Object Oriented Analysis and Design Fundamental Use Cases

COMP 3831

Larman: Chapter 6

Analysis and Design

Introduced in 1996
by Ivar Jacobson

Analysis – "what" *investigation of the problem and requirements*	**Design – "how"** *description of a software solution*
Requirements	Objects
Use cases ✓	Architecture
Constraints	Deployment
Vocabulary	UI

Use Cases

Use cases are requirements that define a promise or contract of how a system will behave.

Focus on the question:

How can using the system provide observable value to the user, or fulfill their goals?

Use cases are text documents, not diagrams. However, UML has notations to illustrate names of use cases and actors.

Use Case Definition

- A **use case** is a description of a system's behavior as it responds to a request that originates from outside of that system.
- A **use case** is the smallest unit of activity that is meaningful to the user. A use case must be self-contained and leave the system in a consistent state.

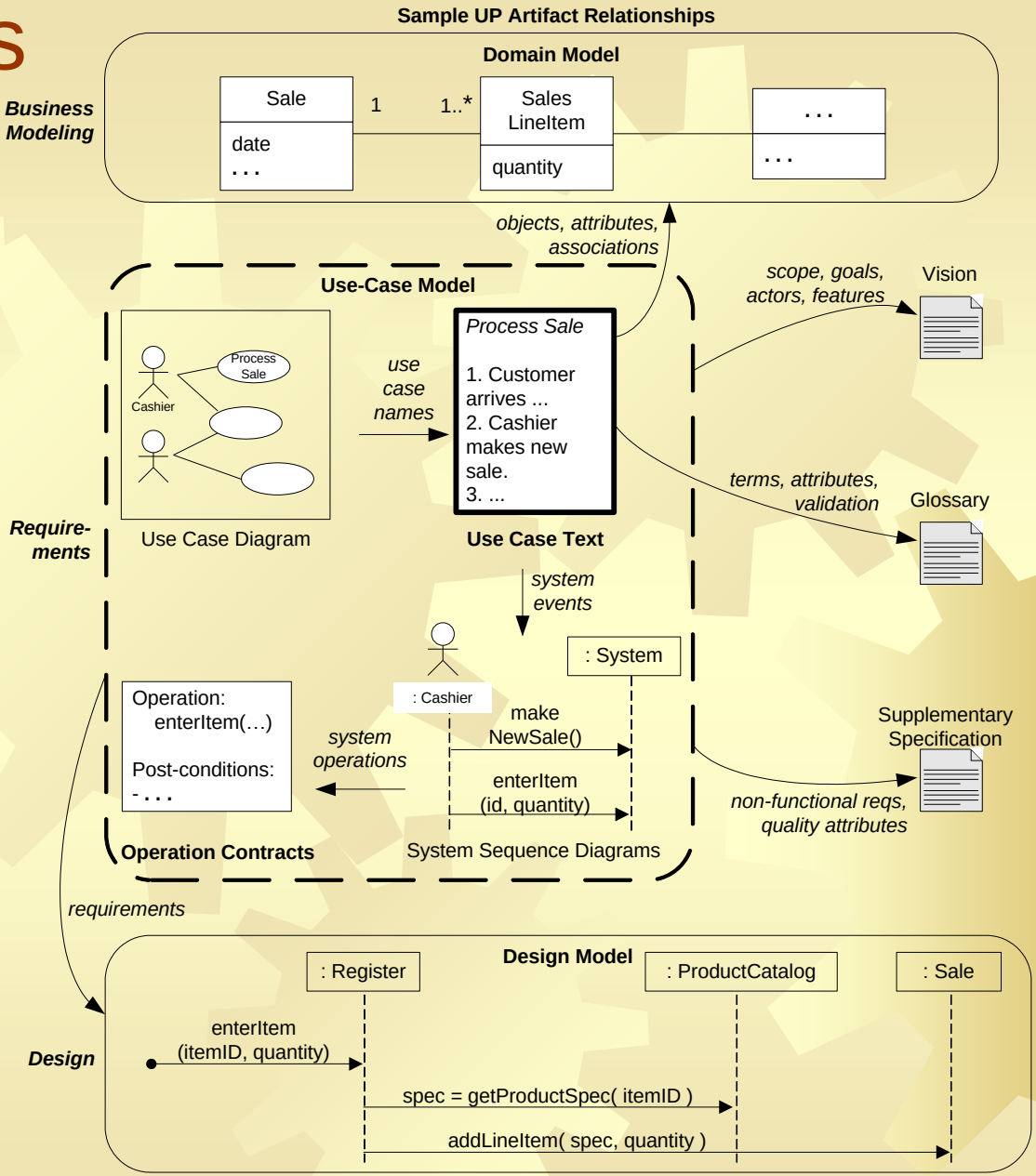
Use Case Definition

- Mechanism to discover and record requirements
- Discovering and recording functional requirements by writing *stories* of using a system to help fulfill various stakeholder goals
- A complete set of use cases for a system encompasses the complete system functionality from **users' point of view**
- Breaking the system down into use cases is an analysis process

Use Case Definition provided by RUP

- A use case defines a set of use-case *instances*, where each *instance* is a sequence of **actions** a system performs that yields an observable **result of value** to a particular *actor*.
- A use-case class contains all main, alternate flows of events related to producing the 'observable result of value'.
- Technically, a use-case is a class whose *instances* are *scenarios*.

UP Artifacts



Other Definitions

- Instance
  - Occurrence or example
- Scenario
  - A specific sequence of actions and interactions between actors and the system under discussion
  - Also described as a **use case instance**
  - One particular story of using a system or one path through the use case. Examples:
    - Successfully purchasing items with cash
    - Failing to purchase item after credit card denial

Actor

- Someone or something, **outside** the system that interacts with it.
  - Person (identified by role)
    - Cashier
    - Customer ….
  - Computer system
  - Organization

Types of Actors

- Primary actor – user goals fulfilled using the system services – cashier
- Supporting actor – provides service to the system – automated payment system
- Offstage actor – has an interest in the use case – government tax agency

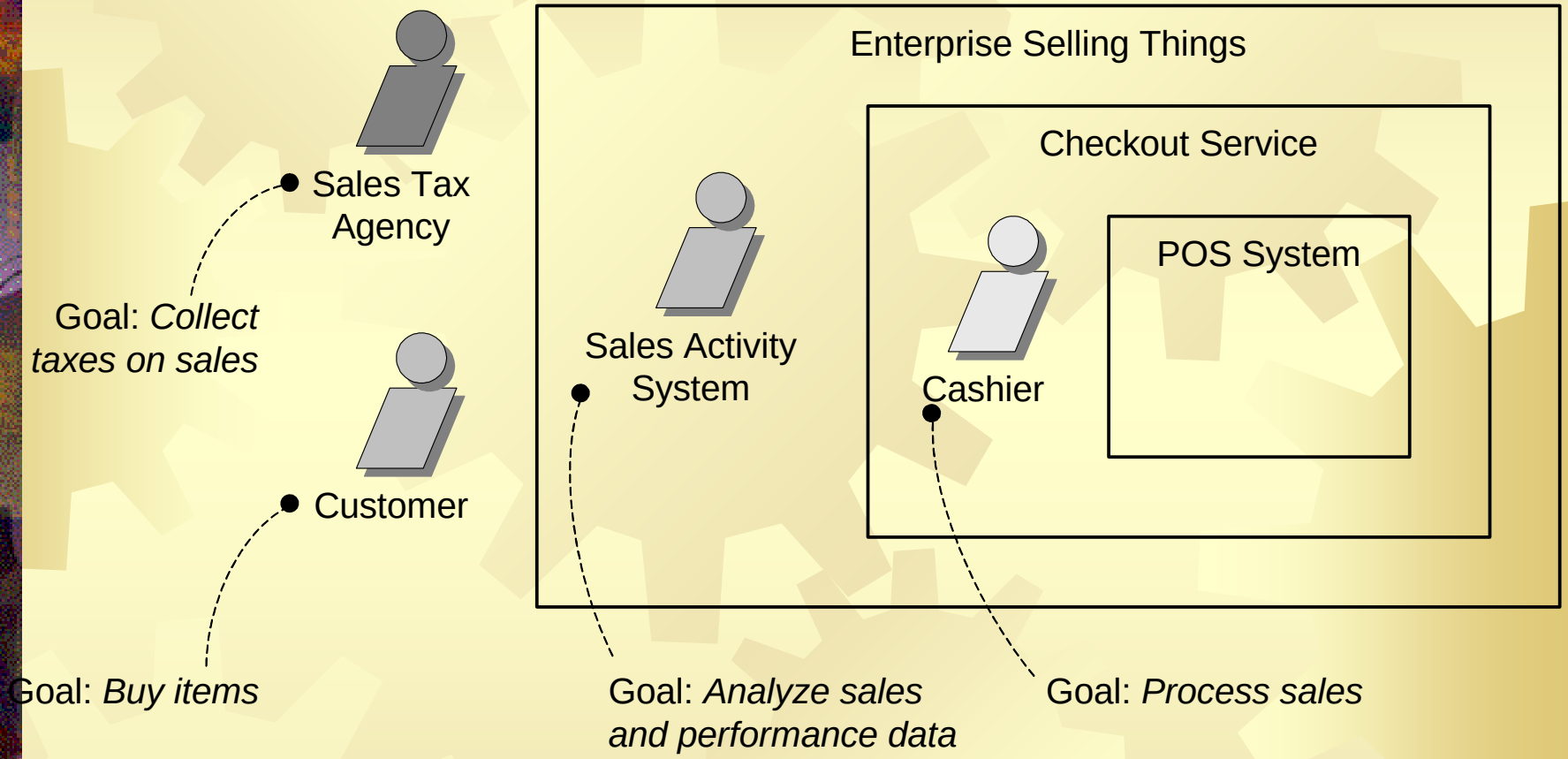
Identifying Actors

- Who uses the system?
- Who installs the system?
- Who starts and stops the system?
- Who does user and security management?
- Who maintains the system?
- Who does system administration?
- What other systems use the system?
- Who gets information from the system?
- Who provides information to the system?
- Does anything happen automatically at a preset time?
- Who evaluates system activity or performance?

Find System Boundary

- Choose system boundary
  - If not clear define: What is outside?
- Identify primary actors
  - Those that have user goals fulfilled through using services of the system
- For each primary actor, identify user goals
  - Highest user goal level that satisfies EBP guideline
- Define use cases that satisfy user goals; name them according to their goal.

POS System Boundary



Identifying Use Cases

- What functions will the actor want from the system?
- Does the system store information?
- What actors will create, read, update or delete information?
- Does the system need to notify any actors about its internal state?
- Are there any external events the system must know about?
- What actor informs the system about these events?

EBP Guideline

- Focus on use cases at the (E)lementary (B)usiness (P)rocess.

EBP is defines as:

- A task performed by one person in one place in one time, in response to a business event, which adds measurable business value and leaves the data in a consistent state. *Example: approve CC*.
- It is probably between a few minutes and an hour in length

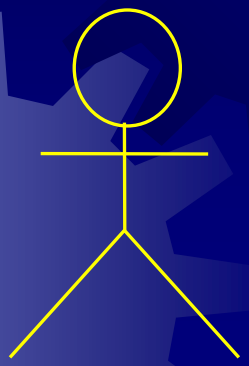
Describing Actors and Use Cases

- Each actor has descriptive name and a one or two sentence description
- Each use case also needs a descriptive name and a one or two sentence description
- Use cases are mostly about text
- Don't usually write it all at once
- Iterate through several times

Actor-Goal list

Actor	Goal	Actor	Goal
Cashier	Process sales, process rentals, handle returns, cash in, cash out …	System administra-tor	Add users, modify users, delete users, manage security, manage system tables, ….
Manager	Start up, shut down, ….	Sales activity system	Analyze sales and performance data ….

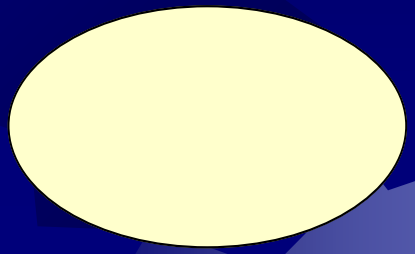
Diagramming Use Cases and Actors



Stick Man

This is an actor

This is an association between an actor and a use case



NewUseCase

This is a use case

Use case types and formats

- Brief: one paragraph summary, usually of the main success scenario
- Casual: informal paragraph format. Multiple paragraphs that cover various scenarios
- Fully dressed: most elaborate. All steps and variations are written in detail with supporting sections, such as:
  - Preconditions
  - Success guarantees

Brief Format example:

Process Sale: A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

Casual Format example:

Handle Returns:

Main Success Scenario: A customer arrives at a checkout with items to return. The cashier uses the POS system to record each returned item. …

Alternate Scenarios:

- If they paid by CC, and the reimbursement transaction to their CC account was rejected, inform the customer and pay them with cash.
- If the item identifier is not found in the system, notify the cashier and suggest manual entry of the identifier code (perhaps it is corrupted).
- If the system detects failure to communicate with the external accounting system, ….

Fully dressed example

- Elements that make up the fully dressed example are:
  - Primary Actor: The principal actor that calls upon system services to fulfill a goal
  - Stakeholders and interests: The parties using or affected by the uc
  - Preconditions: What must always be true before beginning a scenario in a use case?
  - Success Guarantee (post conditions): What must be true on successful completion of a use case?
  - Main Success Scenario (or basic flow): describes typical success path (happy path).
  - Extensions (or alternative flow): Other success and failure scenarios.
  - Special Requirements: a non-functional requirement that specifically relates to a use case (example: touch-screen monitor)
  - Technology and data validation list: technical variation on how something must be done. Example: support for CC reader
  - Frequency of occurrence
  - Open Issues

See example in Craig Larman book: chapter 6, pages 68 to 69.

Flow of events Guidelines

- Describe how the use case starts and ends
- Describe the data exchanged between the actor and the use case
- Do not describe the details of the user interface unless is needed to understand the behaviour of the system
- Describe the flow of events not only the functionality. To enforce, start every action with “When the actor…”
- Describe only the events that belong to the use case and not what happens in other use cases or outside of the system
- Avoid vague terminology such as “for example”. “etc” and “information”
- Detail the flow of events. All “what’s” should be answered. Remember that test designers will use the use case document to identify test cases.

Basic and Alternate Flow of Events

- Each use-case has:
  - Basic Flow - one normal flow which is the main Success Scenario
  - Extensions (alternate flows)
    - **Extensions** - Several alternative flows that can be variants or odd cases
    - **Exceptions** – exceptional flows that handle error situations

Extensions (or alternate flows)

- Alternate paths through use case and error situations
- An extension has two parts:
  - the condition
  - the handling
- If the extension is too complex, express the extension as a separate use case

Identify the Extensions

- Work through the primary scenario (PS) and find:

1. Is there some other action that could take place at a given point in the PS?
2. Is there something that could go wrong at this point in the PS?
3. Is there something that could happen at any time during the PS?

What are scenarios?

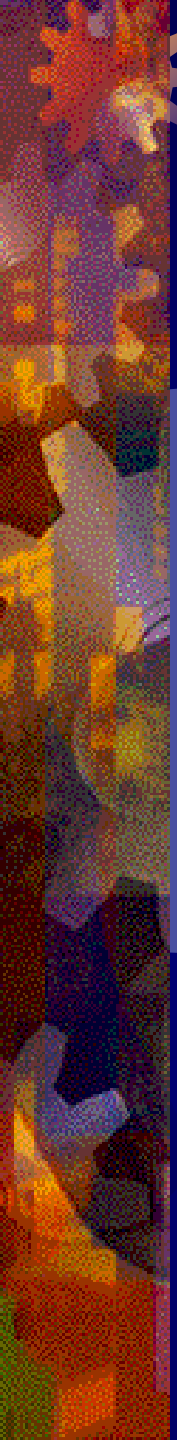
- A scenario is one flow of events through a use case
- It records:
  - An interaction between the actors
  - A validation, usually by the system
  - A state change by the system – like recording or modifying something

Primary Scenario

- Have paths through use case just like code
- Primary scenario is the path when everything goes according to plan
- Indicates how use case begins and ends
- By default the steps happen in order
- There is an appropriate level of detail
- Use case should be read and reread to ensure maximum correctness and completeness

More Secondary Scenarios

- Start with a name
- Document secondary scenarios with significant behavior
- Try to minimize the writing
- Try to document only differences between flow of events for different scenarios
- Don't need to build the whole system in English



Questions

Use Case Example
Process Sale

Process Sale UC

- Name: Process Sale
- Primary Actor: Cashier
- Stakeholders and Interests
  - Cashier: wants accurate, fast entry and no payment errors.
- Preconditions: Cashier is identified and authenticated

Process Sale UC (cont)

- Post-conditions:
  - Sale is saved
  - Tax is correctly calculated
  - Accounting and Inventory are updated
  - Commissions are recorded
  - Receipt is generated
  - Payment authorization approvals are recorded

Process Sale UC (cont)

- Main Success Scenario (BASIC FLOW)

Actor Actions	System Response
1.This use case begins when a customer arrives at POS checkout with goods and/or services to purchase	
2. Cashier starts a new sale	
3. Cashier enters the item identifier	
	4. System records sale line item and presents item description, price and running total. Price calculated from a set of price rules.
5. Cashier repeats step 3-4 until indicates done	
	6. System presents total with taxes calculates
7. Cashier tells customer the total and ask for payment	
8. Customer pays the cashier	
	9. System logs completed sale
	10. System presents receipt

Process Sale UC (cont)

- Main Success Scenario (BASIC FLOW)
- Extensions (Alternative Flows)
- Special Requirement
- Technology and Data Variations List
- Frequency of Occurrence
- Open Issues

Relating Use Cases

- When creating the use case diagram, it can be useful (in terms of comprehension and simplification) to:
  - factor out shared sub-processes
    - use the <<includes>> relationship
  - show precedence order
    - use the <<extends>> relationship

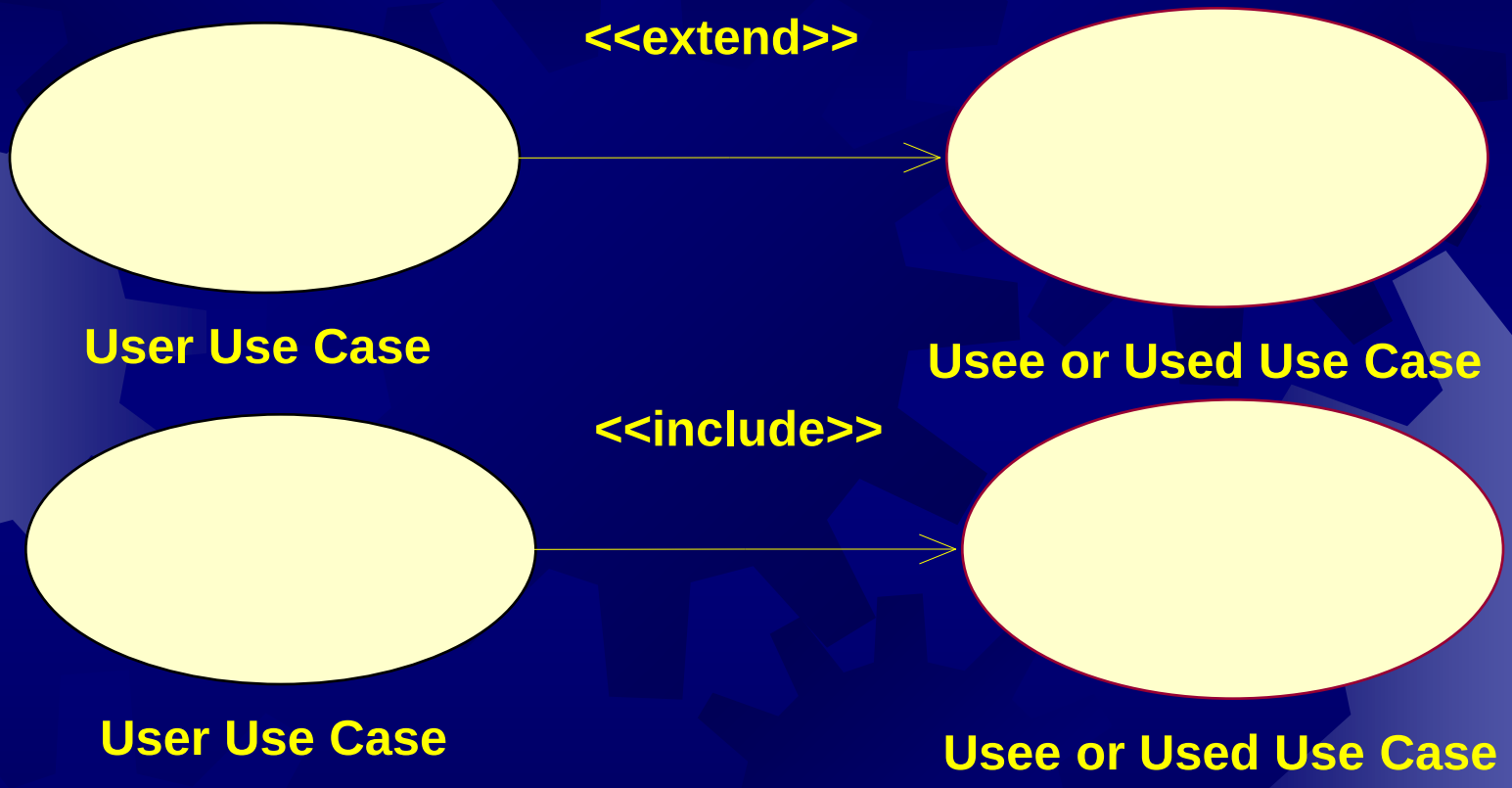
Extends

- Used for an optional sequence of events that can be included in a use case
- Can involve adding something to an existing use case
- Allows original use case to remain unchanged
- Attach at an extension point
- Very common software concept

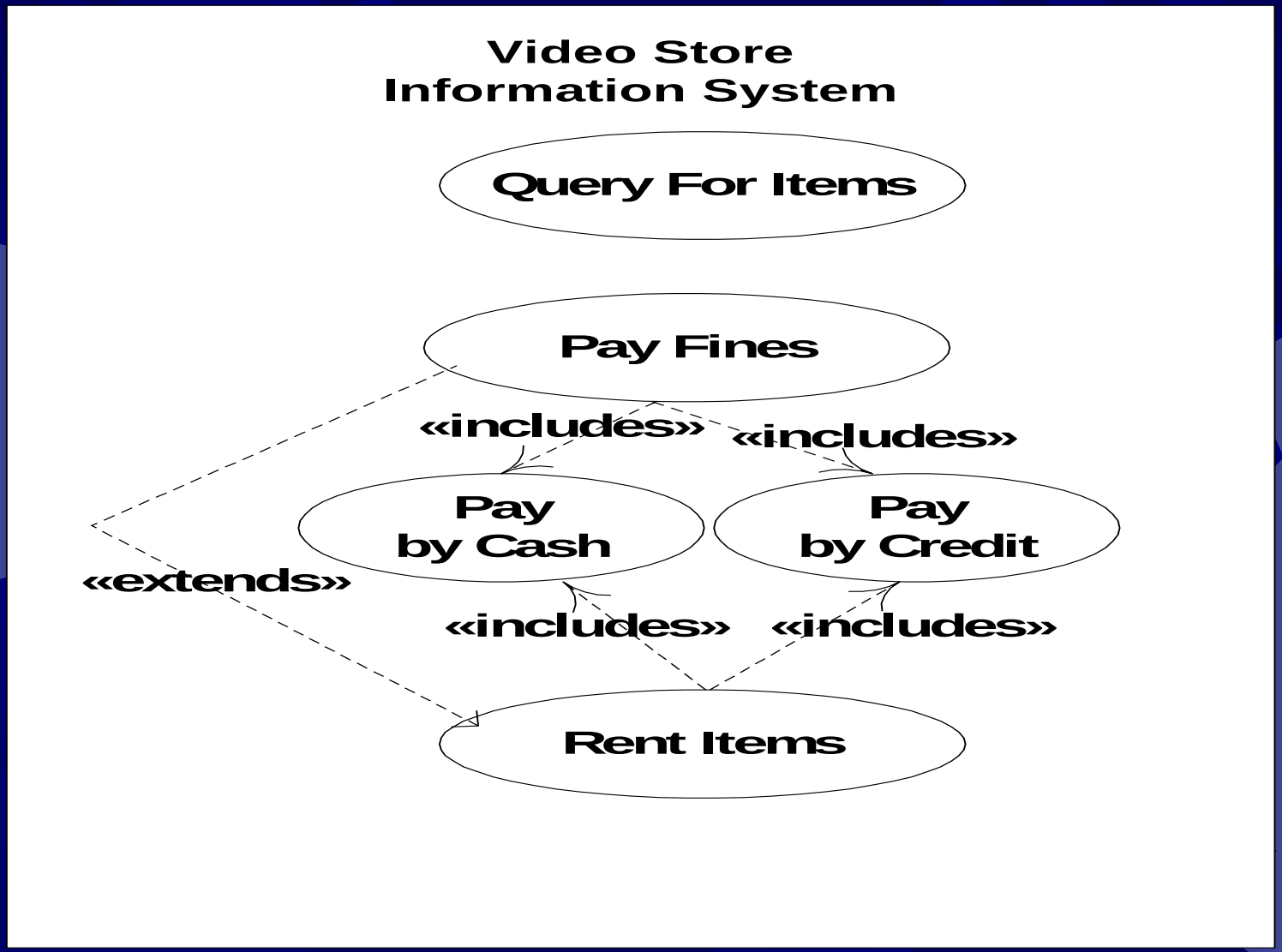
Uses or Includes

- If the same steps are found in multiple use cases
- Abstract the common behavior into a separate new use case
- Any use cases that involve these steps just need to reference the new use case
- Like abstracting repeated code into a function call

Diagramming Uses and Extends Relationships



Example of <<include>> & <<extends>>



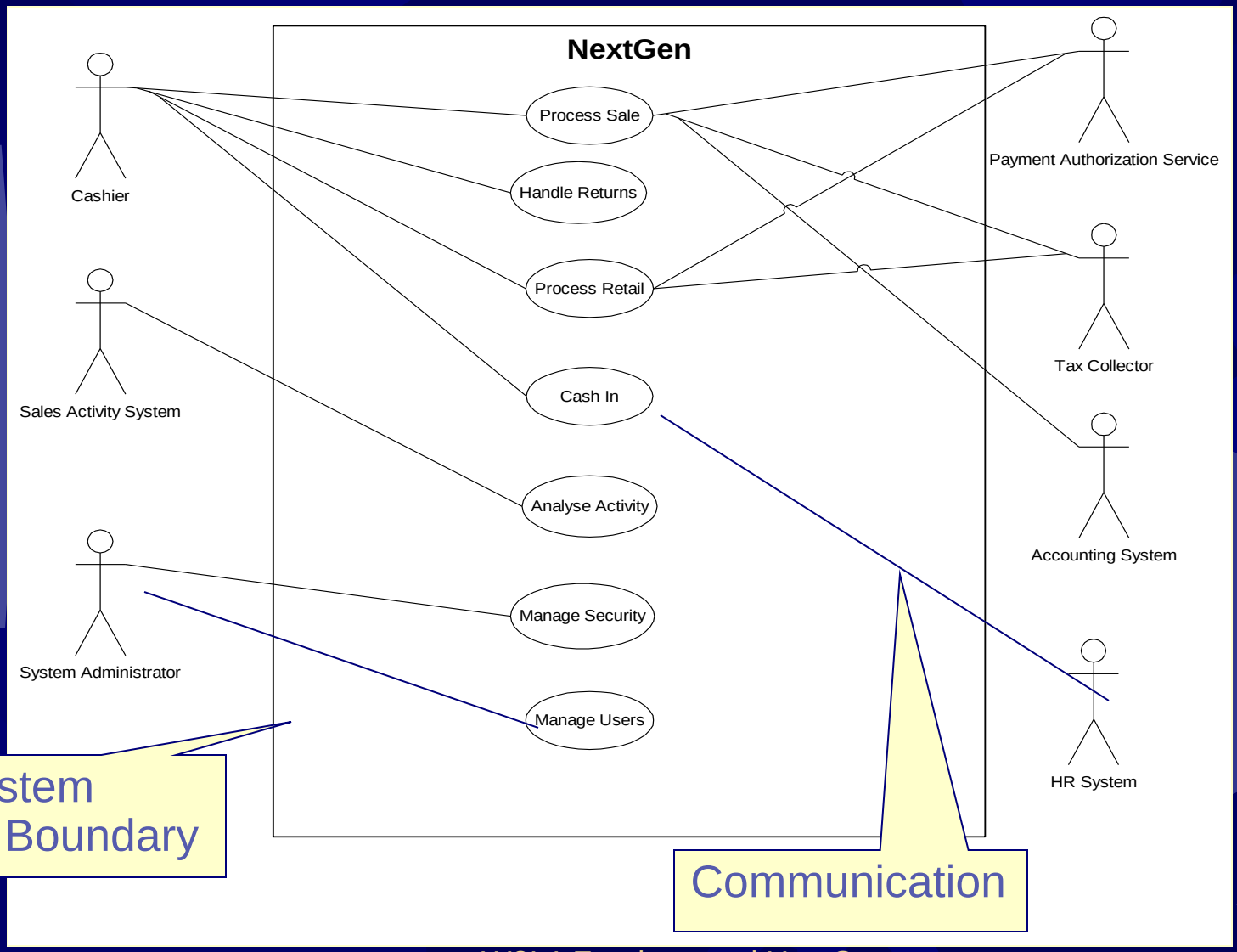
Relationship

- The arrow joining the use case in the diagram represents a directional association relationship
- Association defined as:
  - A relationship that models a bi-directional semantic connection among instances.
  - Some sort of meaningful relationship between the associating parties
- Directional association goes only one way
- With no arrow, it's a 2 way association

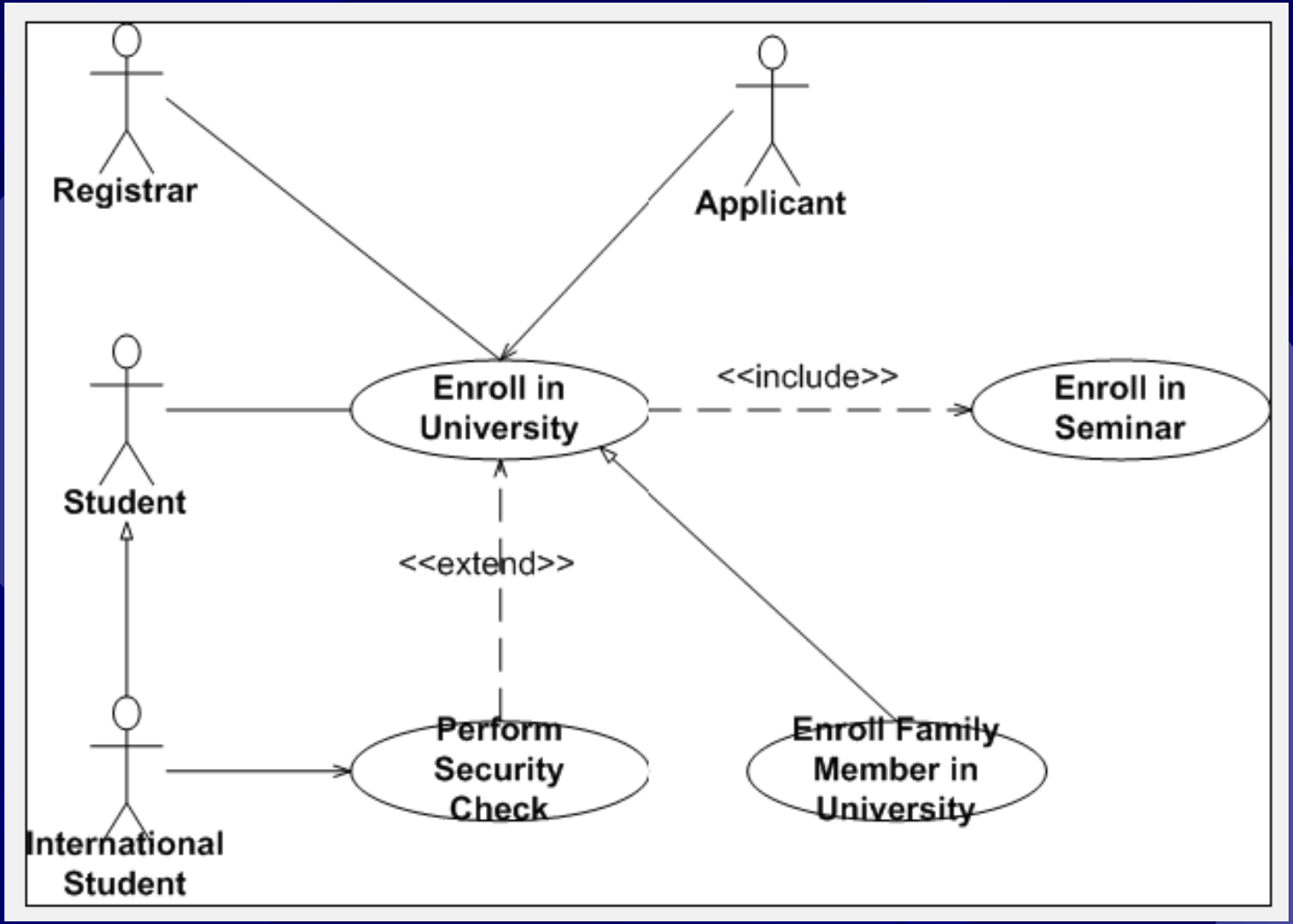
Stereotypes

- The <<extends>> and <<uses>> are called stereotypes
- Stereotype defined as
  - A meta-classification of an element. Stereotypes have semantic implications which can be specified for every specific stereotype value.
- Fancy word for a meaningful name

Partial use case diagram



Use Case Diagram - Example



Questions and Conclusions