

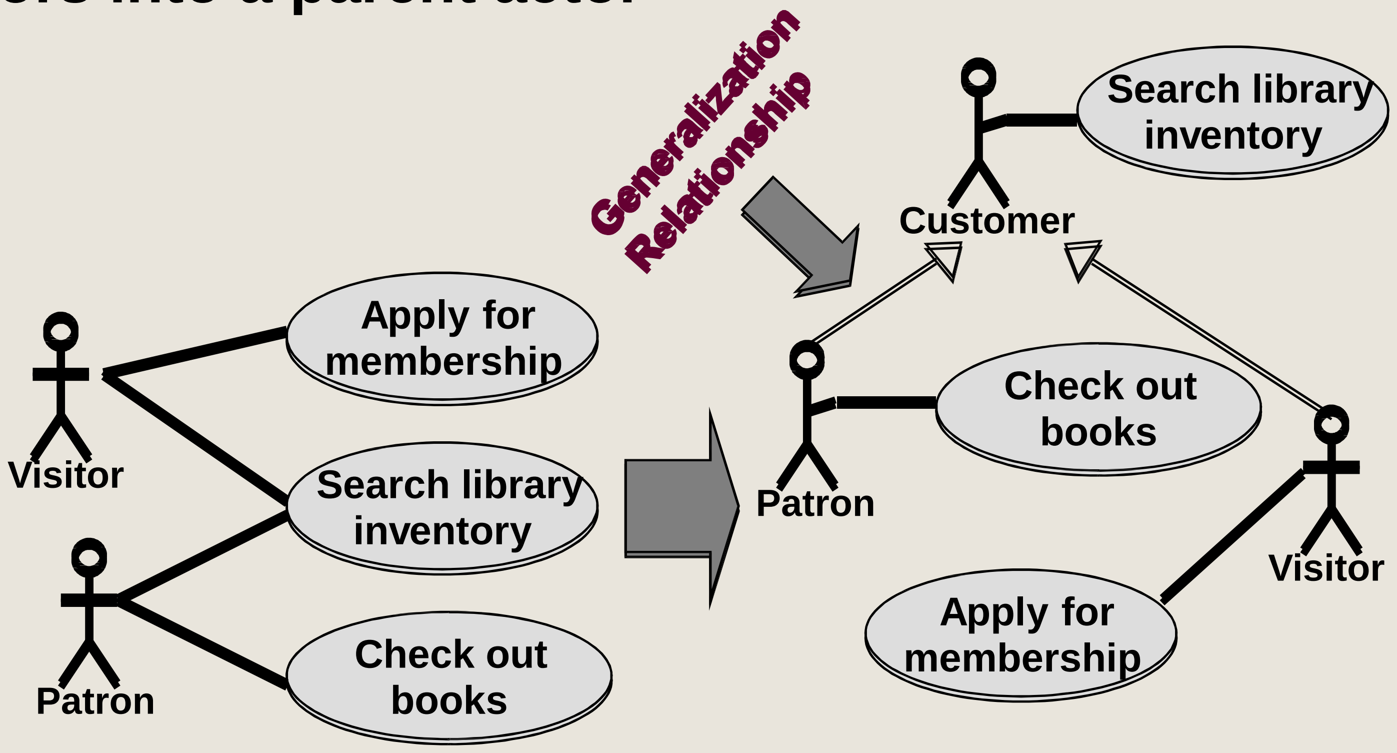
– Document System analysis use case narratives.

eg. Ref 1 page 386

Actor Generalization

We can simplify the model with Actor Generalization.

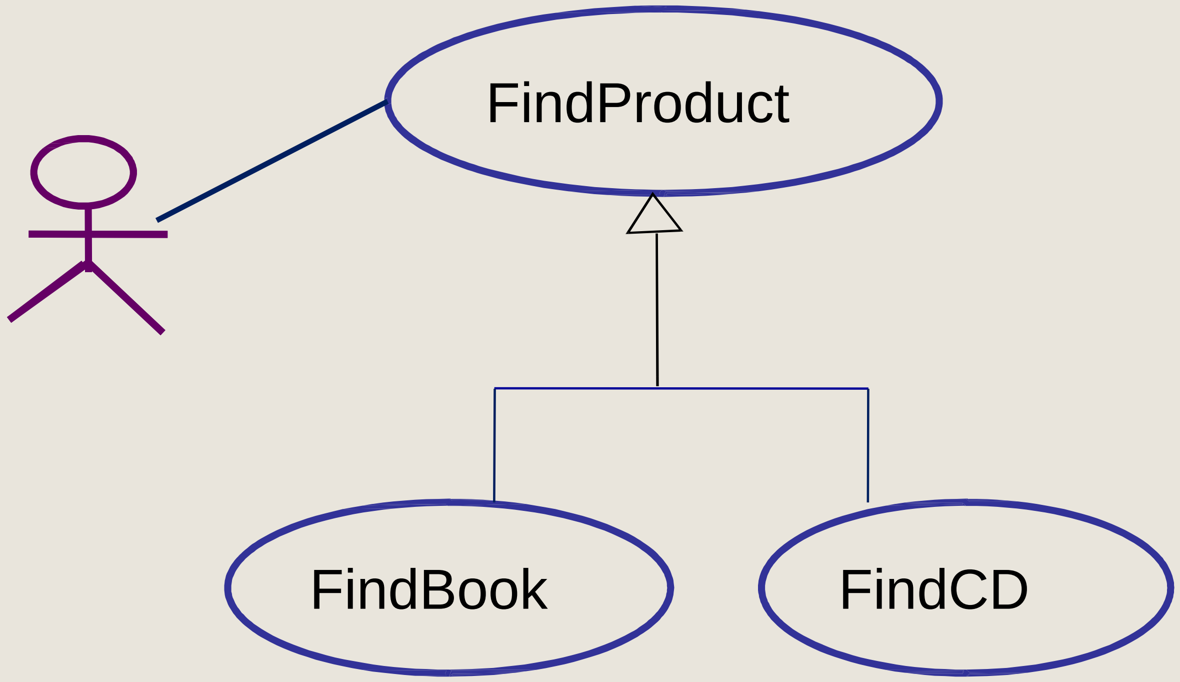
It factors out behaviour common to two or more actors into a parent actor



Use Case Generalization

- Factors out behaviour common to one or more use cases into a parent use case.
- In Use case generalization, the child use cases represent more specific forms of the parent. The children may:
  - Inherit features from their parent use case
  - Add new features
  - Override (change) inherited features

Use Case Generalization cont…



Specification for the parent Use case

Use case : Find Product **ID- 6**
Brief Description: The customer searches for product
Primary Actor : Customer
Secondary Actors : None
Preconditions : None
Main Flow : 1. The Customer selects **find product** 2. System asks for search criteria 3. The Customer enters the requested criteria 4. System srch for products that match the Cust. criteria. 5. If the system find some matching products 5.1 The sys displays a list of matching products 6. Else 6.1 The sys tells customer “No matching Products”
Post Conditions, Alternative Flows : none

Specification for a child Use case

Use case : Find Book ID- 7 , Parent ID -6

Brief Description: The customer searches for a Book

Primary Actor , Secondary Actor: Customer

Overridden

Preconditions : None

Main Flow :

1. The Customer selects **“find book”**
2. System asks for book search criteria (title, ISBN, etc)
3. The Customer enters the requested criteria
4. System srch for books that match the Cust. criteria.
5. If the system find some matching books
   5.1 The sys displays a details of matching books (max 5)
   5.2 While there are more books, cust. Is given the option to display the next page of books.
6. Else
   6.1 The sys tells customer “No matching Products”

Post Conditions, Alternative Flows : none

Specification for a child Use case

Use case : Find Book ID- 7 , Parent ID -6

Brief Description: The customer searches for a Book

Primary Actor , Secondary Actor: Customer

Preconditions : None

Main Flow :

1. The Customer selects "final book"
2. System asks for book search criteria (title, ISBN, etc)
3. The Customer enters the requested criteria
4. System srch for books that match the Cust. criteria.
5. If the system find some matching books
   5.1 The sys displays a details of matching books (max 5)
   5.2 While there are more books, cust. Is given the option to display the next page of books.
6. Else
   6.1 The sys tells customer "No matching Products"

Post Conditions, Alternative Flows : none

Add new features

Inherited without change

Use Case Generalization cont…

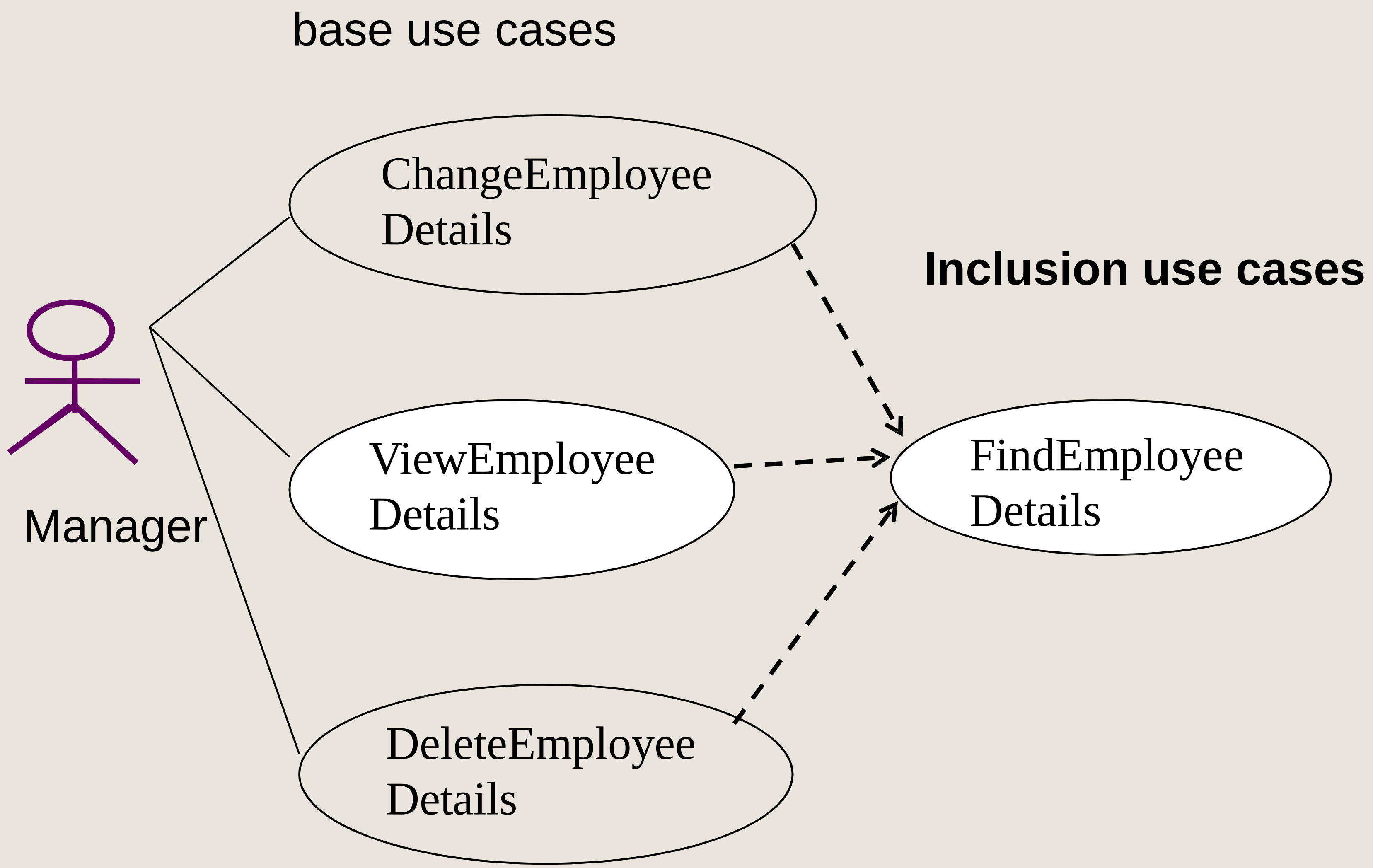
- Use case models are used to communicate with stakeholders.
- Users find it difficult to understand generalization features such as overriding etc.
- Most analyst restrict the parent use cases so that it doest not have any main flow, but only a brief description of its semantics.
- This approach makes use case generalization easy as the parent use case becomes an **abstract** use case.

Have a plain text summary of the high level behaviour that its children will implement.

Use Case Generalization cont…

- Since abstract use cases have missing or incomplete flow of events,
  - they will never be executed by the system
- You should only use Use Case generalization if it simplifies your use case model.

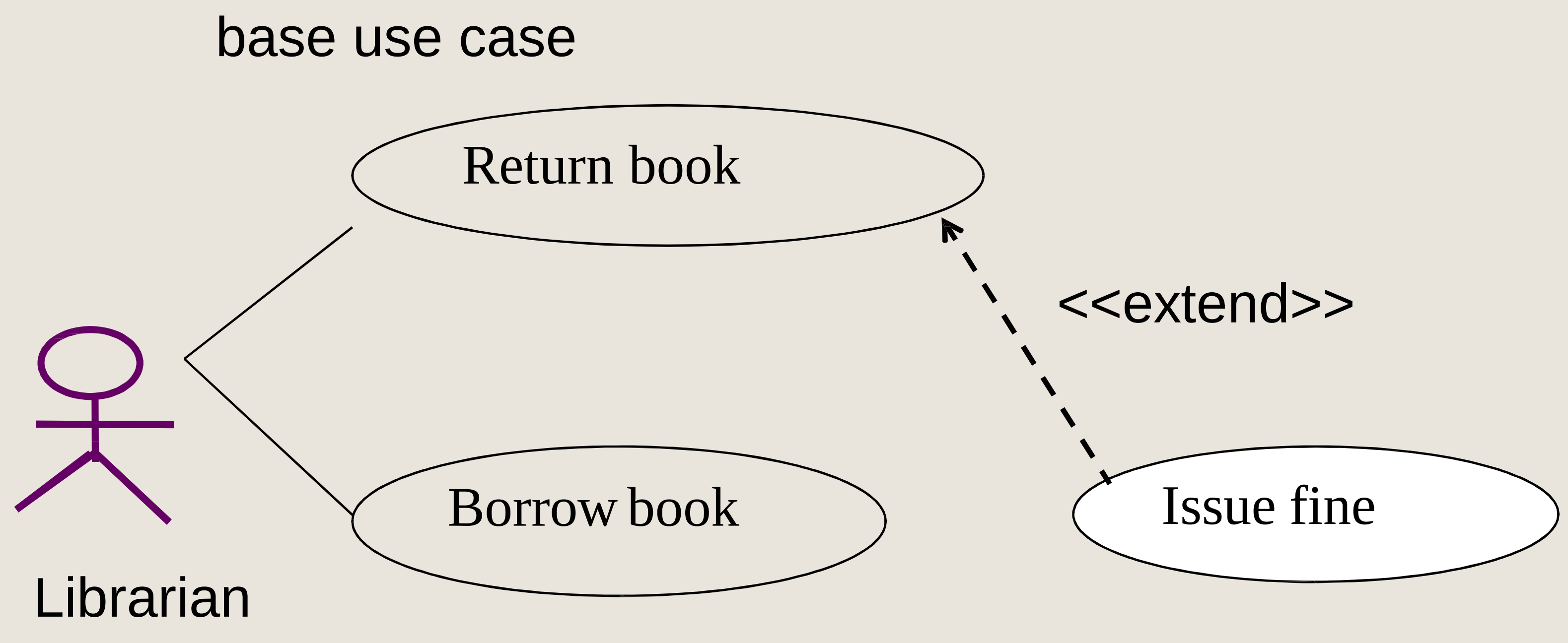
Example of <<include>> Relationship



Example of <<include>> Relationship cont..

- *Inclusion* use case supplies behavior to its base use case.
- The base use case executes until the point of inclusion is reached,
- Then execution passes over to the inclusion use case.
- When the inclusion use case finishes , the control return to the base use case again.

Example of <<extend>> Relationship

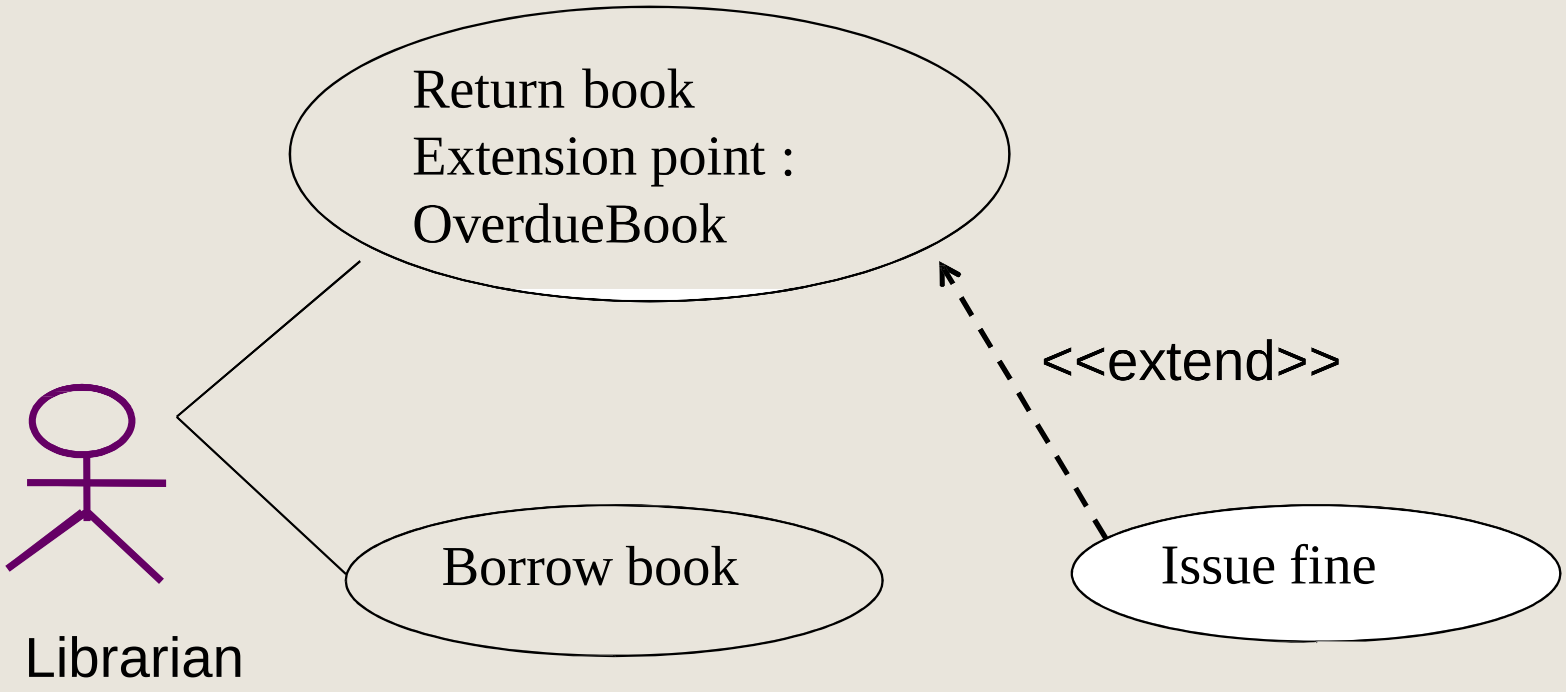


extension use case

Example of <<extend>> Relationship cont..

- <<extend>> provides a way to insert new behaviour into an existing use case.
- The base use case provides a set of extension points that are hooks where new behaviour is added.
- Extension use case provides a set of insertion segments that can be inserted into the base use case at these hooks.

Example of <<extend>> Relationship cont..

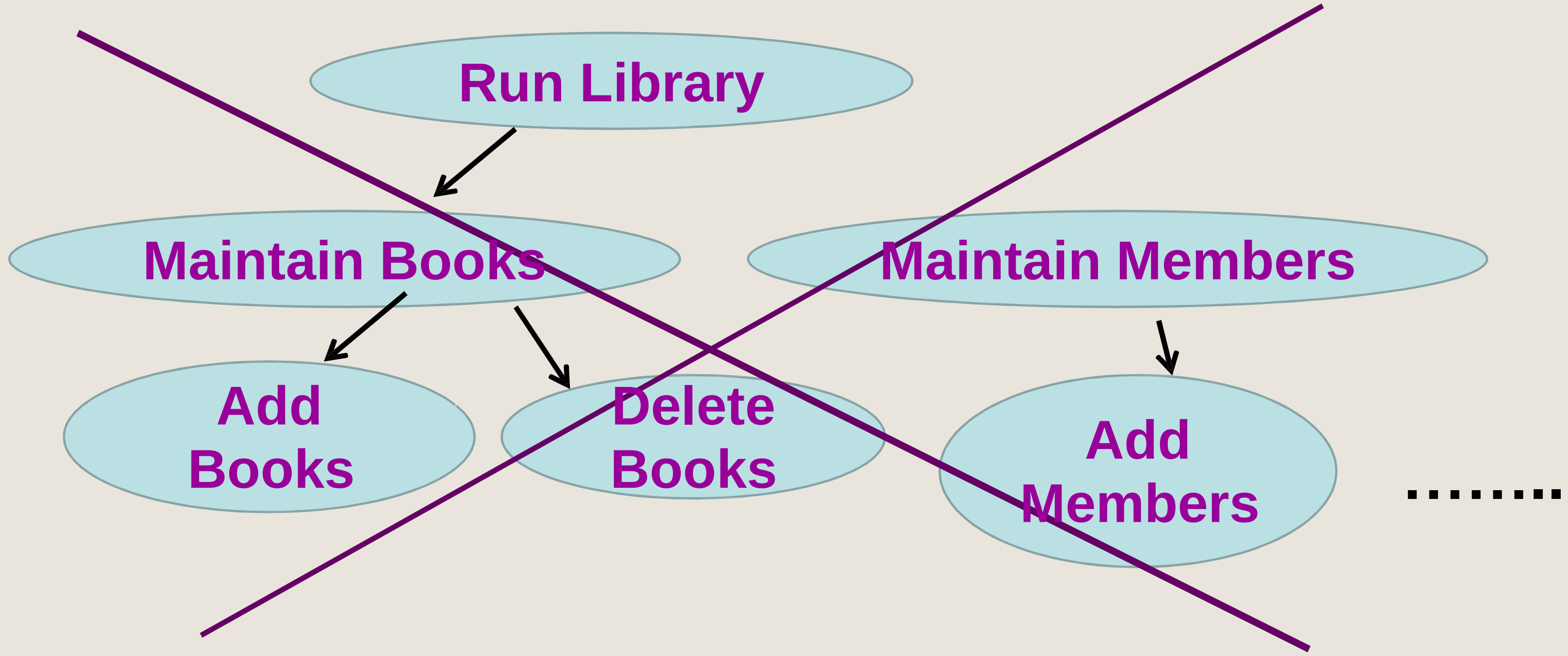


When to apply Use Case Modeling

- When a system is dominated by functional requirements.
- When the system has many type of users to which it delivers different functionality.
- The system has many interfaces.

Hints and Tips for writing Use Cases

- Keep use cases short and simple
- Focus on ***What*** and not *How*.
- Avoid Functional Decomposition



Thank you!!!

- You must read Chapter 5,6, 7, 8 Applying UML Pattern and Design, Craig Larman, 2nd Edition