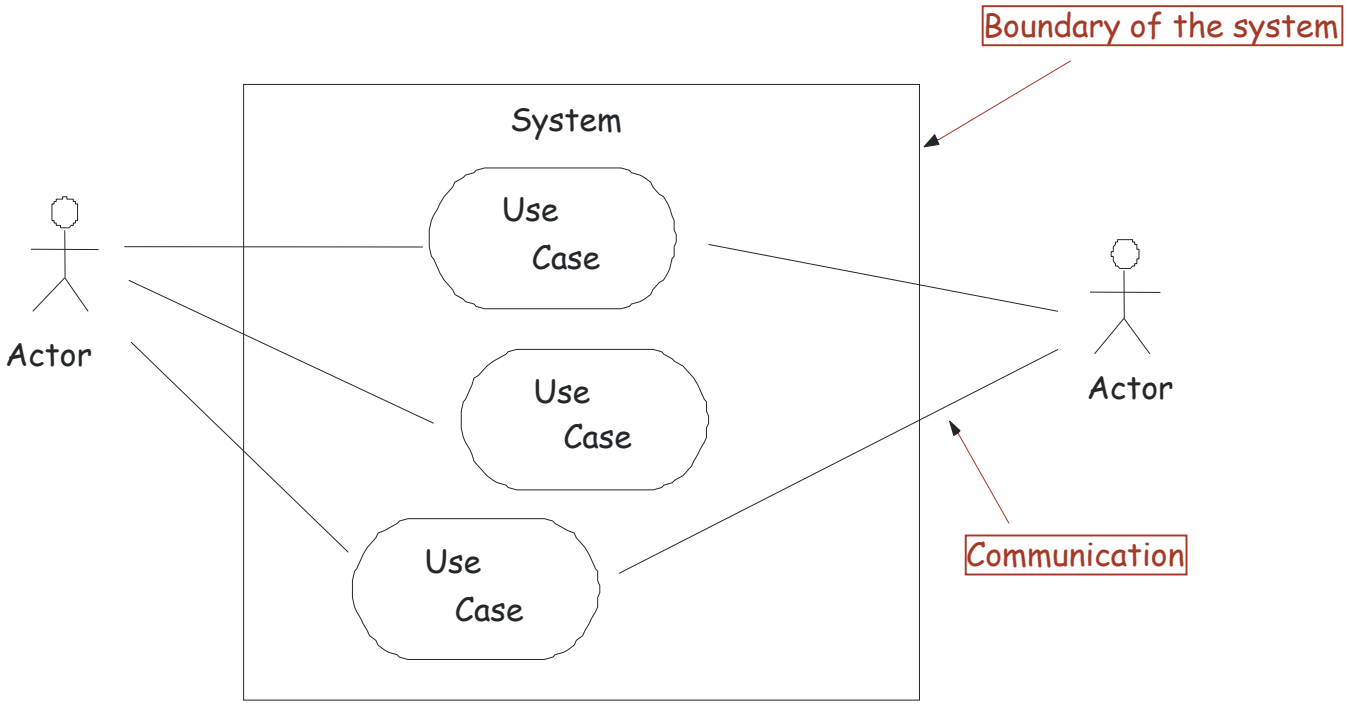
Chapter: Use cases Models

- Actors & Use Cases
- Relationships
- Use cases Diagrams
  - Context Diagram &
  - Initial Diagram
  - Specification Templates
  - Construction Process

Use Cases

- A technique to capture how an existing system works or how a future system should work
- Use to capture functional requirements.

Notation

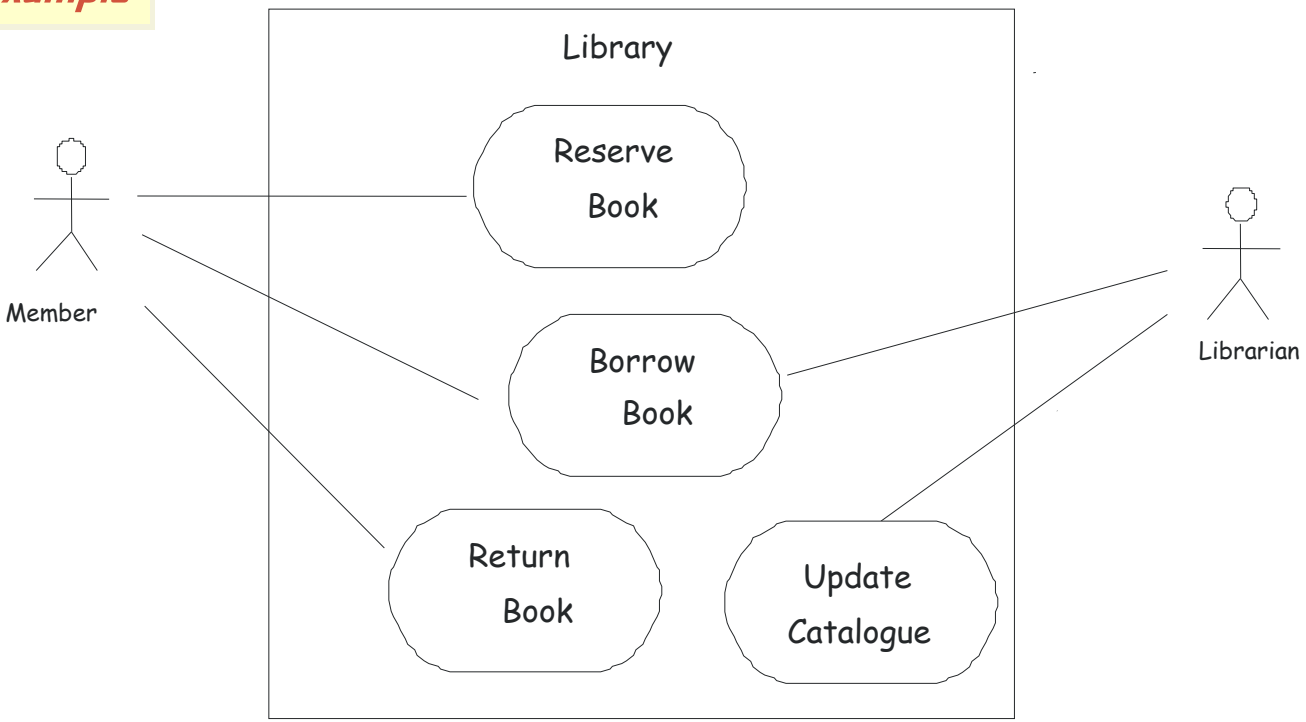


Actors and Use cases

Actor: Entity (Human, Device or another software system) exchanging information with the system

Use case: Consists of the sequence of transactions/messages that actors and the system exchange when a given functionality of the system is executed

Example



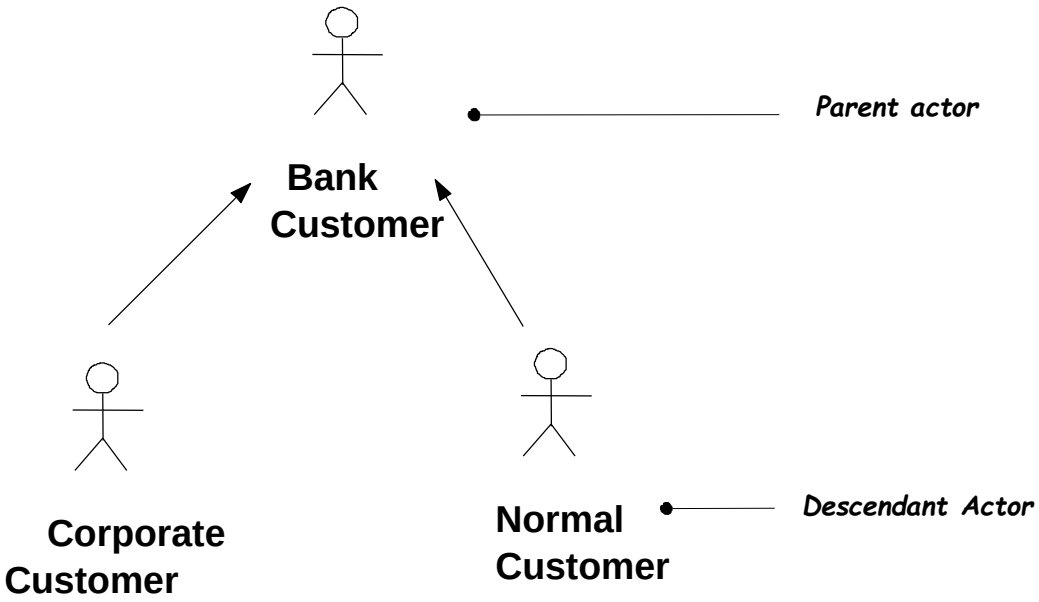
Description Templates

- Use cases are described using templates and natural language

Use Case	
Actors	
Summary	
Preconditions	
Postconditions	
Includes	
Extends	
Inherits from	
Flow of events	
Actor	*System*

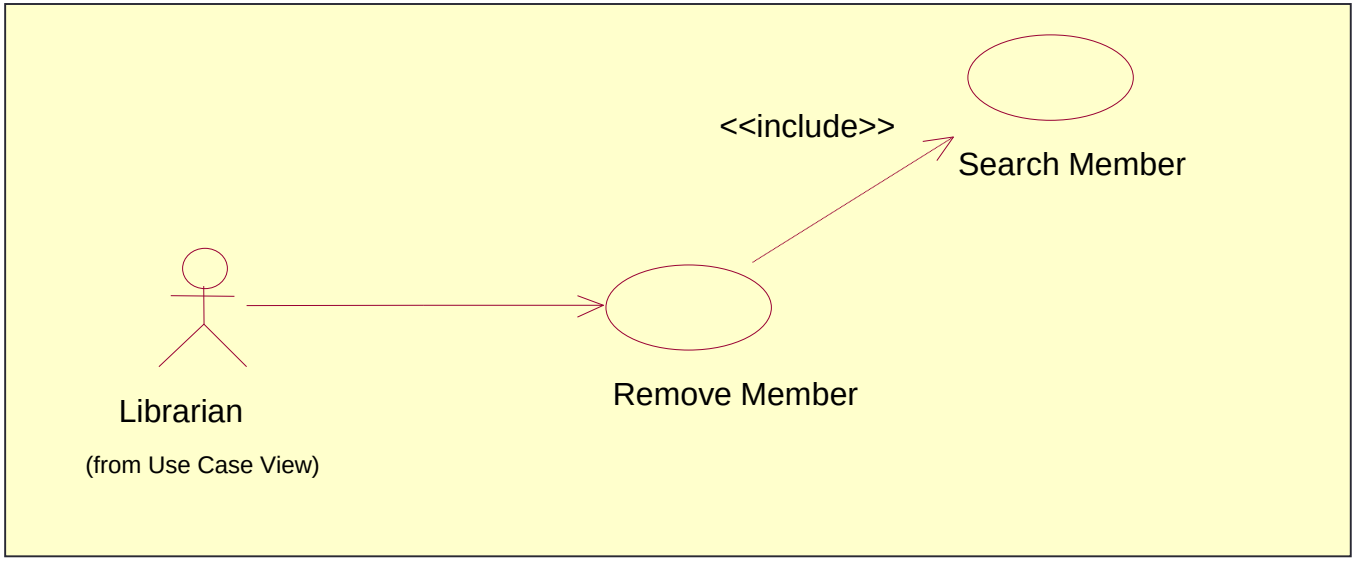
Relationships between actors- Inheritance

- The inheritance relationship indicates that the descendant actor may play all the roles of its predecessor actor.



Relationships between use cases- Inclusion

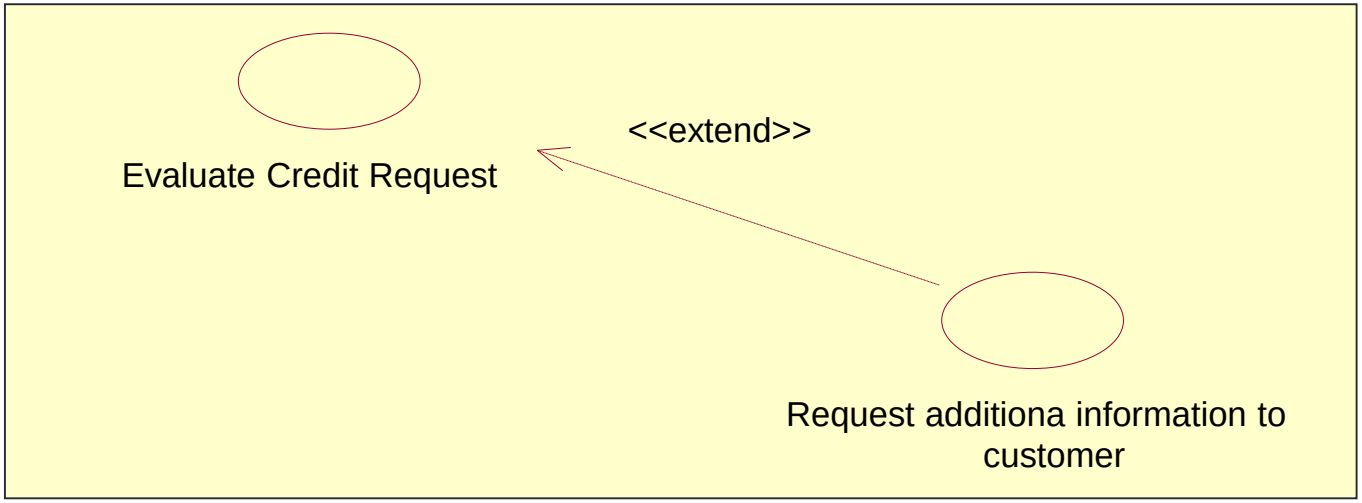
- A use case A includes a use case B, if an instance of A executes all the events that are described in B.



The instantiation of Remove Member always uses the flow of events defined in Search Member

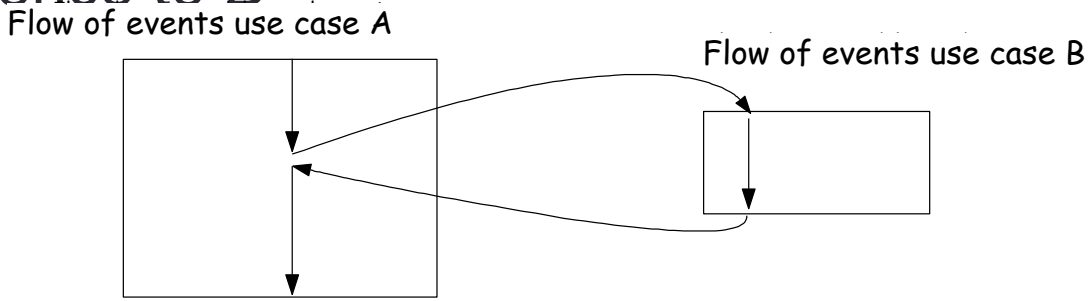
Relationships between use cases- Extension

- A use case B extends a use case A, if there is a condition in the description of A that if evaluated to true all the events described in B are executed.

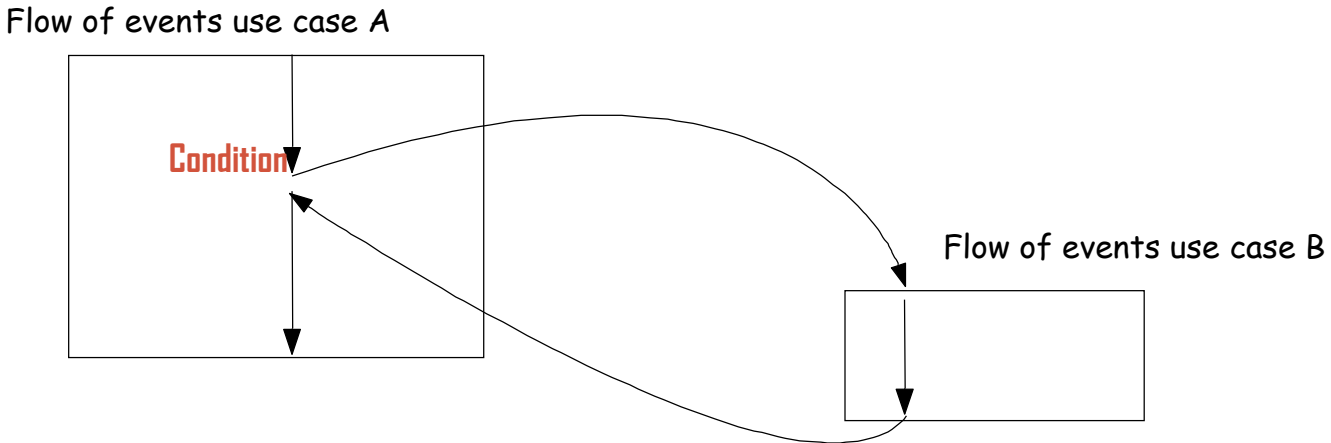


Relationships between use cases

- Inclusion: The description of use case A includes a reference to B

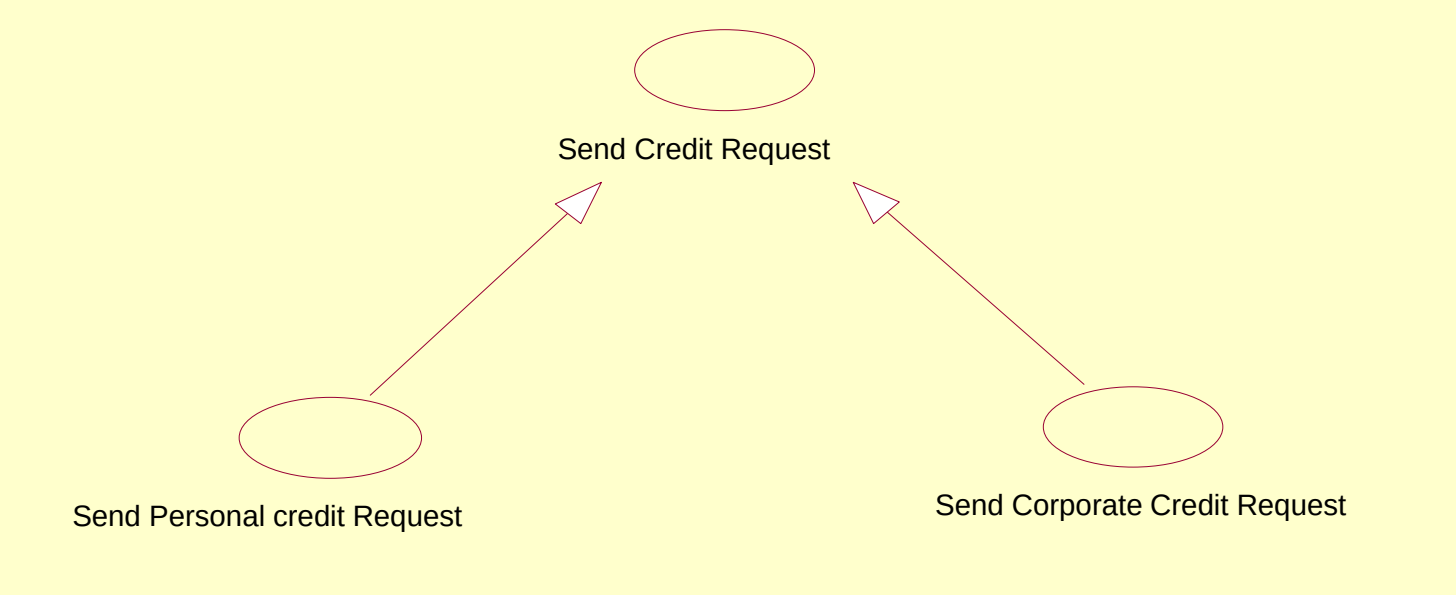


- Extension: Equivalent to an inclusion + a condition



Relationships between use cases- Inheritance

- A use case B is an specialization of another use case A, if the flow of events in B is a refinement of the flow of events in A
  - Similar to inheritance in OO (it allows the separation between a generic interaction pattern (parent use case) and a more specific case (descendant case).



Use cases Diagrams

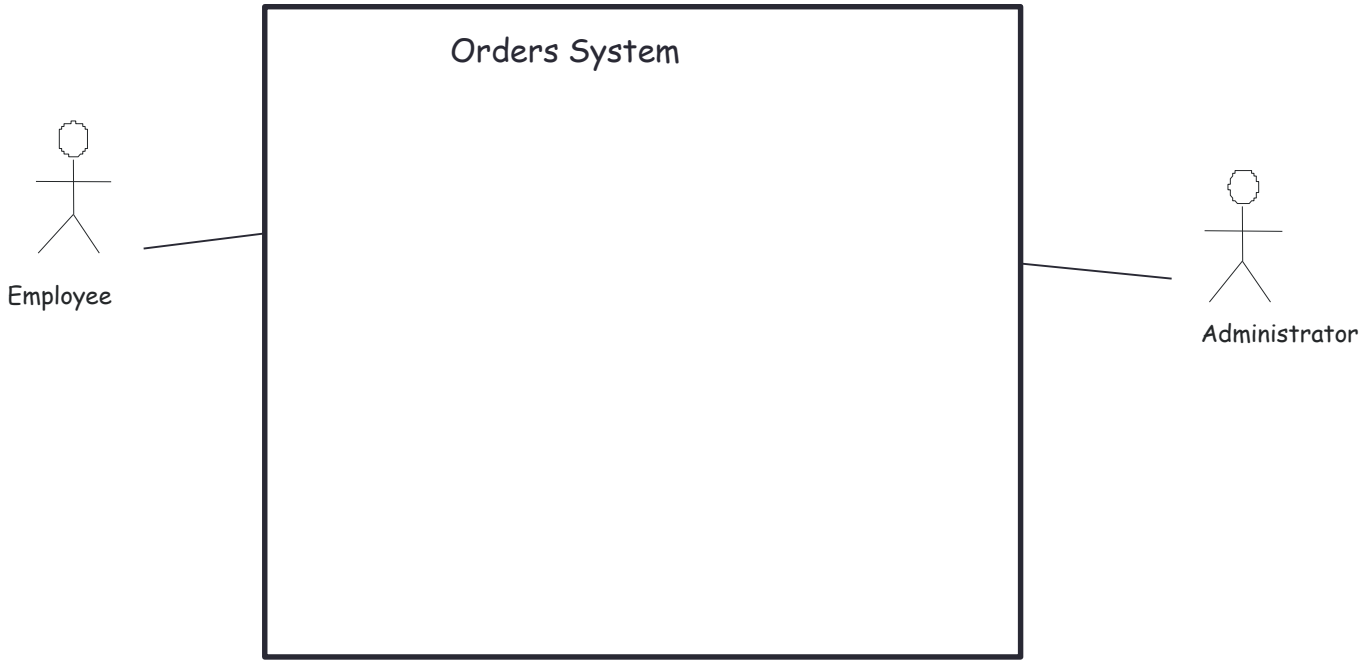
- Structured in three layers
  - Context diagram and Initial context
  - Description templates
  - Structured Diagram

IMPORTANT

Context Diagram

- It shows the boundaries of the system and the interacting actors

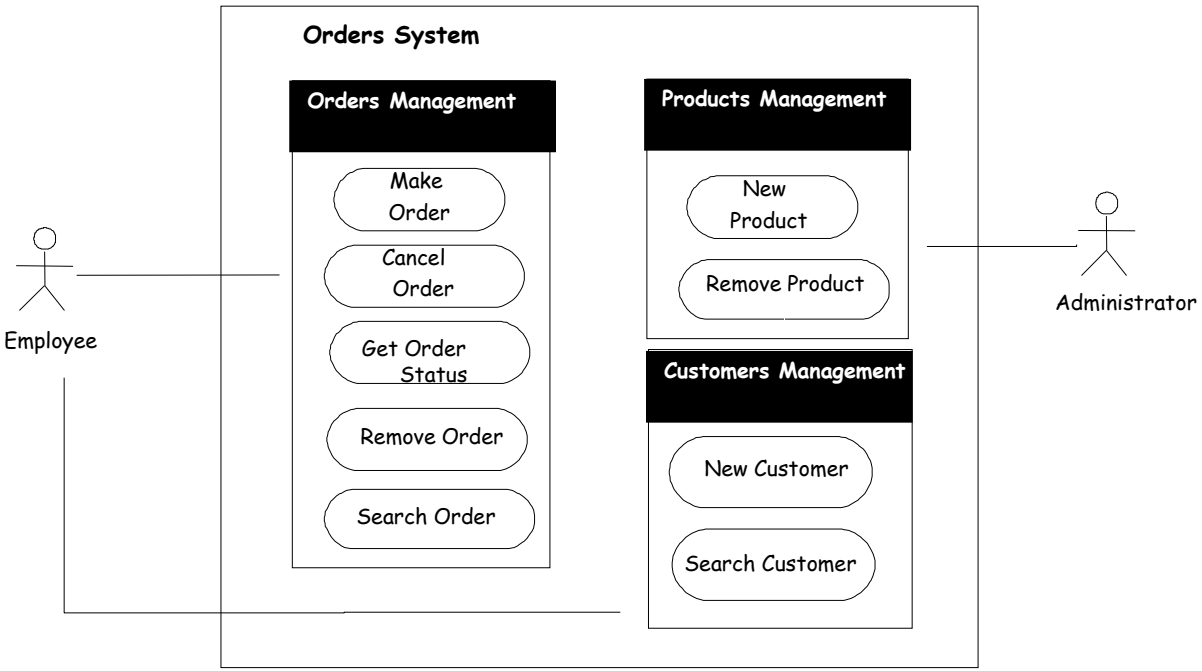
DO NOT FORGET TO DRAW THE BOX!!!!!!!!!!



Initial Diagram

- It contains a grouping of the main use cases

Like the context diagram, but displaying inside the main use cases, without extensions



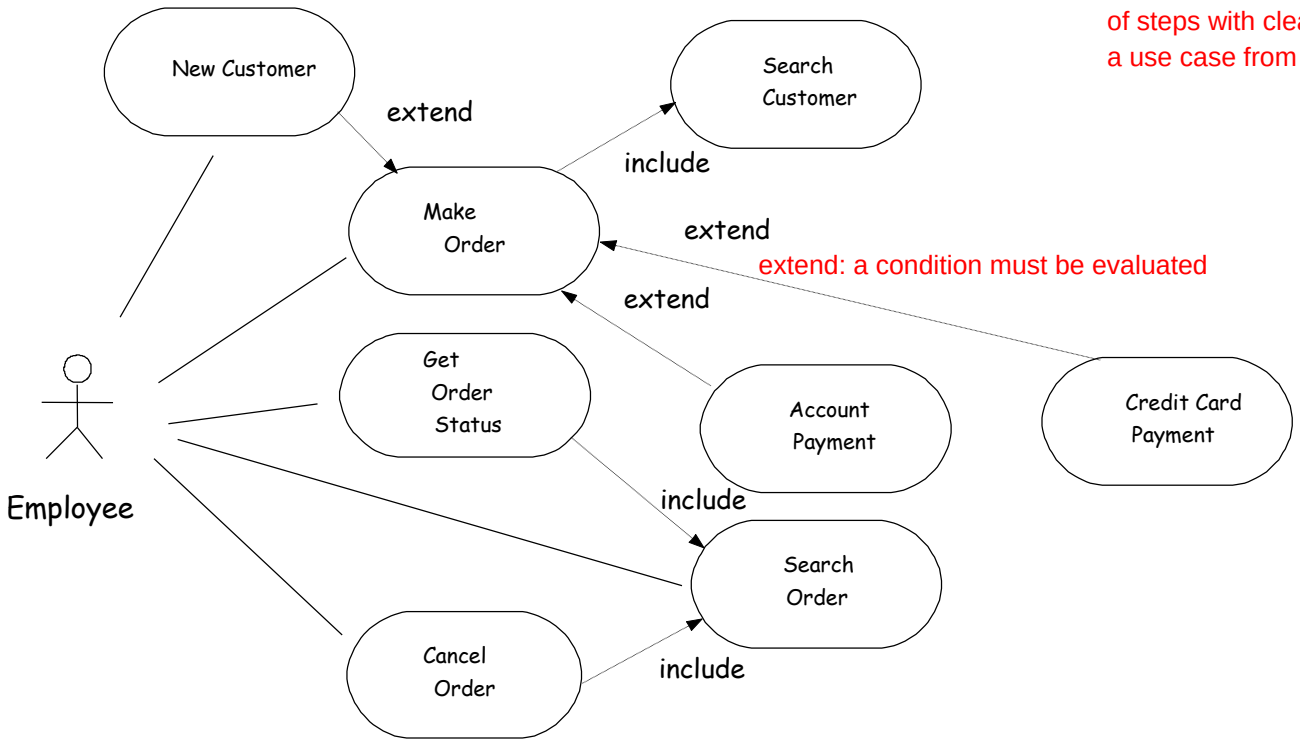
IMPORTANT

Structured Diagram

INHERITANCE: If A2 and A3 inherit A1, then A2 and A3 inherit the same use cases of A1.

A SYSTEM CANNOT BE AN ACTOR!!!!!!!!!

include: it is used to extract a collection of steps with clear intent. Aka, extract a use case from a bigger use case



(... the model is incomplete)

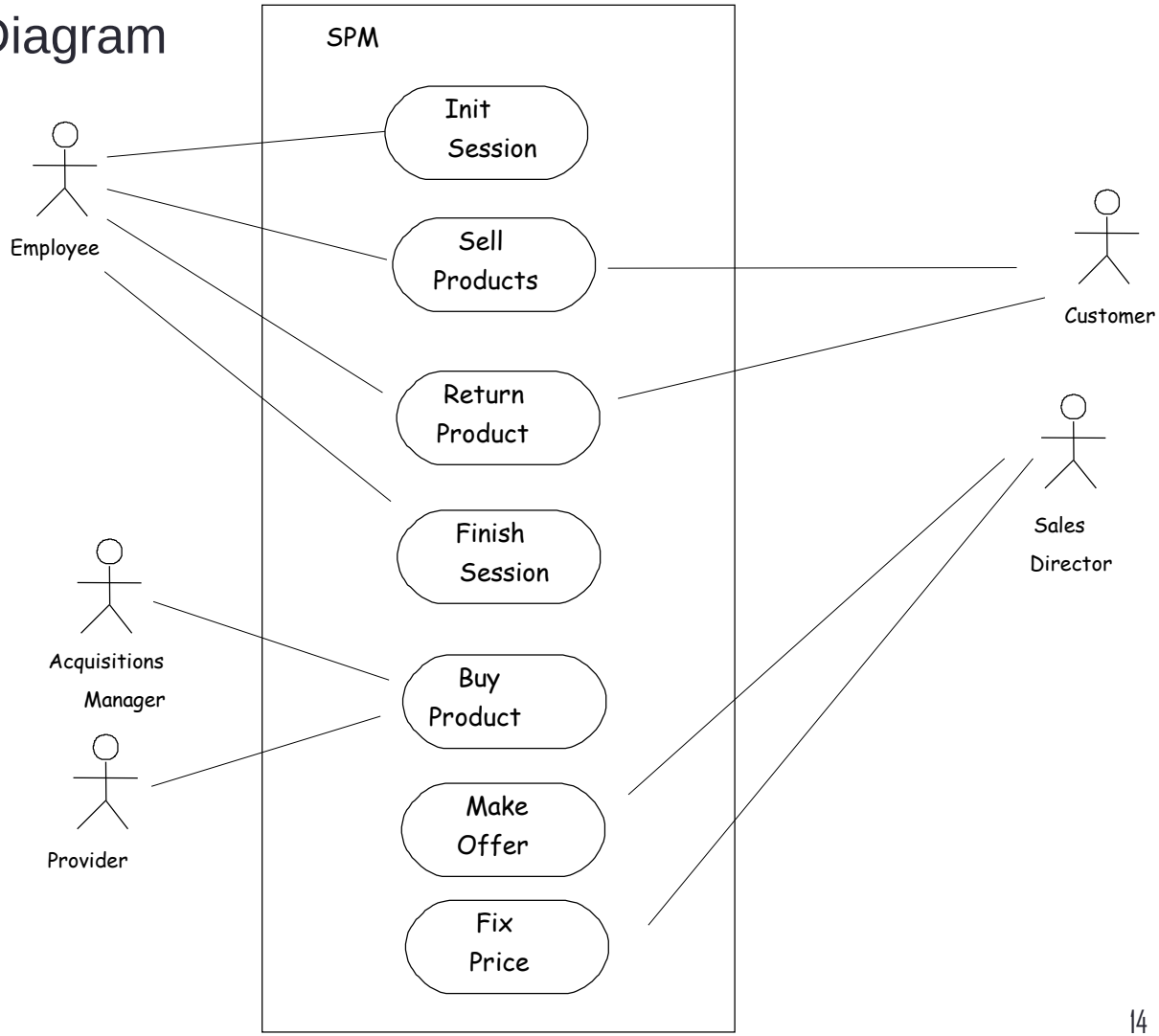
Use case: an action with main intent
it may have a precondition and/or postcondition,

Example SPM

Sales Point Machine

Context and Initial Diagram

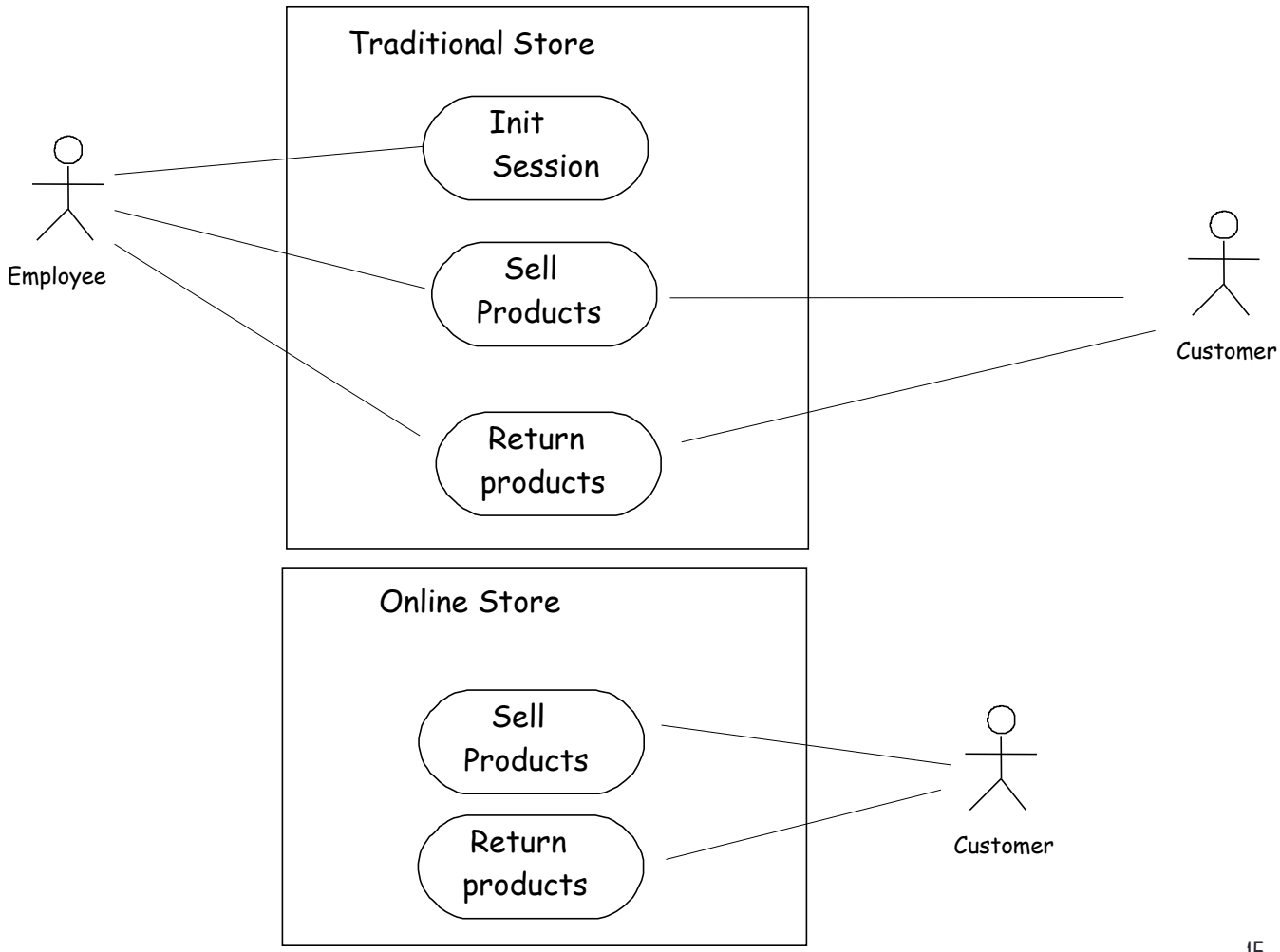
CONTEXT DIAGRAM:



. . . Example SPM

Sales Point Machine

Variations

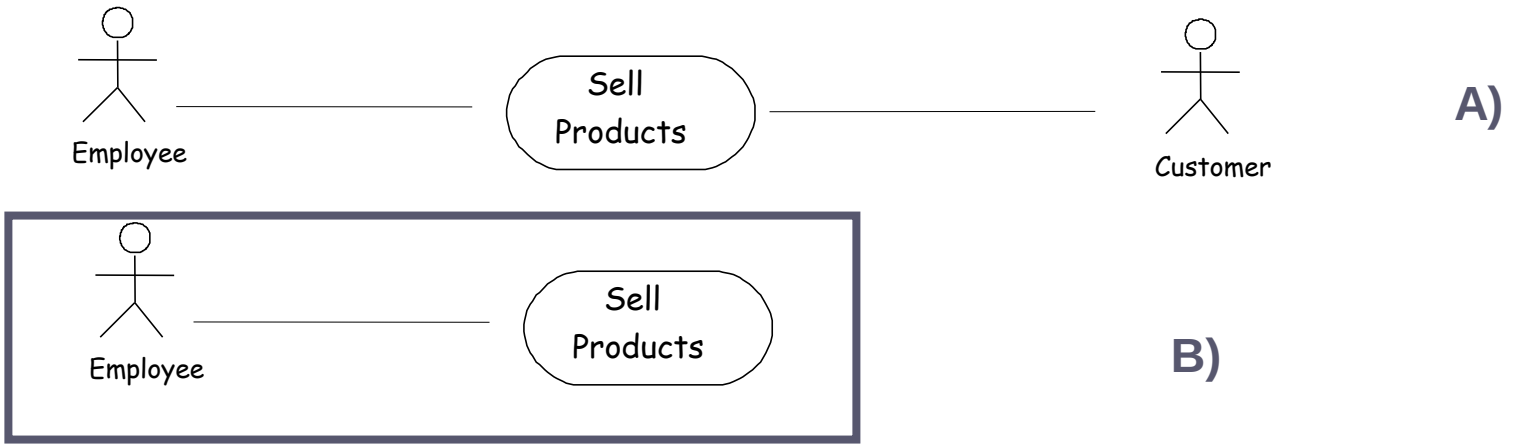


. . . Example SPM

Sales Point Machine

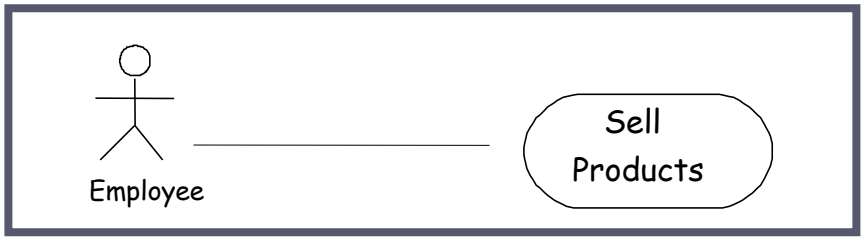
Variations on the example

If only actors interacting with the computer-based system are shown.



. . . Example SPM

Sales Point Machine



B)

Description template

Use Case	Sell Products
Actors	Employee (initiator)
Goal	Capture a sale and its cash payment
Summary	A customer arrives to the sales point with products. The employee registers products and manages payments in cash. The customer leaves with the products.
Preconditions	The employee is logged in the system.
Postconditions	The sale is stored in the system.
Includes	-
Extends	-
Inherits from	-

. . . Example SPM

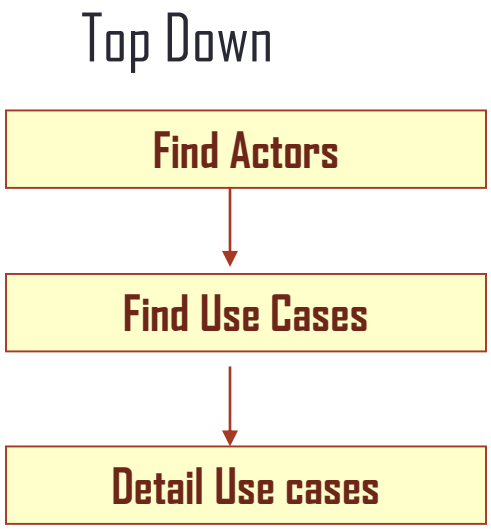
Sales Point Machine

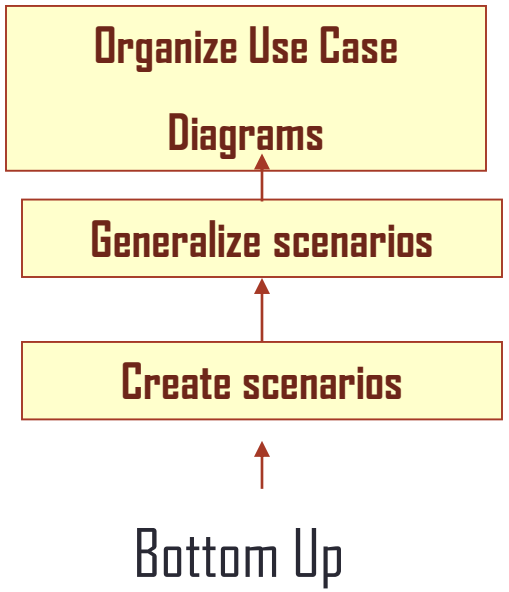
. . . Description template

	User Intentions	**System Obligations**
	1. The employee indicates a new sale starts.	2. The system records the start of a new sale
	3. The employee inserts the code of the product and the quantity	4. The system calculates the cost of the product and adds the information to the bill.
	5. The employee indicates the end of the sale	6. The system calculates and shows the total cost.
	7. The employee indicates the money received	8. The system calculates and shows the change. It prints a receipt and records the sale.
	Synchronous Extensions	
	#1. If at step 3 a code of a non-existent product is inserted the system generates an error message.	
	#2. At 7 the employee may cancel the sale.	
	Asynchronous Extensions	
	None	

Building the diagram

- Top-down technique
- Bottom-up technique





Building the diagram

Rules to find Actors

- Users play roles when interacting with the system
- A user may correspond to many roles

- Any group or individual in one of the following categories:
  - Who will use the system?
  - Who will install the system?
  - Who will maitain the system?
  - Who will switch off the system?
  - What other systems will communicate with this one?
  - Who gets information?
  - Who provides information?

Building the diagram.

Rules to identify Use Cases

- Paying attention to actors
  - What are the tasks required by actors from the system?.
  - Will an actor be able to create, store, change or remove information from the system?.
  - Will an actor inform to the system about changes occuring outside?.
  - Will any actor be informed about state changes of the system?.

The answers to these questions represent flows of events that are associated to potential candidate use cases.