



SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 of the UGC Act, 1956)



THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

Use Cases and Use Case Diagrams



Objectives

- To explain what use cases are
 - To present UML use case diagram notation
 - To suggest processes for creating use case diagrams
 - To present rules and heuristics for making use case diagrams.
 - To list use case diagram quality checks
 - To explain when to use the use case diagrams
-
- Built in early stages of development
 - Purpose
 - Specify the context of a system
 - Capture the requirements of a system
 - Validate a systems architecture
 - Drive implementation and generate test cases
 - Developed by analysts and domain experts



Topics

- Why use case diagrams work
- Use cases, actors, and scenarios
- UML use case diagram notations
- How to make use case diagrams
- Rules of formation
- Heuristics
- Quality checks



Why Use Case Modeling Works

- User-level requirements are generated from stakeholder needs.
- Operational-level requirements are hard to generate in isolation.
- Considering the interactions between a program and its environment
 - Organizes operational level requirements
 - Provides context for refinement
 - Help avoid inconsistencies, omissions, redundancies, etc.



What is a use case?

- A requirements analysis concept
- A case of a use of the **system/product**
- Describes the system's actions from a the point of view of a user
- *Tells a story without specifying the structure of the system*
 - A sequence of events involving
 - Interactions of a user with the system
 - Specifies one aspect of the behavior of a system,
 - Is oriented toward satisfying a user's goal



How do we describe use cases?

- Textual or tabular descriptions
- User stories
- Diagrams



USE CASE TEMPLATE
<PROJECT NAME>

Use Case Identification

Use Case ID: <Unique identifier for the Use Case>
Use Case Name: <Clear and concise Use Case name>

Use Case Version Control

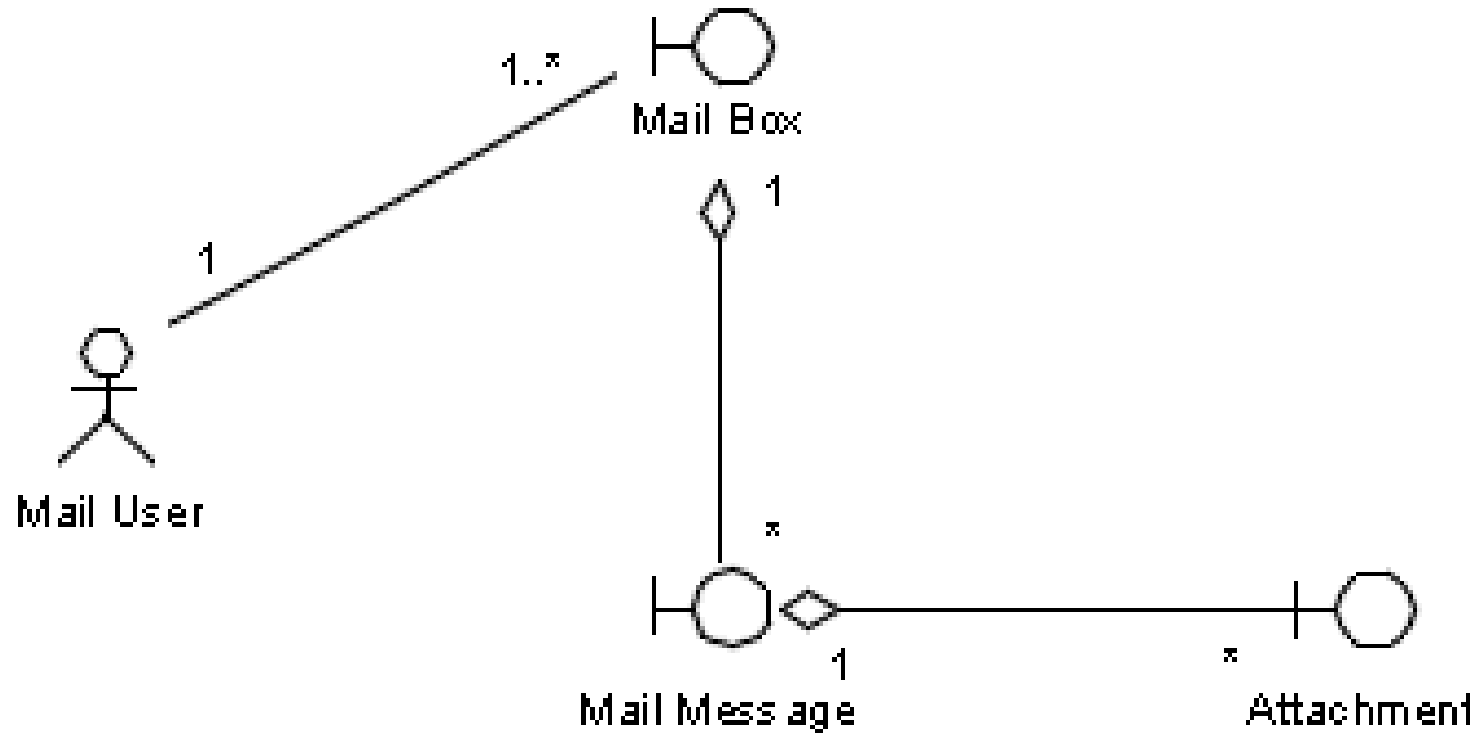
Version	Date Created	Originator	Changes Made

Use Case Definition

Description	<Brief description of the Use Case>	
Primary Roles	<Main actors involved in the Use Case>	
Pre-Conditions	<Assumptions or factors that need to be in place for the Use Case to start>	
Trigger	<Event that will trigger the start of the Use Case>	
Basic Flow	Step	Interaction
	1	<Brief sentence describing the step if all goes as planned>
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
Alternate Flows	10	
	Step	Interaction
	1	<Brief sentence describing the step if there are exceptions in the plan or alternative routes>
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
Post-Conditions	<Assumptions or factors that need to be in place for the Use Case to finish>	



use-case storyboard corresponding to the Manage Incoming Mail Messages use case:





Use Case Descriptions

- **actors** - something with a behavior or role, e.g., a person, another system, organization.
- **scenario** - a specific sequence of actions and interactions between actors and the system, a.k.a. a *use case instance*
- **use case** - a collection of related success and failure scenarios, describing actors using the system to support a goal.



What is an Actor?

- Include all user roles that interact with the system
- Include system components only if they are responsible for initiating/triggering a use case.
 - For example, a timer that triggers sending of an e-mail reminder
- **primary** - a user whose goals are fulfilled by the system
 - **importance:** define user goals
- **supporting** - provides a service (e.g., info) to the system
 - **importance:** clarify external interfaces and protocols
- **offstage** - has an interest in the behavior but is not primary or supporting, e.g., government
 - **importance:** ensure all interests (even subtle) are identified and satisfied



Use Cases and Actors

A **use case** is a type of complete interaction between and product and its environment.

An **actor** is a type of agent that interacts with a product.

- The collection of use cases characterizes all externally observable behavior.
- A use case is an entire, coherent interaction.
- Actors are roles not individuals.
- The product is *never* an actor.



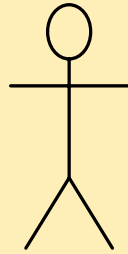
Use Case Diagrams

A **use case diagram** represents a product's use cases and the actors involved in each use case.

- Static model
- Catalog of all use cases
- UML diagram



Use Case Diagram Symbols

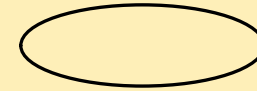


Actor Name

Actor Symbol



Association Line



Use Case Name

Use Case Symbol



Scenarios

A **scenario** is an interaction between a product and particular individuals.

- Instance of a use case
- Help envision how a product can be used
- Abstracted into use cases



A Scenario for Car Wash

Michael drives to the **kiosk outside the carwash** and attempts to insert five dollars.

The kiosk accepts the money and displays the message “You have paid enough for a complete wash.

Insert another dollar for a deluxe wash.” Michael presses the button for an express wash.

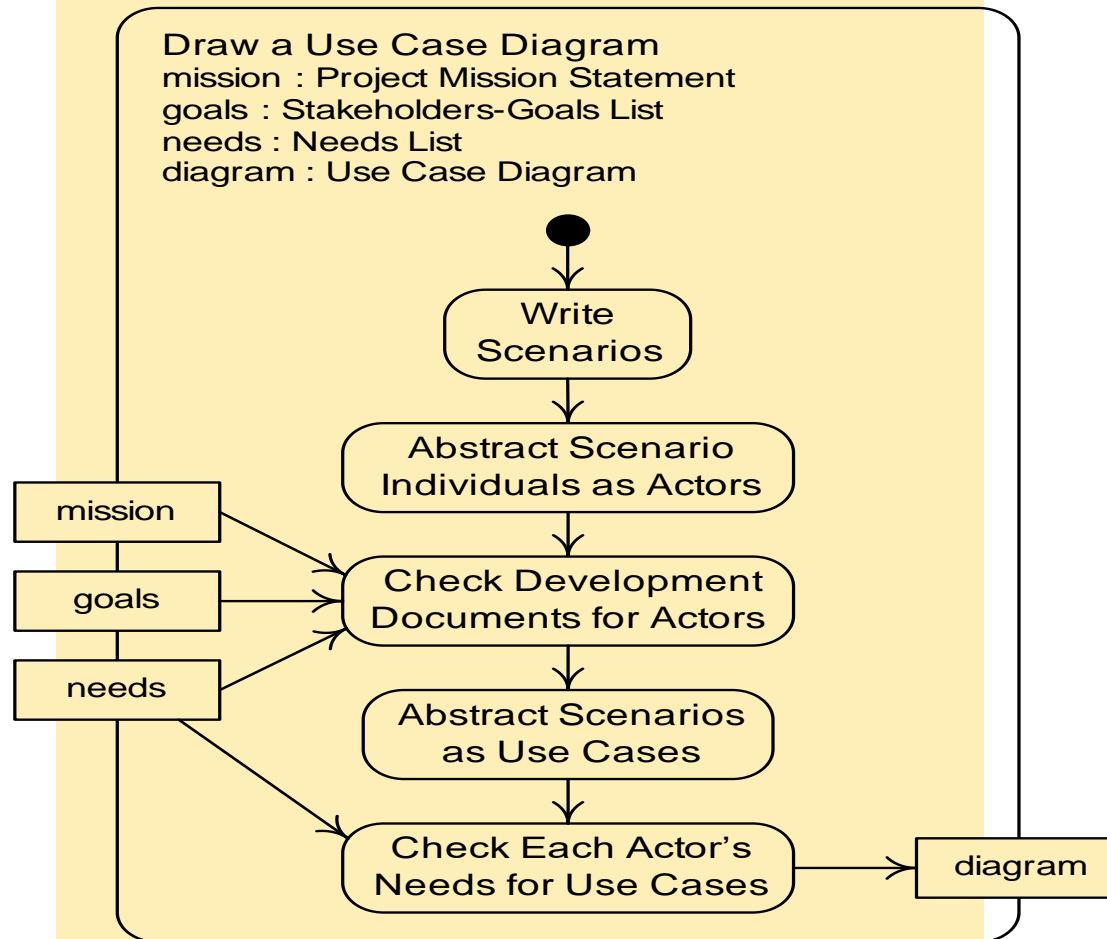
The kiosk refunds one dollar.

The carwash queries its stall sensors.

The stall sensors report that no car is detected, so the kiosk displays the message “Please drive forward into the wash stall.”



Making Use Case Diagrams 1





Making Use Case Diagrams 2

- An event list a list of all internal and external events to which the product must respond.
- Invent use cases to handle each event in the list; add it to the diagram
- Consider each use case and determine its actors; add them to the diagram



Use Case Briefs

A **use case** or **actor brief** is a short description of a use case or actor.

- Supplements the use case diagram
- Usually only one or two sentences or phrases



Formation Rules

- Every use case diagram must have
 - At least one use case
 - At least one actor
 - At least one actor associated with each use case
 - At least one use case associated with each actor
 - No association line between actors
 - No association line between use cases
 - Name every actor and use case
 - Not label any association line



Use Case Diagram Heuristics

- Never make the product an actor.
- Name actors with noun phrases.
- Name use cases with verb phrases.
- Achieve a stakeholder's goal in a use case.
- Make use cases that can be finished in a single session.
- Make use cases of uniform size and complexity.
- Draw use case diagrams on one page.
- Organize use cases by actor, problem domain categories, or solution categories.



Checking Use Case Diagrams

- Review the stakeholders-goals list to make sure no actors are missing.
- Review the needs list to make sure no use cases are missing.
- Review constraints and limitations to make sure they are not violated.
- Generate an event list and check that all events are handled.
- Check that the collection of use cases covers all externally visible behavior.
- Check the diagram against the use case heuristics above.

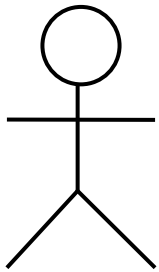


When to Use the Use Case Diagrams

- User-level product design alternatives (more on this later)
- Summarizing design alternatives
- Catalog use cases elaborated in use case descriptions (more on this later)
- A component of a use case model (more on this later)



Actors



Passenger

- An actor models an external entity which communicates with the system:
 - User
 - External system
 - Physical environment
- An actor has a unique name and an optional description.
- Examples:
 - Passenger: A person in the train
 - GPS satellite: Provides the system with GPS coordinates



Finding Actors

- Who or what will use the main functionality of the system?
- Who or what will provide input to this system?
- Who or what will use output from this system?
- Who will need support from the system to do their work?
- Are there any other software systems with which this one needs to interact
- Are there any hardware devices used or controlled by this system?



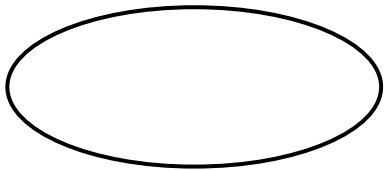
Basics Steps to write use cases

- Who is going to use the product?
- What will it be used for?
- How are they going to use it?



Use Case

A use case represents a class of functionality provided by the system as an event flow.



PurchaseTicket

A use case consists of:

- Unique name
- Participating actors
- Entry conditions
- Flow of events
- Exit conditions
- Special requirements



Use Case Diagram: Example

Name: Purchase ticket

Participating actor: Passenger

Entry condition:

- Passenger standing in front of ticket distributor.
- Passenger has sufficient money to purchase ticket.

Exit condition:

- Passenger has ticket.

Event flow:

1. Passenger selects the number of zones to be traveled.
2. Distributor displays the amount due.
3. Passenger inserts money, of at least the amount due.
4. Distributor returns change.
5. Distributor issues ticket.



Describing Use-cases

- Use-case Name:
- Use-case Number: system#.diagram#.Use-case#
- Authors:
- Event(Stimulus):
- Actors:
- Overview: brief statement
- Related Use-cases:
- Typical Process description: Algorithm
- Exceptions and how to handle exceptions:

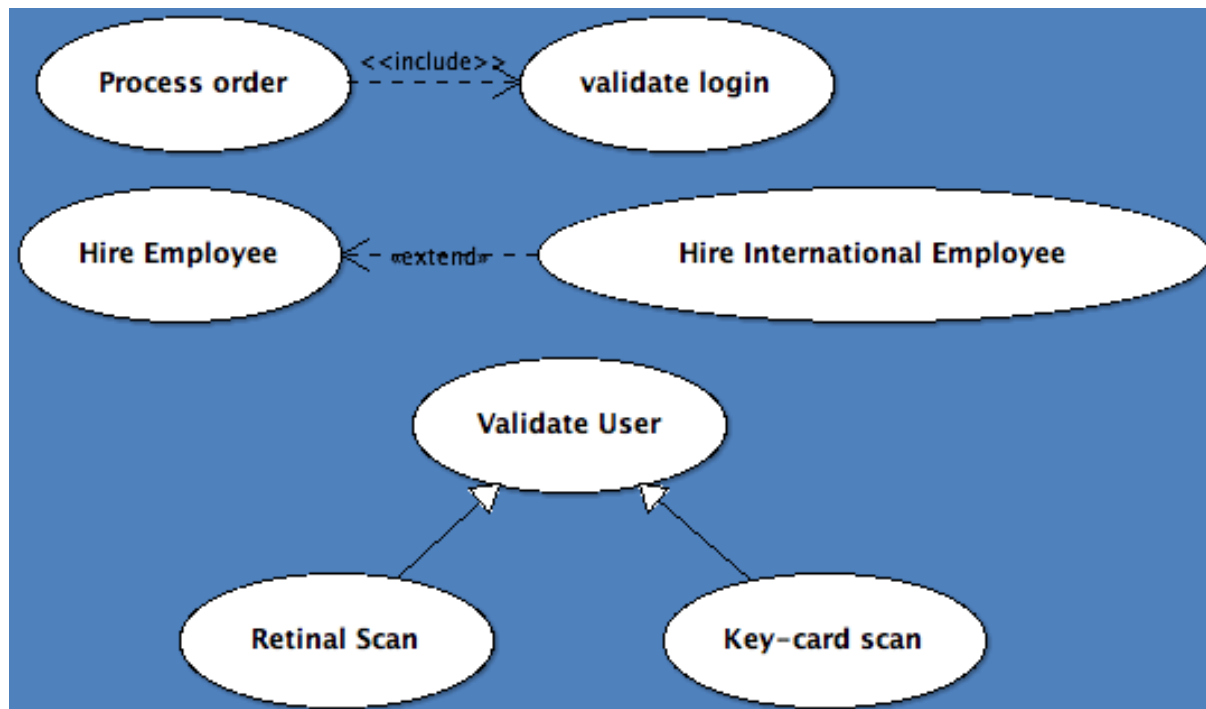


Use Case Relationships

Includes

Extends

Generalization



After a while you realize extends and generalization are not too different. Just know generalization and includes... forget about extends (the difference is only in intent)



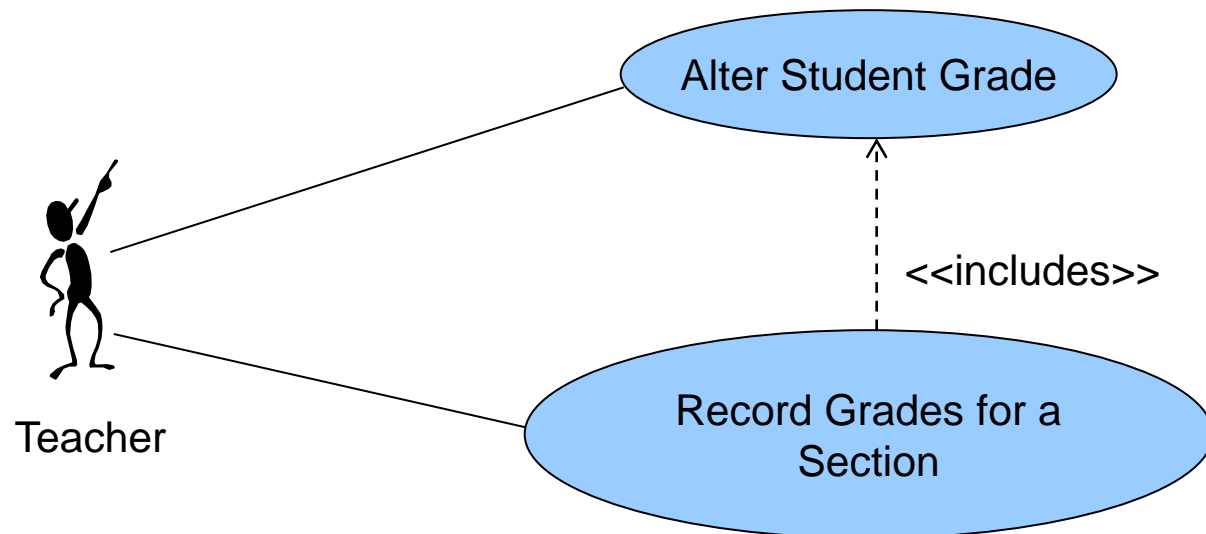
Bank

- actor Customer
- actor Bank Teller
- usecase Withdraw Money
- usecase Deposit Money
- usecase Transfer Funds
- Customer -> Withdraw Money
- Customer -> Deposit Money
- Customer -> Transfer Funds
- Bank Teller -> Withdraw Money
- Bank Teller -> Deposit Money



Use-Case Relationships

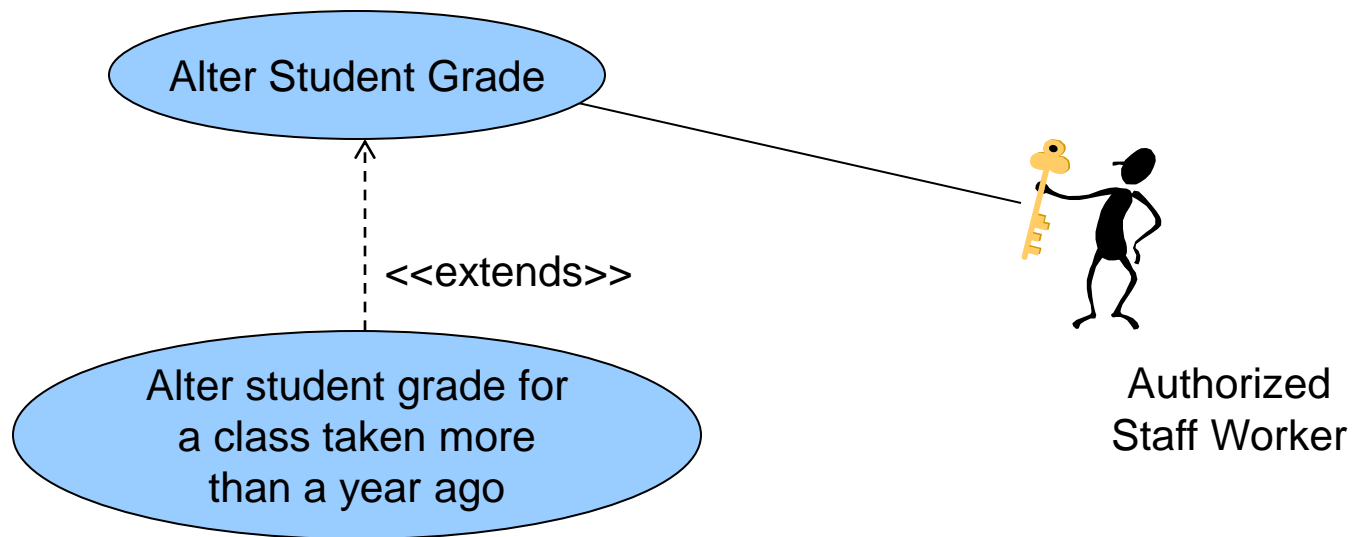
- *Includes Dependency*: Defines how one use case can invoke behavior defined by another use case





Use-Case Relationships

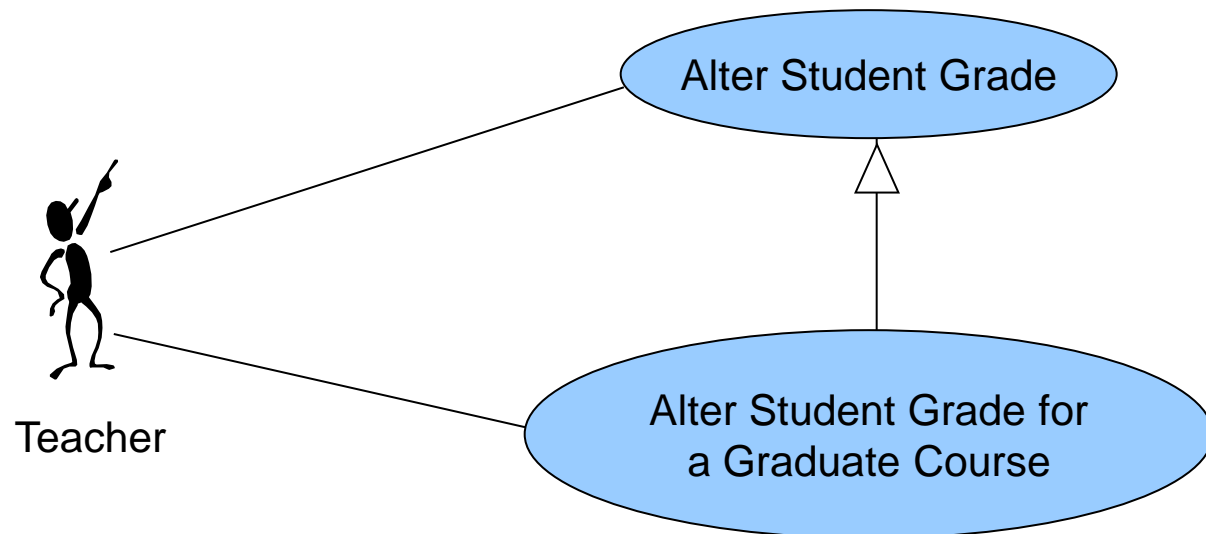
- *Extends dependency*: defines a use-case that is a variation of another, usually for handling an abnormal situation





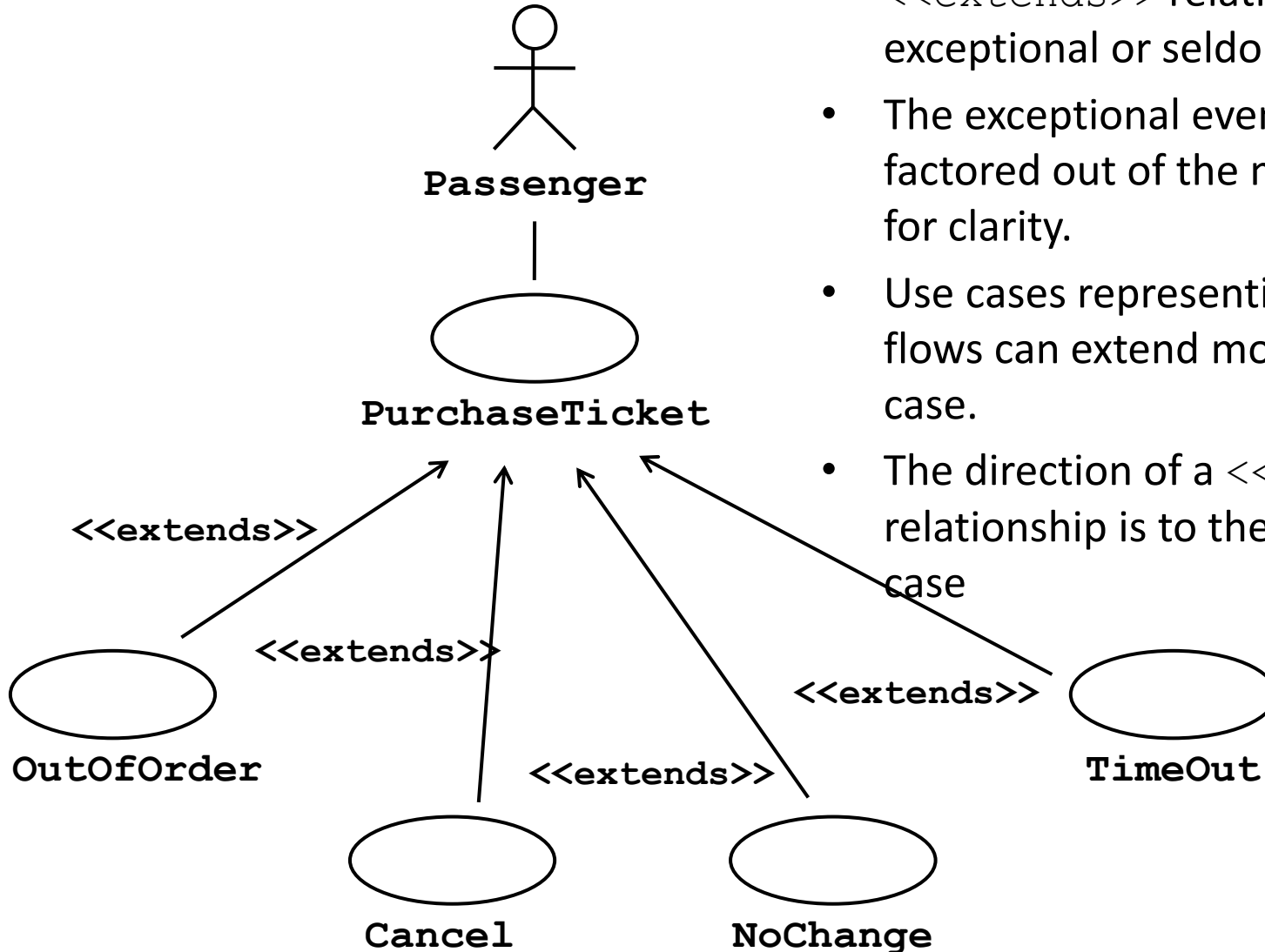
Use-Case Relations

- *Generalization*: Defines one use case as a **generalization of another**. Replaces generic functionality with alternate implementation





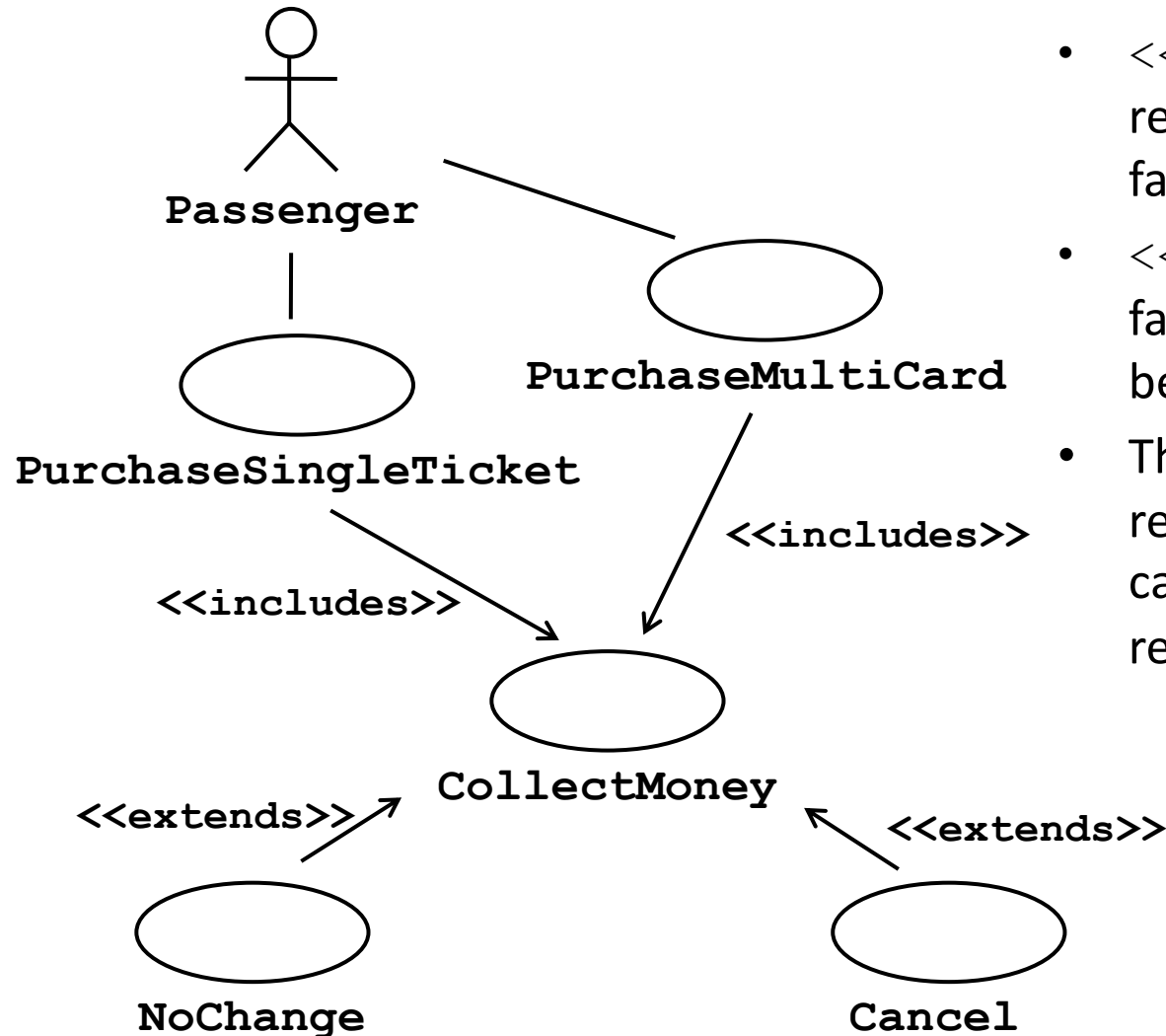
The <<extends>> Relationship



- <<extends>> relationships represent exceptional or seldom invoked cases.
- The exceptional event flows are factored out of the main event flow for clarity.
- Use cases representing exceptional flows can extend more than one use case.
- The direction of a <<extends>> relationship is to the extended use case



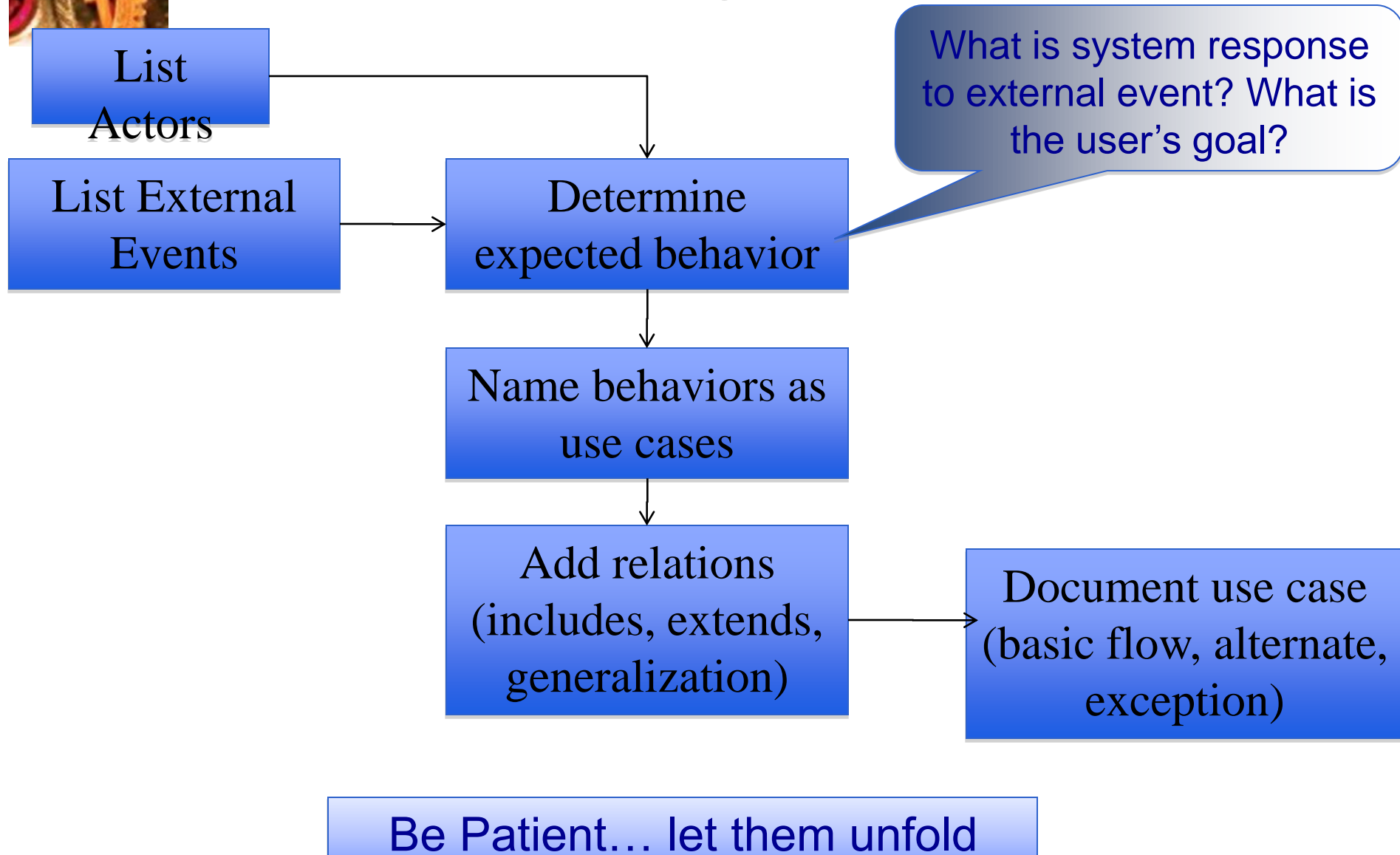
The <<includes>> Relationship



- <<includes>> relationship represents behavior that is factored out of the use case.
- <<includes>> behavior is factored out for reuse, not because it is an exception.
- The direction of a <<includes>> relationship is to the using use case (unlike <<extends>> relationships).



Documenting Use Cases





Use Cases

□ When to Use Use Cases

□ Fowler's View: do use cases first before object modeling

- Capture the **simple, normal use-case** first
- For every step ask “*What could go wrong?*” and how it might work out differently
- Plot all variations as *extensions* of the given use case

What did we do?

□ Another view: do object modeling first, then use cases

□ Another: iterate model - use case - model - use case ...

□ Scenarios describe a single path, or a particular sequence

- E.g., Use Case: Order Goods
 - Scenario 1: all goes well
 - Scenario 2: insufficient funds
 - Scenario 3: out of stock

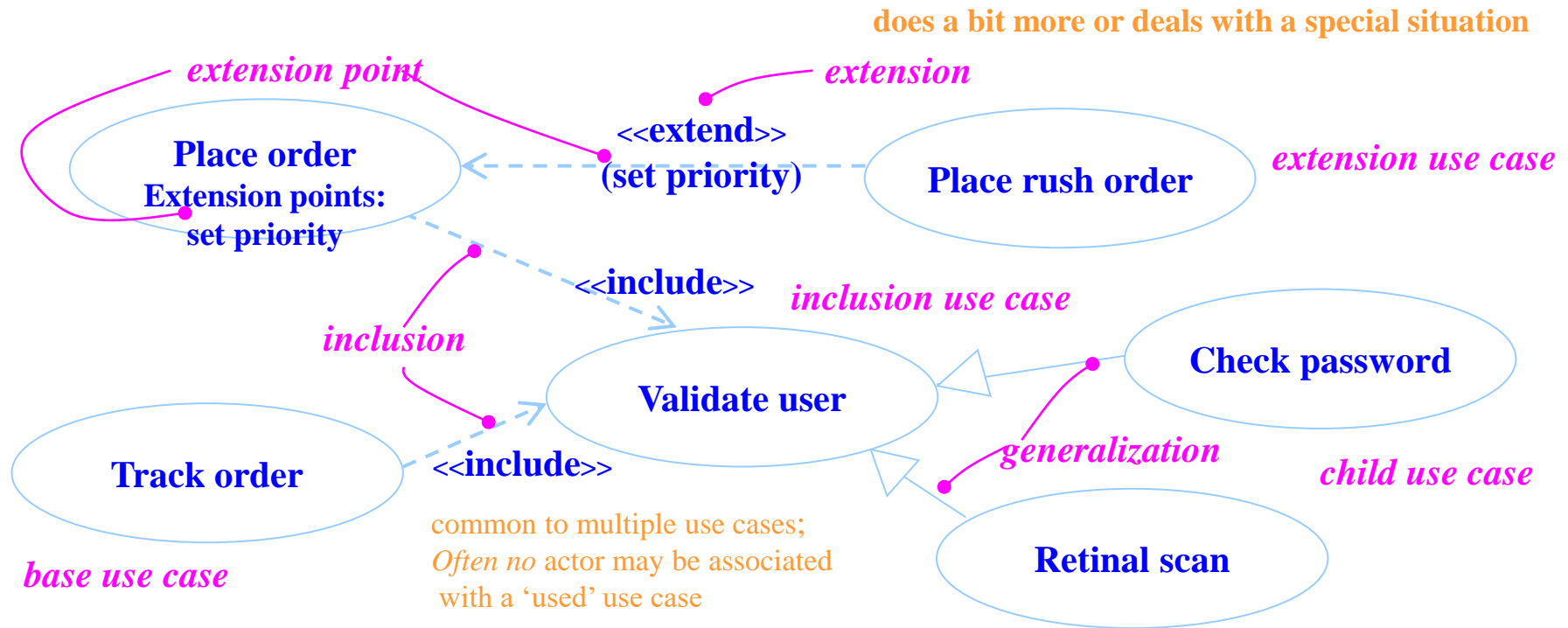
□ System test cases: Generate a test script for each scenario (flow of events).

- Obtain initial state from preconditions.
- Test success against post conditions.



Organizing Use Cases

- Generalization, Extend, Include/Use, packages



- Track Order** - Obtain and verify the order number; For each part in the order, query its status, then report back to the user.
- Place Order** - Collect the user's order items. (set priority). Submit the order for processing.



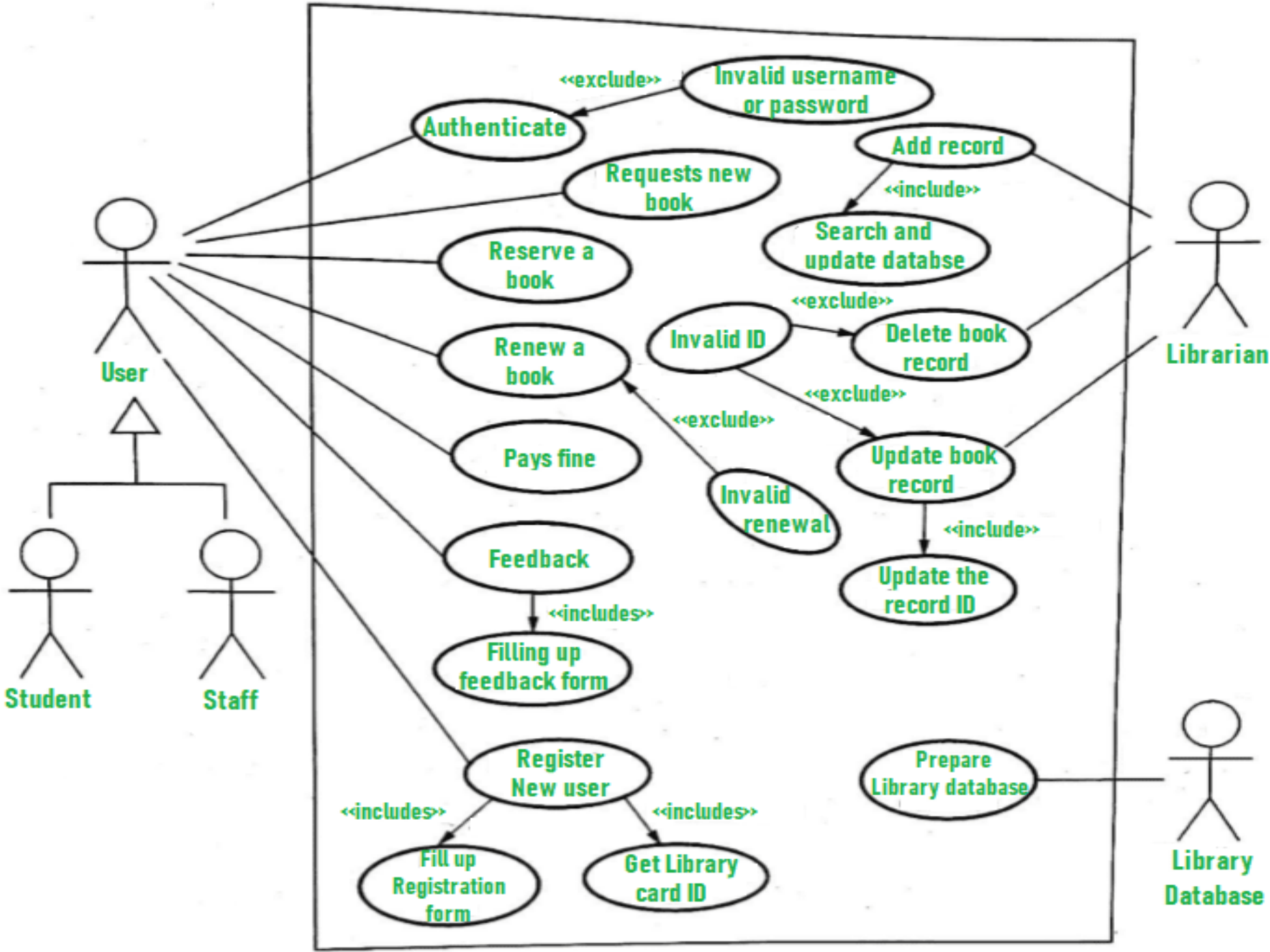
Benefits of Use Cases

- Use cases diagrams capture user-visible functions
- Identifying actors help capture who needs the system functionality
- Relationships between use cases document opportunities for reuse
- Use cases provide a basis planning and scheduling incremental development
- Use cases can provide a basis for system testing



LMS – Use case Representation


- **Actors:**
 - User (Staff or Student)
 - Librarian
- **Use Cases:**
 - Register New User
 - Issue Library Card
 - Request New Book
 - Reserve Book
 - Renew Book
 - Pay Fine
 - Fill Feedback Form
 - Manage Records
 - Delete Records
 - Update Database
- **System Boundary:** The system boundary will encompass all the use cases





Use Case Analysis

- Name Login
 - Actors Student/Faculty/Admin
 - Preconditions Existence of database and correctly installed web-based server.
 - Description When program is opened using browser the user sees logon page.
- Name Manage Users
 - Actors Admin
 - Preconditions Existence of database and correctly installed web-based server.
 - Description Manages the users and their actions.

- 
- Name Add/remove Books to the library Actors Admin
 - Preconditions Existence of database and correctly installed web-based server.
 - Description Manages the books in the library.
-
- Name Processing issues Actors Admin
 - Preconditions Existence of database and correctly installed web-based server.
 - Description Manages the request of issues and requests by users/faculty.
-
- Name Request issue/return
 - Actors Student/Faculty
 - Preconditions Existence of database and correctly installed web-based server.
 - Description Student/Faculty can request for books issuing or can return the issued books.



- Name Search books
- Actors Student/Faculty
- Preconditions Existence of database and correctly installed web-based server.
- Description Student or Faculty can search for the required books.
- Name Recommend books
- Actors Faculty
- Description Faculty can recommend the new required books



Example 1: Quiz Instant Feedback

- **Description:** An [educational technology](#) company wants to develop a feature that allows students to take quizzes and receive instant feedback
- **Primary actor:** Students
- **Goals:** Take a quiz, view quiz results
- **Stakeholders:** Educational technology company, students, educators, school administration, investors
- **Pre-conditions:** Student must be logged in, student must have access to the quiz
- **Post-conditions:** Student can take the quiz and receive instant feedback on their performance
- **Basic flow:**
 - Student logs into the educational technology platform and selects the option to take a quiz
 - System presents the student with the quiz questions
 - Student answers the questions and submits the quiz
 - System evaluates the student's answers and provides instant feedback on their performance
 - System records the quiz results and makes them available for the student, educator, and administration to view
 - Student views the feedback and can review their answers if they desire
- **Alternate path:**
 - Student logs into the educational technology platform and selects the option to view previous quiz results
 - System presents the student with a list of previous quizzes they have taken, along with their results
 - Student selects a previous quiz to view
 - System displays the student's answers and the correct answers for each question
 - Student reviews their answers and receives feedback on their performance



Social Media Live Streaming Feature

Description: A social media company wants to develop a feature that allows users to live stream to their followers

- **Primary Actor:** Social media users

- **Goals:** Start a live stream, end a live stream, view live stream

- **Stakeholders:** Social media company, social media users, followers, advertisers, investors

- **Pre-conditions:** The user must have a [stable internet connection](#)

- **Post-conditions:** The user's followers can view the live stream, user can end the live stream at any time

- **Basic flow:**

- User logs into the social media platform and selects the option to start a live stream

- System accesses the user's camera and microphone and displays a preview of the live stream to the user

- User starts the live stream, which is broadcast to their followers in real time

- Followers can view the live stream, engage with the user through comments or other interactions, and can share the live stream to their own followers

- User ends the livestream

- **Alternate path:**

- User logs into the social media platform and selects the option to schedule a live stream

- System prompts the user to enter the date, time, and title of the scheduled live stream.

- System sends a notification to the user's followers, letting them know when the scheduled live stream is starting

- At the scheduled time, the user starts the live stream, which is broadcast to their followers in real-time

- Followers can view the live stream, engage with the user through comments or other interactions, and can share the live stream with their own followers

- User ends the livestream



Use Cases are useful to...

- Determining requirements
 - New use cases often generate new requirements as the system is analyzed and the design takes shape.
- Communicating with clients
 - Their notational simplicity makes use case diagrams a good way for developers to communicate with clients.
- Generating test cases
 - The collection of scenarios for a use case may suggest a suite of test cases for those scenarios.



Summary

- Use cases are types of episodes of interaction between a product and its environment.
- Working with use cases help organize, analyze, generate, evaluate, and select functional requirements.
- UML use case diagrams are static models of all the use cases in a product.
- There are several processes for creating use case diagrams.
- Rules govern use case diagram formation, and heuristics and checks help achieve and ensure their quality.



1. Use case description consists of the following...

- (A) Actors
- (B) Number and Use case name
- (C) Need and stakeholder
- (D) Both a and b
- (E) All of the above

2. Select the methods used by the use case to write a description.

- (A) Actors in a use case are mostly stakeholders
- (B) Preconditions always be true in advance statement start
- (C) Requires a list should be checked when writing each use case
- (D) Both A and C
- (E) All of the mentioned

3. Select from the following which is included by the Use case Description Heuristics.

- (A) Fill up in the use case template from top to bottom
- (B) put down easy declarative sentences in the active voice
- (C) Keep away from the sequence of pace through the actors and product
- (D) Both A and C
- (E) All of the above