Structured Versus Object Oriented Methodologies

METHODOLOGY OVERVIEW

- Methodology defined: The way something gets done. The strategy, steps, directions, or actions.

- Classifications of Methodologies:

  - Structured Systems Analysis and Design (SSAD)

  - Object-Oriented Analysis and Design (OOAD)

  - Others

Structured Systems Analysis and Design (SSAD)

- SSAD techniques
  - Data Modeling – The data requirements of the system being designed are identified, modeled and documented. This data is separated into entities and relationships between these entities identified.
  - Process Modeling – Concerned with how the data moves around the information system. Examines processes, data stores, external entities and data flows.
  - Entity Behavior Modeling – The identifying, modeling and document events with respect to the entities in the system and the order in which these events take place.

Structured Systems Analysis and Design (SSAD)

- Some of SSAD tools
  - DFD
  - ERD
  - Decision Table.

Object-Oriented Analysis and Design (OOAD)

- An investigation of the problem (rather than how a solution is defined)
- During OO analysis, there is an emphasis on finding and describing the objects (or concepts) in the problem domain.
  - For example, concepts in a Library Information System include Book, and Catalog.

Object-Oriented Analysis and Design (OOAD)

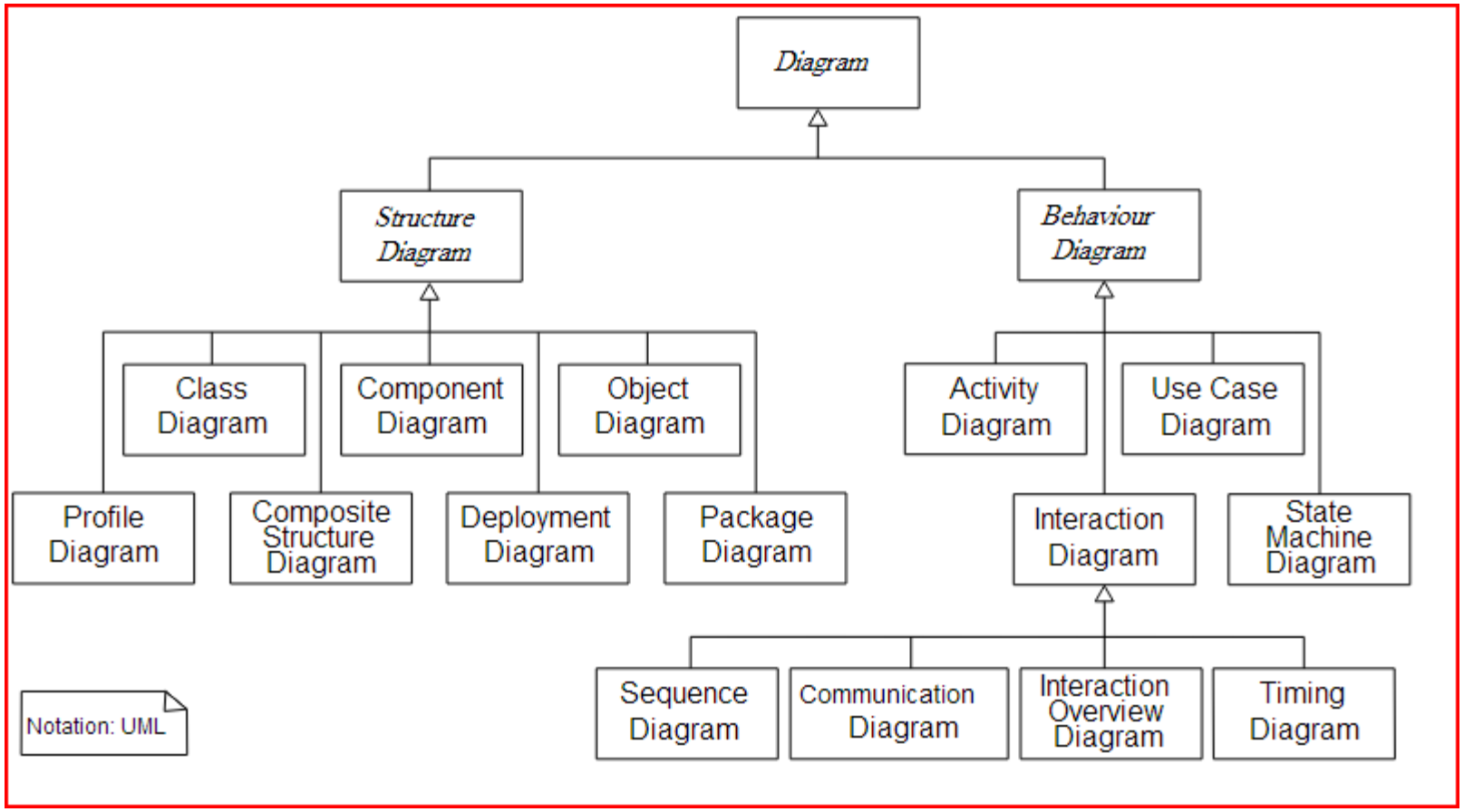
- Need to define software objects and how they collaborate to fulfill the requirements.
  - For example, in the Library Information System, a Book software object may have a title attribute and a display() method.
- Designs are implemented in a programming language.
  - For example, a Book class written in Java.

What is UML?

- Unified Modeling Language.

- A notational system aimed at modeling systems using object-oriented concepts.
  - Powerful, but complex language
  - Can be misused to generate unreadable models
  - Can be misunderstood when using too many exotic features

UML diagrams overview



USE CASE

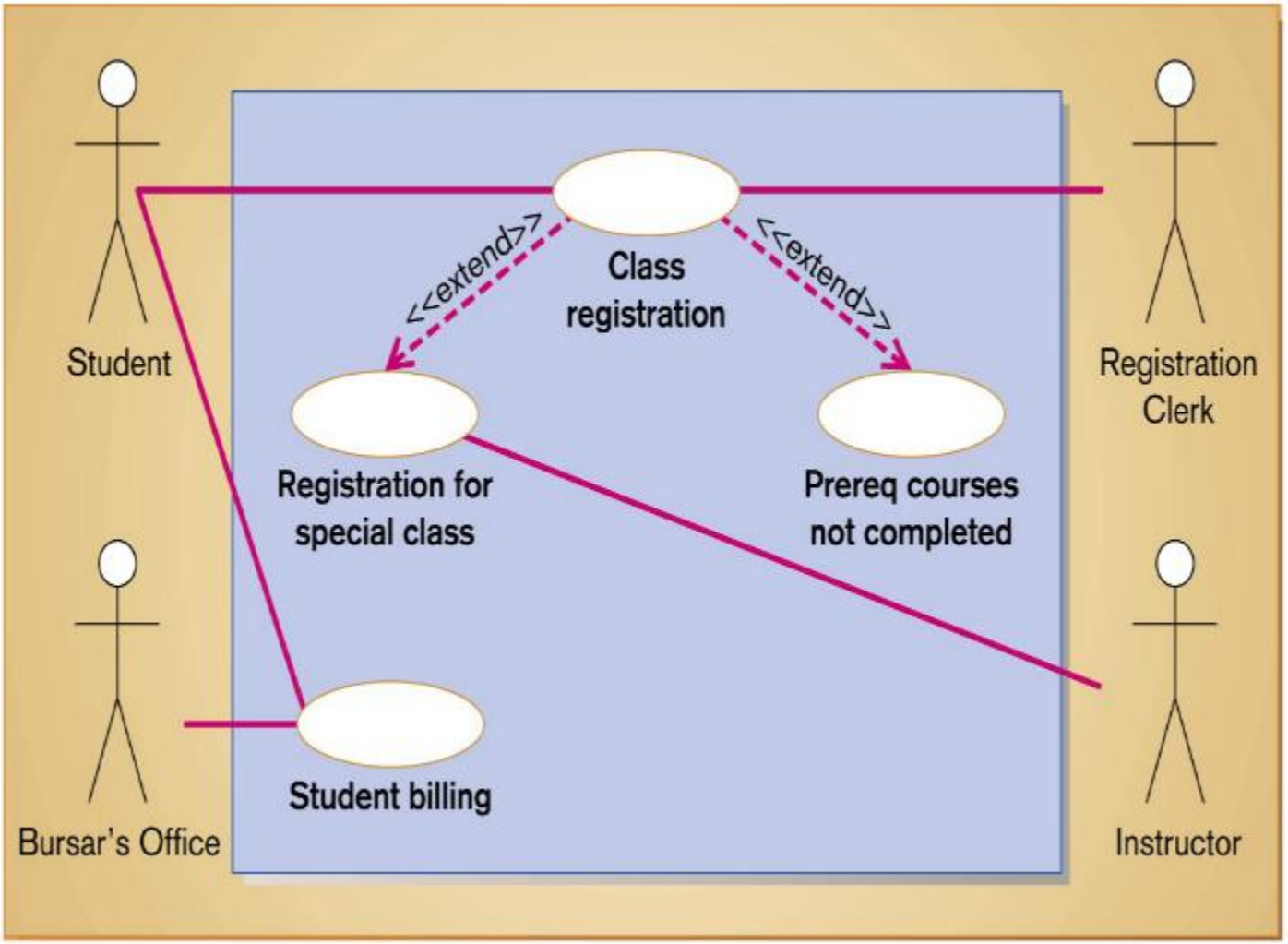
APPENDIX 7A

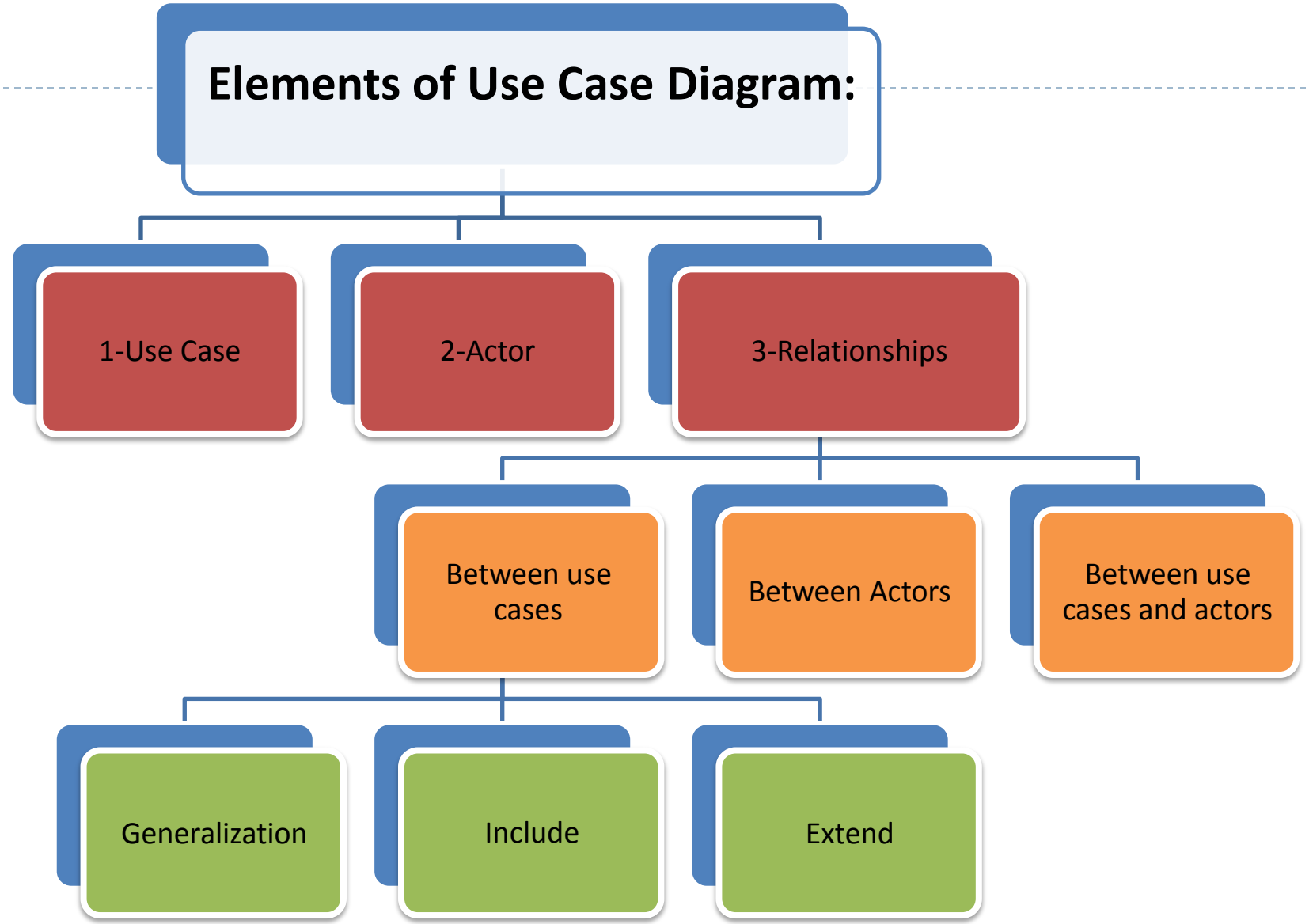
Using Use Case Diagrams

- Use case diagrams are used to visualize, specify, construct, and document the (intended) behavior of the system, during requirements capture and analysis.

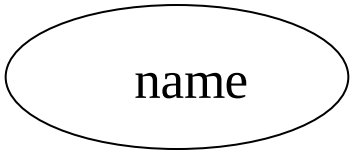
- Use case diagrams contain use cases, actors, and their relationships.

A use case diagram for a University registration system




Elements of Use Case Diagram:
1-Use Case
2-Actor
3-Relationships
Between use cases
Between Actors
Between use cases and actors
Generalization
Include
Extend

1- Use Case



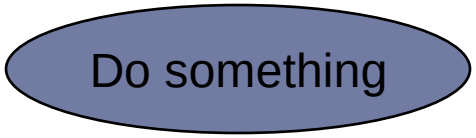
- Use cases specify desired behavior.
- A use case is a description of a set of sequences of actions, including variants, a system performs to yield an observable result of value to an actor.
- Each sequence represent an interaction of actors with the system.

Specifying the Behavior of a Use Case

- Describing the flow of events within the use case.
- Can be done in natural language, formal language or pseudo-code.
- Includes: how and when the use case starts and ends; when the use case interacts with actors and what objects are exchanged; the basic flow and alternative flows of the behavior.

(More details in slide 36)

Use Case

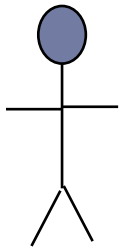


- System function (process – automated or manual). Named by verb.

- Each Actor must be linked to a use case, while some use cases may not be linked to actors.

USER/ACTOR	USER GOAL = Use Case
Order clerk	Look up item availability Create new order Update order
Shipping clerk	Record order fulfillment Record back order
Merchandising manager	Create special promotion Produce catalog activity report

2- Actor



Name

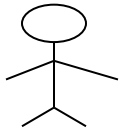
- *Actor* is someone interacting with use case (system function). Named by noun.

- Similar to the concept of user, but a user can play different *roles*; while Actor represent a role that the user can play (example: a professor can be an instructor and a researcher – plays 2 roles with two systems).

- Actor *triggers* use case.

- Actor has responsibility toward the system (inputs), and Actor have expectations from the system (outputs).

Actors



- An actor represents a set of roles that users of use case play when interacting with these use cases.
- Actors can be human or automated systems.
- Actors are entities which require help from the system to perform their task or are needed to execute the system's functions.
- Actors are not part of the system.

Use Cases and Actors

- From the perspective of a given actor, a use case does something that is of value to the actor, such as calculate a result or change the state of an object.

- The Actors define the environments in which the system lives

3-1 Relationships between Use Cases

1. Generalization - use cases that are specialized versions of other use cases.

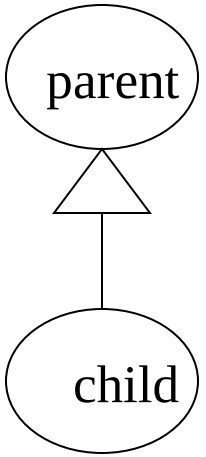
2. Include - use cases that are included as parts of other use cases. Enable to factor common behavior.

3. Extend - use cases that extend the behavior of other core use cases. Enable to factor variants.

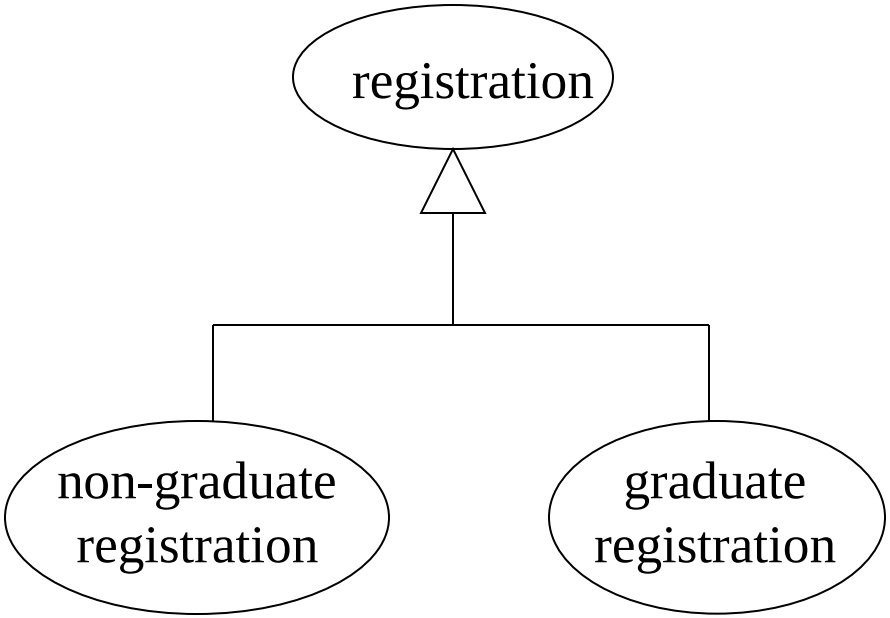
1.Generalization

- The child use case inherits the behavior and meaning of the parent use case.

- The child may add to or override the behavior of its parent.

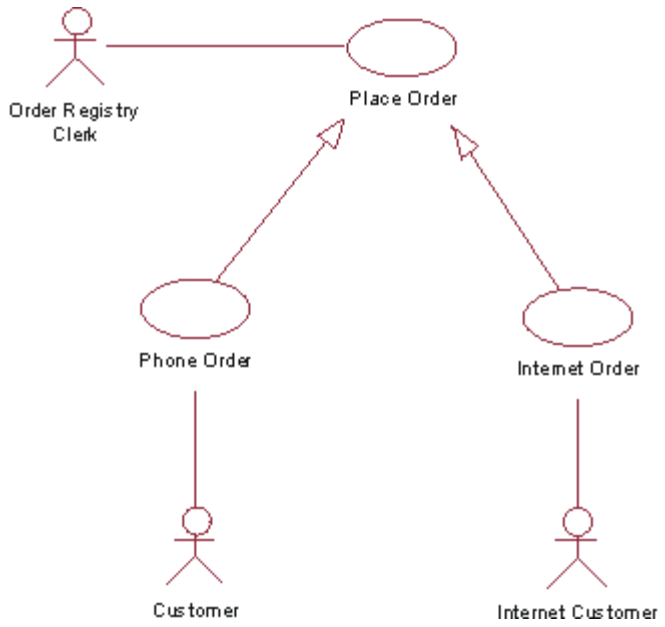


More about Generalization

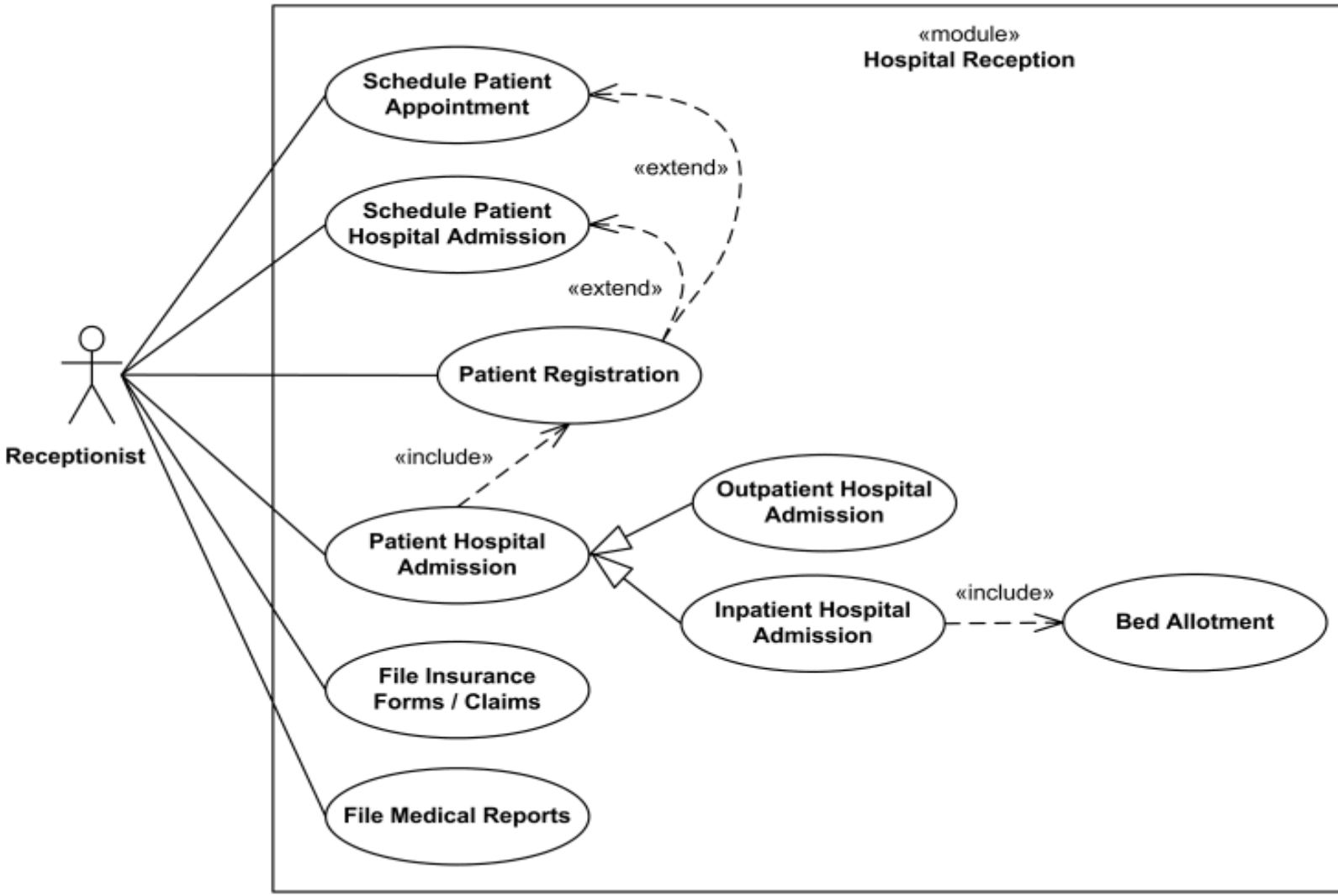


Generalization Example

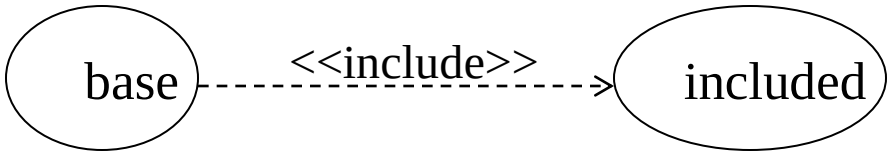
- The actor Order Registry Clerk can instantiate the general use case Place Order. Place Order can also be specialized by the use cases Phone Order or Internet Order.



Generalization Example



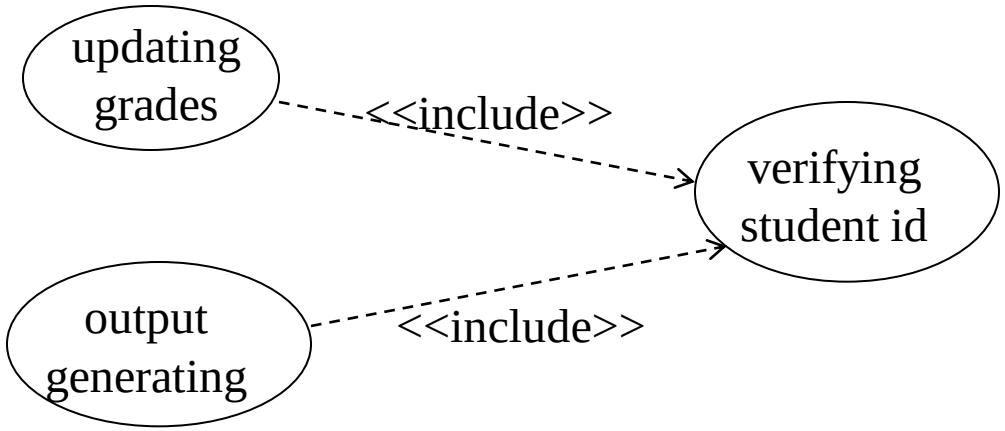
2. Include



- The base use case explicitly incorporates the behavior of another use case at a location specified in the base.
- The included use case never stands alone. It only occurs as a part of some larger base that includes it.

More about Include

- Enables to avoid describing the same flow of events several times by putting the common behavior in a use case of its own.



Include relationship

- Include relationship – a standard case linked to a ***mandatory*** use case.

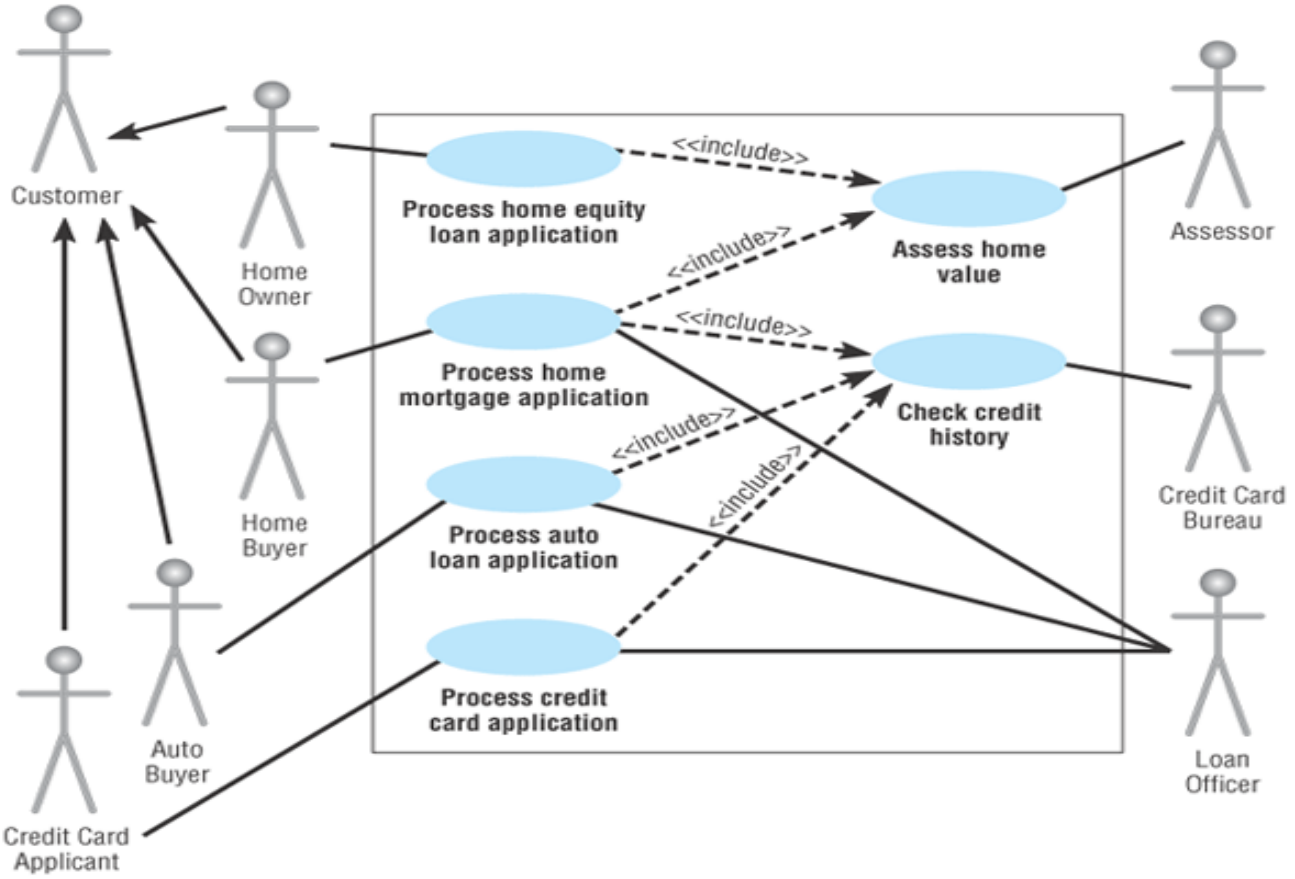
- Example: to *Authorize Car Loan* (standard use case), a clerk must run *Check Client's Credit History* (include use case).

- The standard UC include**s** the mandatory UC (use the verb to figure direction arrow).

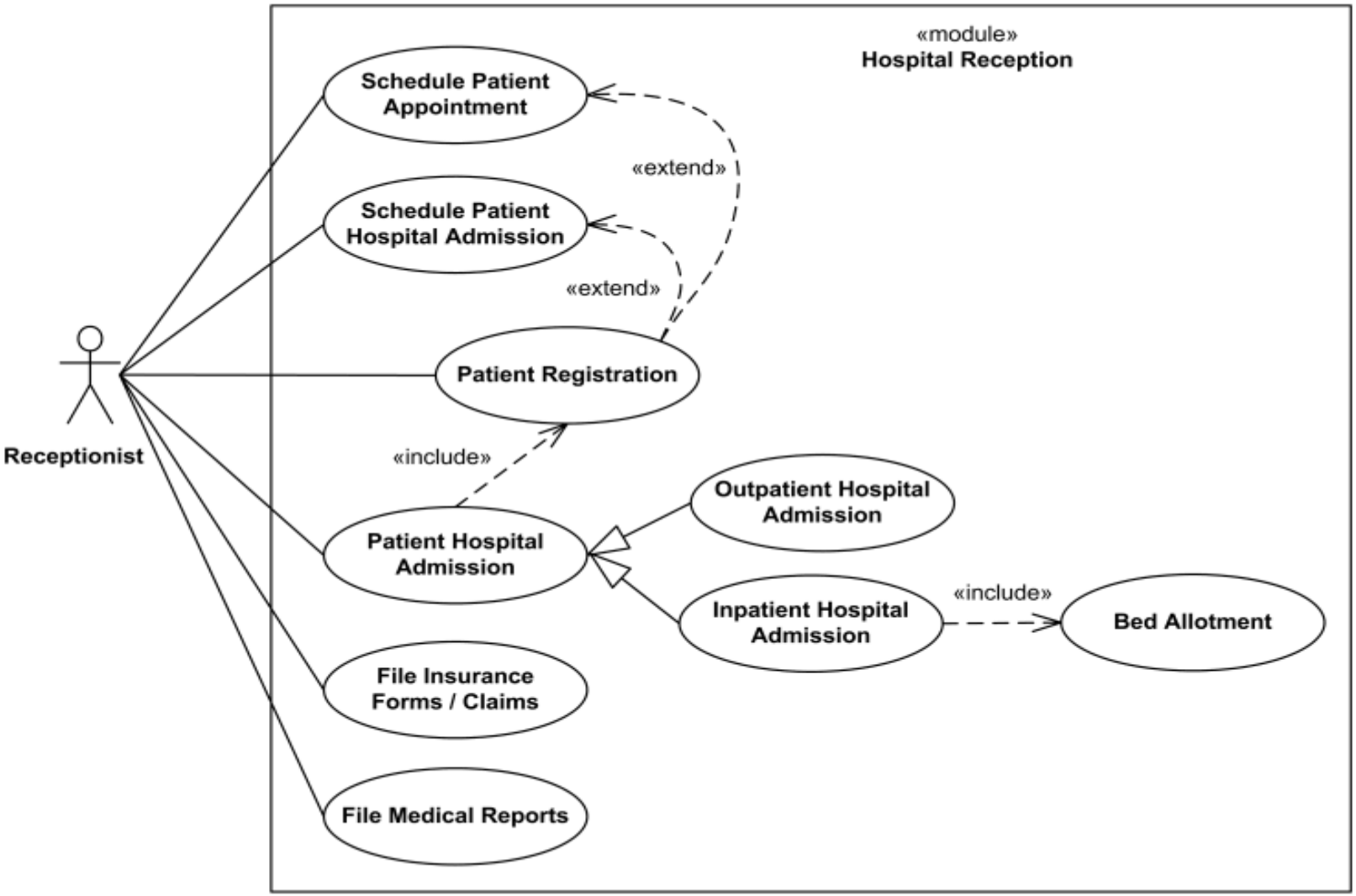
- Standard use case can NOT execute without the include case → tight coupling .

Reading use case diagram with Include relationship

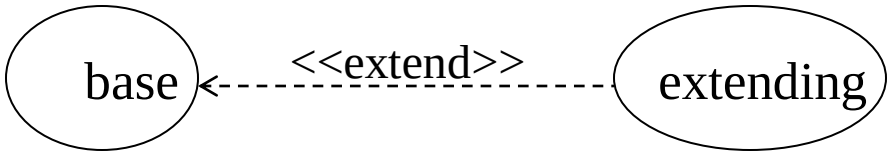
Figure 6.6
Examples of Generalized Use Cases and Include Relationships



Include Example



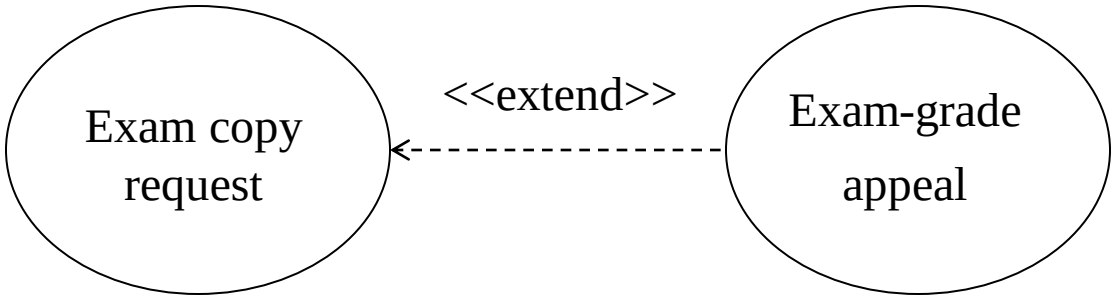
3. Extend



- The base use case implicitly incorporates the behavior of another use case at certain points called extension points.
- The base use case may stand alone, but under certain conditions its behavior may be extended by the behavior of another use case.

More about Extend

- Enables to model optional behavior or branching under conditions.



Extend relationship

- Extend relationship – linking an ***optional*** use case to a standard use case.

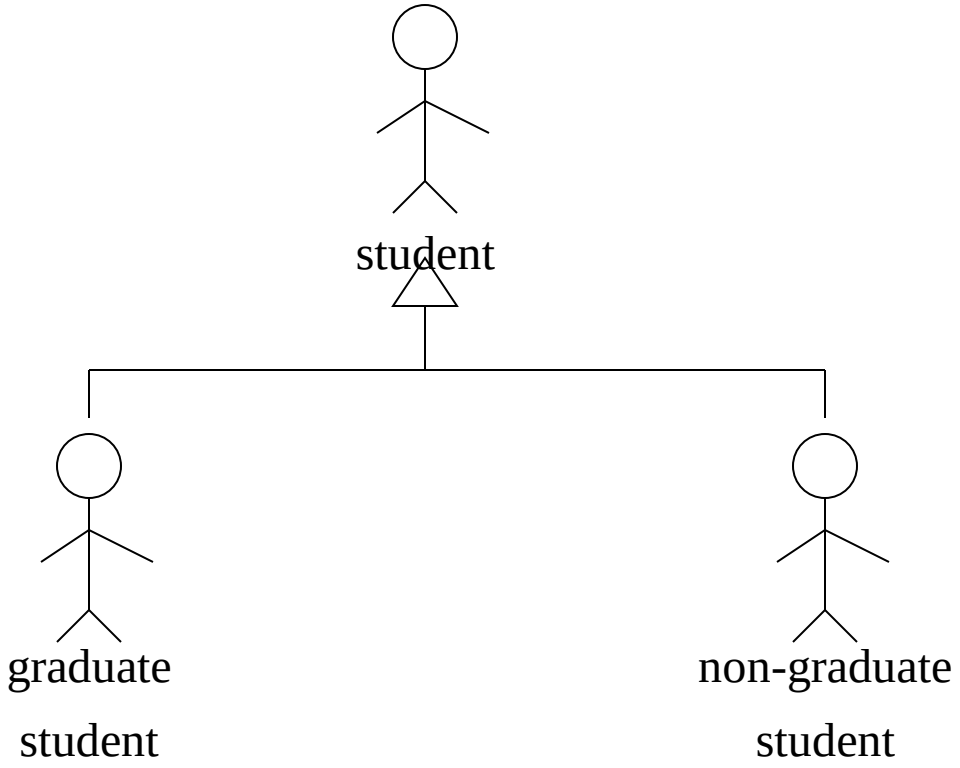
- Example: *Register Course* (standard use case) may have *Register for Special Class* (extend use case) – class for non-standard students, in unusual time, with special topics, requiring extra fees…).

- The optional UC extend**s** the standard UC

- Standard use case can execute without the extend case → loose coupling.

3.2 Relationships between Actors

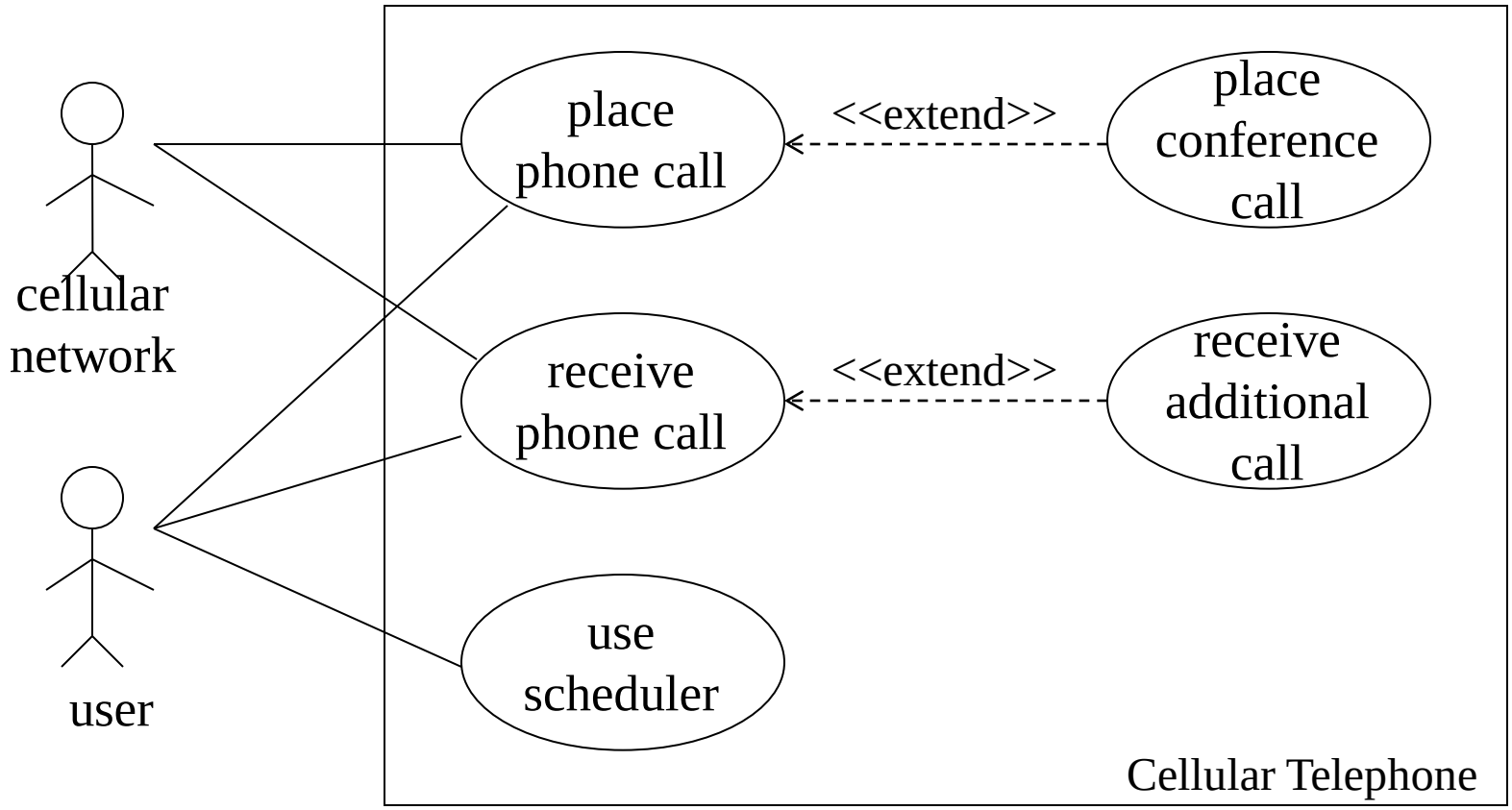
- Generalization.



3.3 Relationships between Use Cases and Actors

- Actors may be connected to use cases by associations, indicating that the actor and the use case communicate with one another using messages.

Example


cellular network
user
place phone call
receive phone call
use scheduler
<<extend>>
<<extend>>
place conference call
receive additional call
Cellular Telephone

Elements of use case diagram Summary

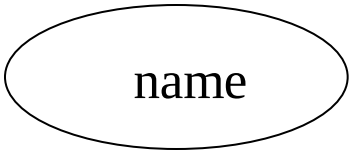
Connection between Actor and Use Case

Boundary of system

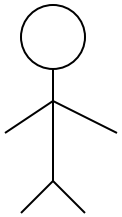
Include relationship between Use Cases (one UC must call another; e.g., Login UC includes User Authentication UC)

Extend relationship between Use Cases (one UC calls Another under certain condition; think of if-then decision points)

Elements of use case diagram Summary

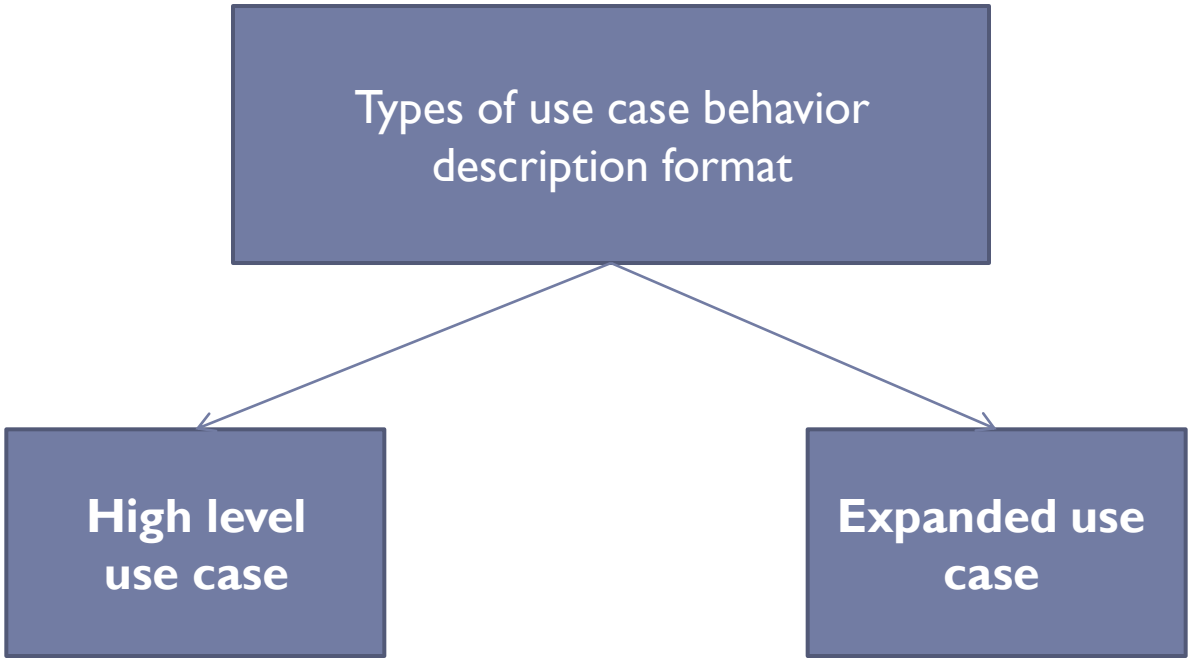


Use case , starts with a verb



Actor can be human or other system

Use Case Behavior Description



HL use case describes a process very briefly, usually in 2 or 3 sentences.

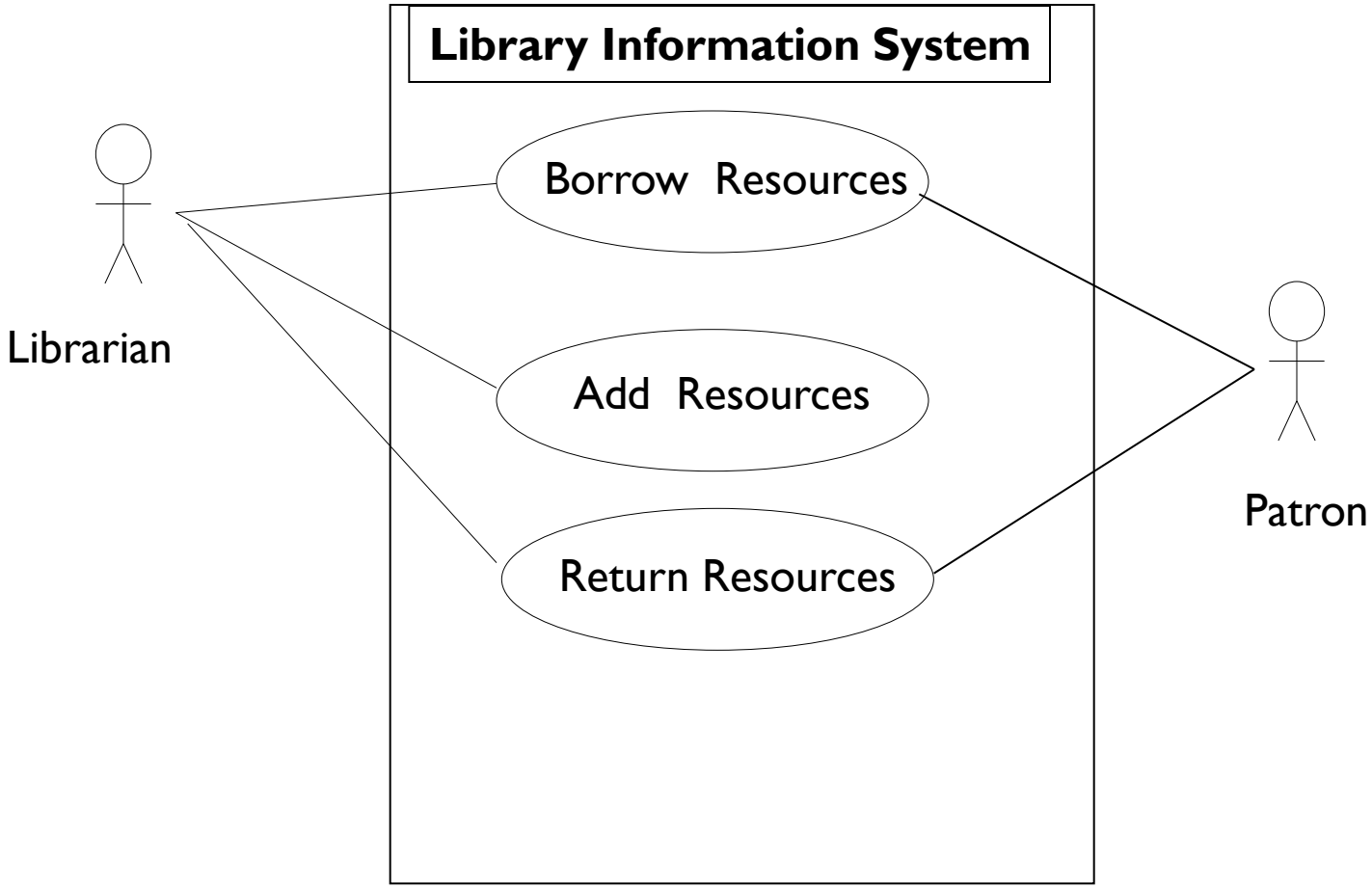
Expanded use case describes a process in details. It has an additional section not present in HL, Typical course of events

High level use case format

- It has the following format:

  - Use Case: Use case name
  - Actors: List of actors (external agents), indicating who initiates the use case
  - Description (Success scenario): Narrative description of the process

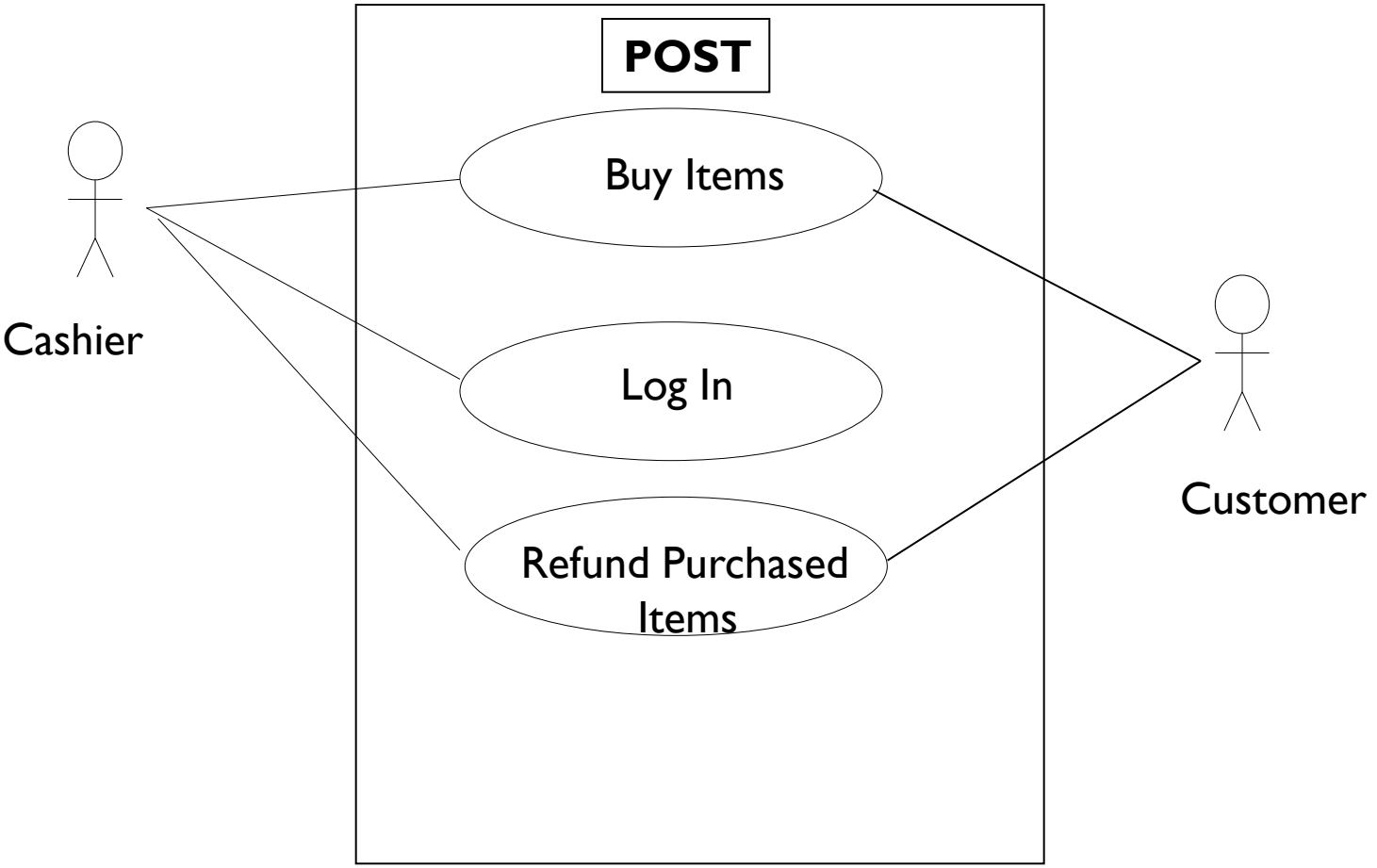
Use case diagram example



Example of High level use case

- Use Case: Add Resources.
- Actors: Librarian.
- Description: The use case begins when the Librarian receives new resources (books and 17 videos) to add to the catalog. The title, call number, and other information are recorded. Then the resources are placed on a shelf organized by resource type and call numbers.

Use case diagram example



Example of High Level Use case: Buy Items

- Use Case: Buy Items
- Actors: Customer(initiator) , Cashier
- Description : A customer arrives at a checkout with items to purchase. The Cashier records the purchase items and collects payment. On completion, the Customer leaves with the items.

Expanded use case format

- Use Case: Name of use case
- Actors: List of actors (external agents), indicating who initiates the use case
- Purpose: Intention of the use case
- Overview (Success scenario):
  - Repetition of HL use case, or some similar summary
- Type:
  - 1-primary, secondary or optional
  - 2- essential or real
- Cross References: Related use cases and system functions.
- Typical course of actions: describes in detail the conservation of interaction between the actors and the system.

Example: Expanded Use case Buy Items with Cash

- Use Case: Buy Items with Cash
- Actor: Customer (initiator), cashier
- Purpose: Capture a sale & its cash payment
- Overview (Success scenario):
  - A customer arrives at a checkout with items to purchase. The Cashier records the purchase items and collects a cash payment. On completion, the Customer leaves with the items
- Type: primary
- Cross References: Functions R1.2,…

Typical Course of Events

Actor Action	System Response
1. This use case begins when a Customer arrives at the POST checkout with items to purchase	
2. The Cashier records the identifier from each item If there is more than one of the same item, the Cashier can enter the quantity as well	3. Determines the item price and adds the item information to the running sales transaction
4. On completion of item entry, the Cashier indicates to the POST that item entry is complete	5. Calculate and presents the sale total
6. The Cashier tells the Customer the total	

Typical Course of Events

Actor Action	System Response
7. The Customer gives a cash payment possibly greater than the sale total	
8. The Cashier records the cash received amount	9. Shows the balance due back to the Customer & generate a receipt.
10. The Cashier deposits the cash received & extracts the balance owing The Cashier gives the balance owing, & the printed receipt to the Customer	11. Logs the completed sale
12. The Customer leaves with the items purchased	

Alternatives:
Line 2. Invalid identifier entered. Indicate errors
Line 7. Customer didn't have enough cash. Cancel sales transaction

Essential vs. Real Use Cases

Essential use cases are expanded use cases that are expressed in an ideal form free of technology and implementation details

Real use cases concretely describes the process in terms of its real current design, committed to specific input and output technology

Essential vs. Real Use Cases
Buy Items

Actor Action	System Response
1. This Cashier records the identifier for each item	2. Determines the item price & adds the item information to the running sales transaction
	The description & price of the item are presented

Essential Use Case

Actor Action	System Response
1. For each item, the Cashier types in the UPC in the UPC field of Window1. They then press "Enter Item" button with the mouse or Enter key	2. Display the item price & adds the item information to the running sales transaction
	The description & price of the current item are displayed in Textbox 2 of Window 1

Real Use Case

Plan & Elaborate Phase Steps

- Define system function
- Define system boundary, actors & use cases
- HL use cases
- Draw use case diagram
- Expand critical use cases (essential / Analysis)
- Real use case (Design)