Edith Cowan University
School of Science

CSI6207
Systems Analysis and Database Design

Process Modeling and Use Case Diagrams

Contents

- Use Case Description
- Use Case Diagrams
- Activity Diagram
- System Sequence Diagram (SSD)
- CRUD

Learning Objectives

- Write fully developed use case descriptions
- Develop use case diagrams to model use cases
- Develop activity diagrams to model flow of activities
- Develop system sequence diagrams
- Use the CRUD technique to validate use cases
- Explain how use case descriptions and UML diagrams work together to define functional requirements

Use Case Descriptions

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- *Brief Use Case description*
  - Suitable for simple use cases

Use case	Brief use case description
Create customer account	User/actor enters new customer account data, and the system assigns account number, creates a customer record, and creates an account record.
Look up customer	User/actor enters customer account number, and the system retrieves and displays customer and account data.
Process account adjustment	User/actor enters order number, and the system retrieves customer and order data; actor enters adjustment amount, and the system creates a transaction record for the adjustment.

Use Case Descriptions

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- *Fully developed use case description*
  - For more complex use cases
  - Typical use case description templates include
    - Use case name
    - Scenario (if needed)
    - Triggering event
    - Brief description
    - Actors
    - Related use cases
    - Stakeholders
    - Preconditions
    - Postconditions
    - Flow of activities/typical course of events
    - Exception conditions

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- Use case name
  - Verb-noun
  - Should be identical to the name of the use case on the diagram
- Scenario (if needed)
  - A use case can have more than one scenario (special case or more specific path)
- Triggering event
  - Based on event decomposition technique

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- Brief description
  - Written previously when use case was identified (Slide#4)
- Actors
  - One or more users from use case diagrams
  - The name should be identical to one of the actor names on the use case diagram
  - If you have more than one actor, make sure that they all do something-role playing the flow of activities/typical course of events is particularly good for detecting this problem

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

- Related use cases
  - <<Includes>> <the name of a use case that this use case includes (on the use case diagram)>
    - i.e. the child use case
  - <<Extends>> <the name of a use case for which this use case is the base>
    - i.e. the extension
  - This should match the flow of activities section of a use case description and the use case diagram
- Stakeholders
  - Anyone with an interest in the use case
  - Some books consider them as secondary actors

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

- Preconditions
  - What must be true before the use case begins
  - What can we assume about the system state before the use case starts?
  - Examples might be:
    - The user has been authenticated
    - The customer has chosen a product
    - The customer has registered with the system
  - If there aren't any, write "Nil" or "None" (but preconditions are helpful, so use them)

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

- Postconditions
  - What must be true when the use case is completed
  - Examples:
    - User is logged in
    - The customer is sent his/her registration details
  - Given that the point of a use case is to "yield an observable result of value to an actor", then there must be **at least one postcondition**

Use Case Description Details

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- Flow of activities/ Typical Course of Events
  - The activities that go on between actor and the system
  - Some examples:
    - "User enters name and address."
    - "At any time, user can request the money back."
    - "The system verifies that the name and account are current."
    - "The system updates the customer's balance to reflect the charge."

- Exception conditions
  - Where and what can go wrong
  - It is used to model:
    - Failures
    - Errors
    - Rare behaviour

Some Examples of Fully Developed Use Cases from RMO CSMS System

Fully Developed Use Case Description

Use case: *Create customer account*

Use case name:	*Create customer account.*	
Scenario:	Create online customer account.	
Triggering event:	New customer wants to set up account online.	
Brief description:	Online customer creates customer account by entering basic information and then following up with one or more addresses and a credit or debit card.	
Actors:	Customer.	
Related use cases:	Might be invoked by the *Check out shopping cart* use case.	
Stakeholders:	Accounting, Marketing, Sales.	
Preconditions:	Customer Account subsystem must be available. Credit/debit authorization services must be available.	
Postconditions:	Customer must be created and saved. One or more Addresses must be created and saved. Credit/debit card information must be validated. Account must be created and saved. Address and Account must be associated with Customer.	
Flow of activities:	**Actor**	**System**
	1. Customer indicates desire to create customer account and enters basic customer information. 2. Customer enters one or more addresses. 3. Customer enters credit/debit card information.	1.1 System creates a new customer. 1.2 System prompts for customer addresses. 2.1 System creates addresses. 2.2 System prompts for credit/debit card. 3.1 System creates account. 3.2 System verifies authorization for credit/debit card. 3.3 System associates customer, address, and account. 3.4 System returns valid customer account details.
Exception conditions:	1.1 Basic customer data are incomplete. 2.1 The address isn't valid. 3.2 Credit/debit information isn't valid.	

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

Another Fully Developed Use Case Description

Example

Use case:
Ship items

Use case name:	*Ship items*.	
Scenario:	Ship items for a new sale.	
Triggering event:	Shipping is notified of a new sale to be shipped.	
Brief description:	Shipping retrieves sale details, finds each item and records it is shipped, records which items are not available, and sends shipment.	
Actors:	Shipping clerk.	
Related use cases	None.	
Stakeholders:	Sales, Marketing, Shipping, warehouse manager.	
Preconditions:	Customer and address must exist. Sale must exist. Sale items must exist.	
Postconditions:	Shipment is created and associated with shipper. Shipped sale items are updated as shipped and associated with the shipment. Unshipped items are marked as on back order. Shipping label is verified and produced.	
Flow of activities:	**Actor**	**System**
	1. Shipping requests sale and sale item information.	1.1 System looks up sale and returns customer, address, sale, and sales item information.
	2. Shipping assigns shipper.	2.1 System creates shipment and associates it with the shipper.
	3. For each available item, shipping records item is shipped.	3.1 System updates sale item as shipped and associates it with shipment.
	4. For each unavailable item, shipping records back order.	4.1 System updates sale item as on back order.
	5. Shipping requests shipping label supplying package size and weight.	5.1 System produces shipping label for shipment. 5.2 System records shipment cost.
Exception conditions:	2.1 Shipper is not available to that location, so select another. 3.1 If order item is damaged, get new item and updated item quantity. 3.1 If item bar code isn't scanning, shipping must enter bar code manually. 5.1 If printing label isn't printing correctly, the label must be addressed manually.	

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

Fully Developed Use Case Description
Ship items (part 1)

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

Use case name:	*Ship items*.
Scenario:	Ship items for a new sale.
Triggering event:	Shipping is notified of a new sale to be shipped.
Brief description:	Shipping retrieves sale details, finds each item and records it is shipped, records which items are not available, and sends shipment.
Actors:	Shipping clerk.
Related use cases	None.
Stakeholders:	Sales, Marketing, Shipping, warehouse manager.
Preconditions:	Customer and address must exist. Sale must exist. Sale items must exist.
Postconditions:	Shipment is created and associated with shipper. Shipped sale items are updated as shipped and associated with the shipment. Unshipped items are marked as on back order. Shipping label is verified and produced.

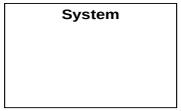
Fully Developed Use Case Description
Ship items (part 2)

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

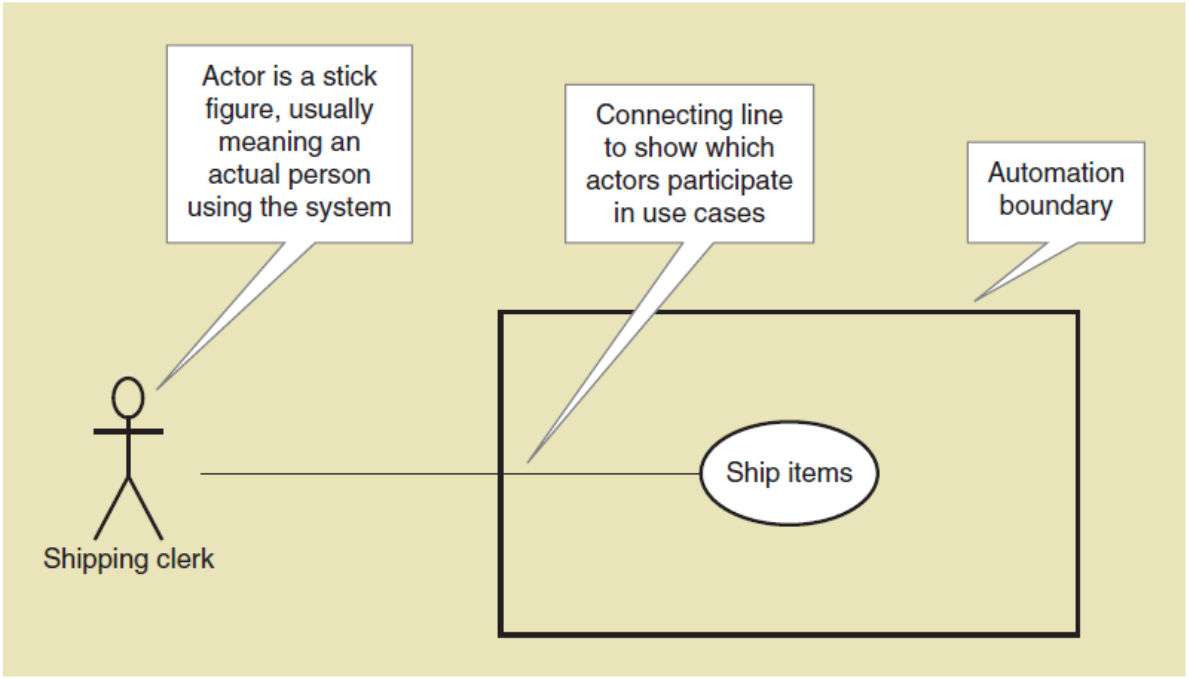
Flow of activities:	Actor	System
	1. Shipping requests sale and sale item information.	1.1 System looks up sale and returns customer, address, sale, and sales item information.
	2. Shipping assigns shipper.	2.1 System creates shipment and associates it with the shipper.
	3. For each available item, shipping records item is shipped.	3.1 System updates sale item as shipped and associates it with shipment.
	4. For each unavailable item, shipping records back order.	4.1 System updates sale item as on back order.
	5. Shipping requests shipping label supplying package size and weight.	5.1 System produces shipping label for shipment. 5.2 System records shipment cost.
Exception conditions:	2.1 Shipper is not available to that location, so select another. 3.1 If order item is damaged, get new item and updated item quantity. 3.1 If item bar code isn't scanning, shipping must enter bar code manually. 5.1 If printing label isn't printing correctly, the label must be addressed manually.	

Use Case Diagrams

- **Use case diagram**— A UML model used to graphically show uses cases and their relationships to actors
- The four components of use case models are:
  - **System boundary** – the boundary between the computerized portion of the application and the users who operate the application. Usually represented by a box.
  - **Actors** – roles played by people or things that use the system. Usually, represented by a stick figure
  - **Use cases** – things that the actors can do with the system. Represented by an oval
  - **Relationships** – meaningful relationships between actors and use cases. Represented by a line

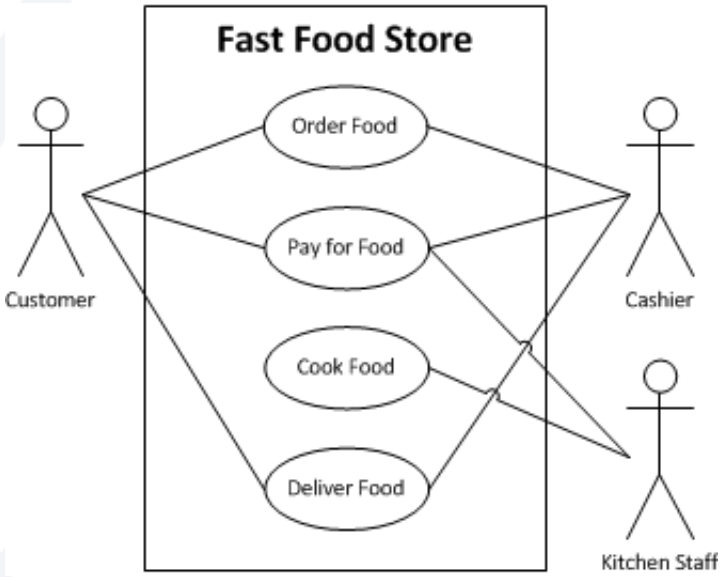


Use Case Diagrams Symbols…



Parts of the model: System Boundary

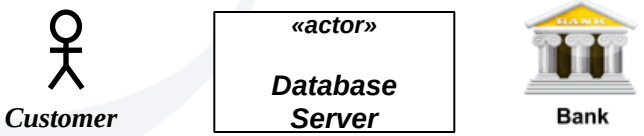
- Defines what is *part* of the system (inside the system boundary) and what is *external* to the system (outside)
  - The subject is defined by who or what uses the system (actors) and what benefits the systems offers to them (use cases)
  - May be hard to define until all actors and use cases are identified, but it is important to know the boundary of the system



Parts of the model: Actors

- Actors are outside the system boundary
- An actor specifies **a role** that an external entity adopts when interacting with the system *directly*
- Actors are roles that can be played by:
  - Human
  - Other systems/sub-systems
  - Hardware devices

- Actors are usually represented by a stick figure, however, other non-human icons representing the actor are also allowed:



Parts of the model: Actors…

- To identify actors, consider who or what interacts with a system. Generalise specific interactions to identify roles

- Ask questions such as:
  - Who/What uses the system?
  - What roles do they play in the interaction?
  - Who/What installs, starts, shuts down and maintains the system?
  - What other systems interact with this system?
  - Who/What gets and provides information to the system?
  - Does anything happen at a fixed time?
    - "Time" can be an actor, e.g. to represent an automatic daily backup

Parts of the model: Relationships

- Association (between use cases and actors)
- Stereotypes
  - <<Include>> (between use cases)
  - <<Extend>> (between use cases)
- Generalisation (between use cases or between actors)

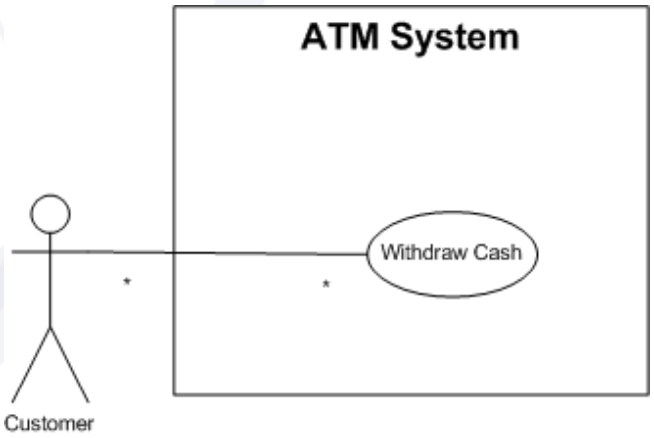
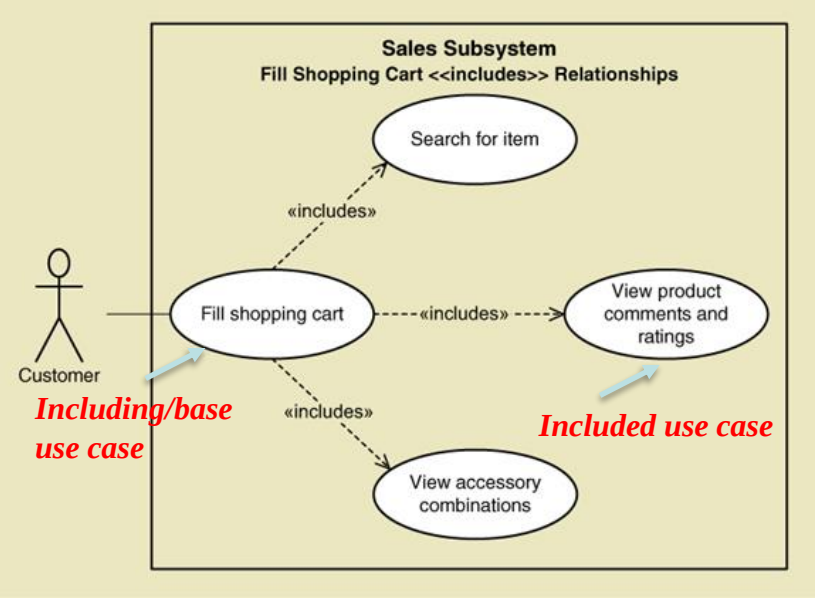


Fig. Association

The «include» Stereotype

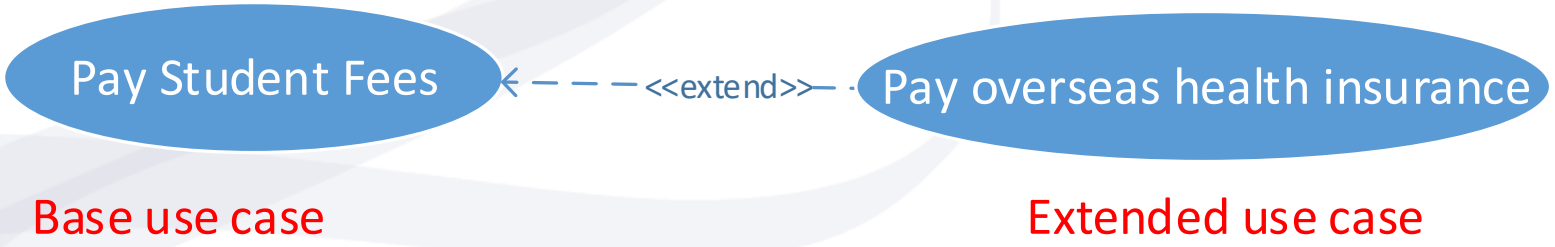
- A relationship between use cases where one use case is stereotypically included within the other use case— like a called subroutine.
- In other word, a relationship in which one use case (the base use case) includes the functionality of another use case (the included use case or subroutine)



Important-notice the direction and style of the arrow (towards subroutine)

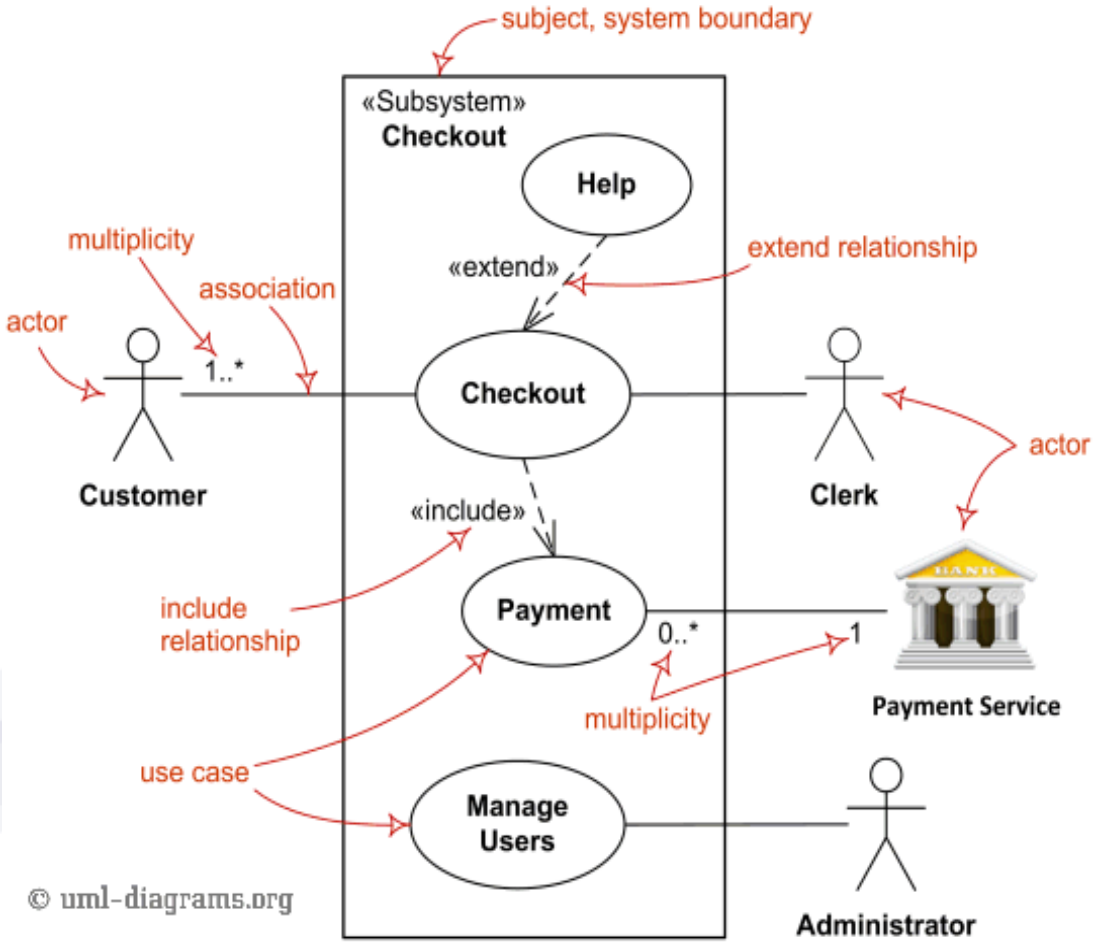
The «extend» Stereotype

- Use when there is some additional behavior that should be added, possibly conditionally, to the behavior defined in another use case
- Notice the direction and style of the arrow



Putting it all together

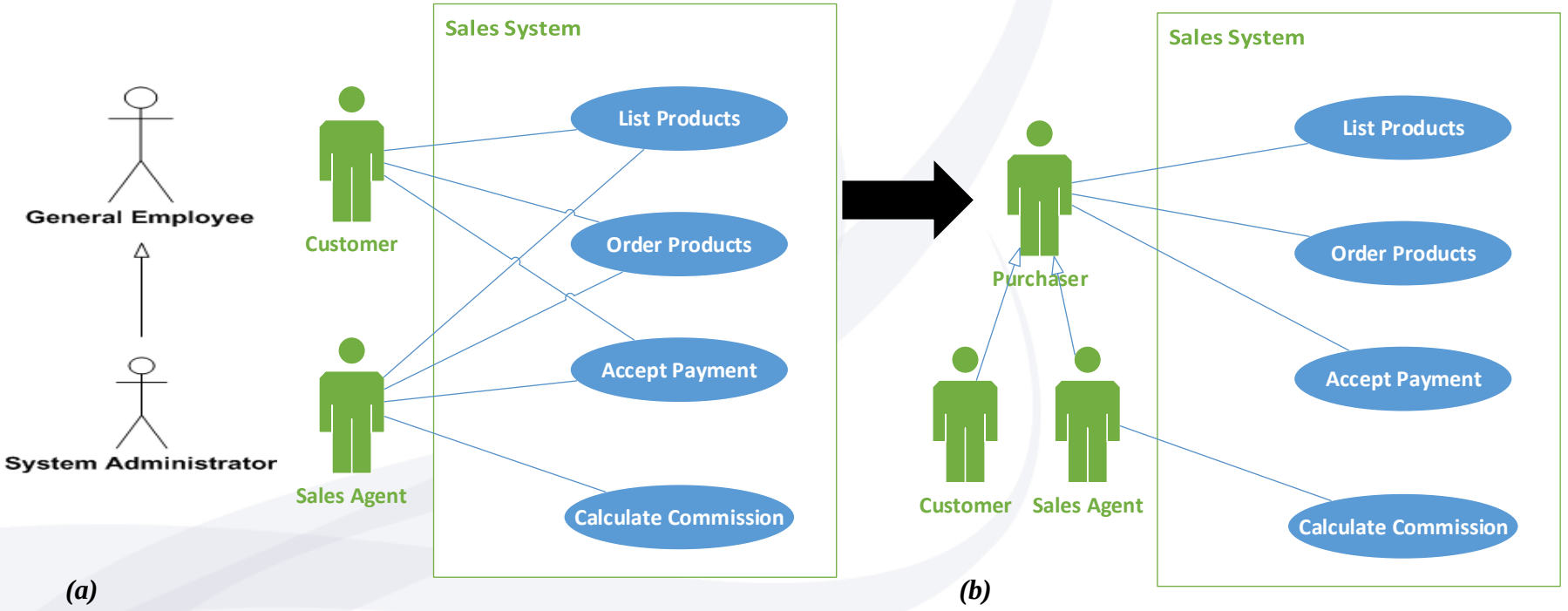
(http://www.uml-diagrams.org/use-case-diagrams/use-case-diagram-elements.png)



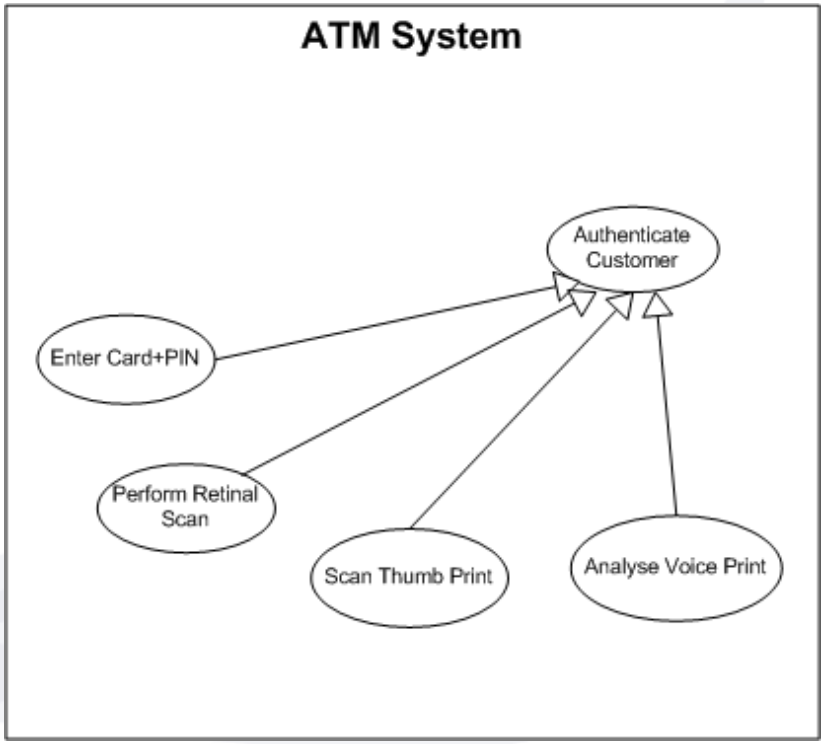
Generalisation

- A generalisation relationship is one in which one model element (the child) is based on another model element (the parent)
- The child receives all of the attributes, operations, and relationships that are defined in the parent
- Note: We can generalise actors or use cases

Generalisation of Actors



Generalisation of Use Cases



Use Case Diagrams: Steps

1. Identify all the stakeholders and users who would benefit by seeing a use case diagram.
2. Determine what each stakeholder or user needs to review in a use case diagram: each subsystem, for each type of user, for use cases that are of interest.
3. For each potential communication need, select the use cases and actors to show and draw the use case diagram. There are many software packages that can be used to draw use case diagrams (including UML use case diagram in MS Visio).
4. Define the relationships (e.g. <<include>>, <<extend>>, generalization etc) among use cases or actors if any.
5. Carefully name each use case diagram and then note how and when the diagram should be used to review use cases with stakeholders and users

(Note that, above steps assumes that all functional requirements and use cases are already identified)

Some Don'ts in Use case

- Do not confuse a role that something plays in the context of the system with the thing itself. Always think:

  "What role does this thing play with regards to the system?

- Only use the advanced features when they simplify and clarify your model – the aim is model system requirements
  - Do not break high level use cases into series of primitive use cases ("functional decomposition"). Does not model requirements well
  - Hierarchies of use cases and actors are rarely more than 2 levels deep

- Use case models should be readable by stakeholders, who will struggle with generalisations, «include» and «extend»
- Do not use system as an actor.

Activity Diagrams

(Modeling the flow of activities and exception conditions)

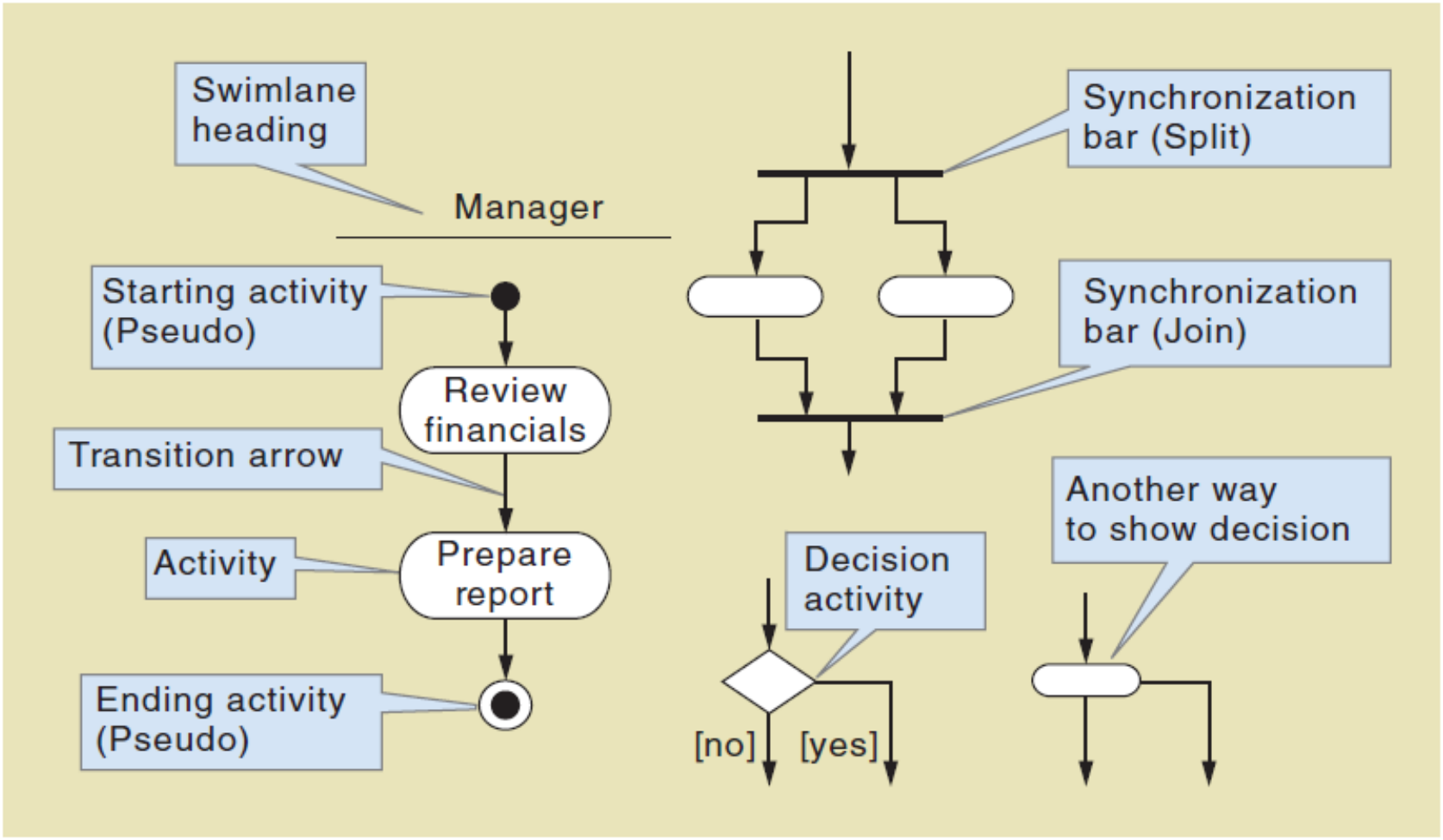
Documenting Workflows with Activity Diagrams

- **Workflow**– sequence of processing steps that completely handles one business transaction or customer request

- **Activity Diagram**– describes user (or system) activities, the person who does each activity, and the sequential flow of these activities
  - Useful for showing a graphical model of a workflow
  - A UML diagram

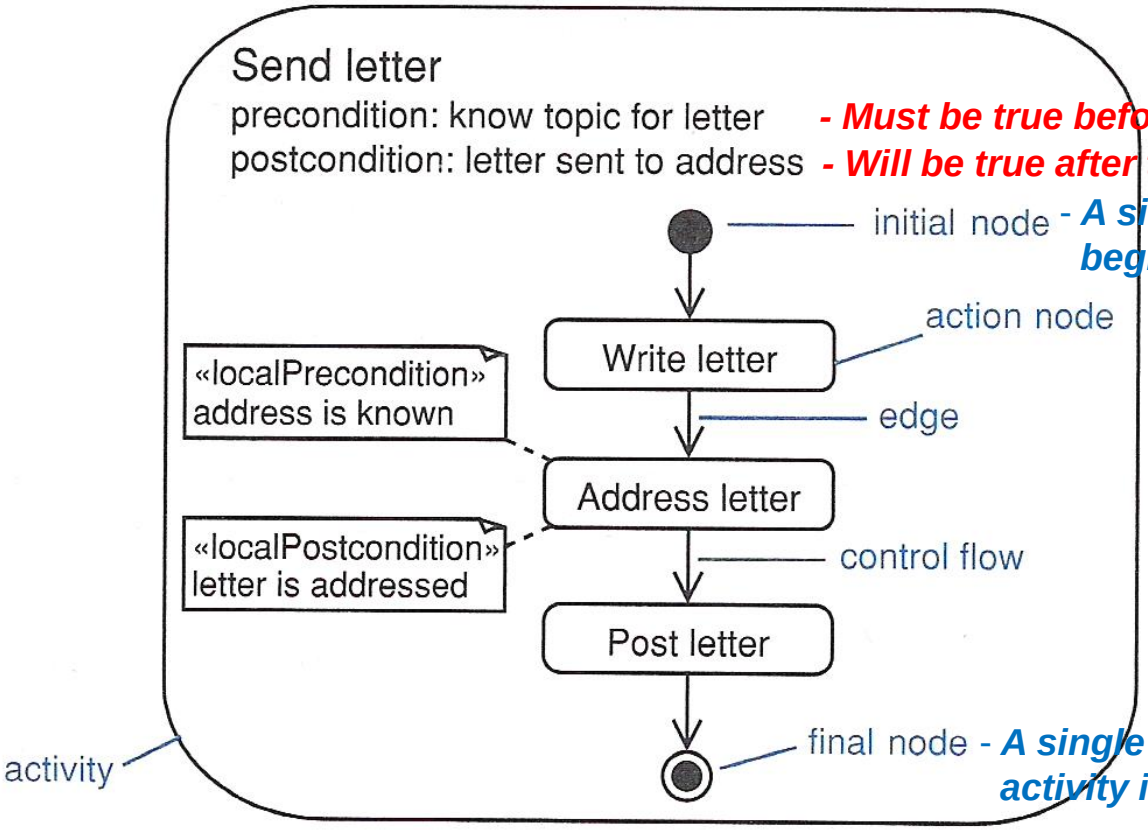
Activity Diagrams Symbols

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).



Examples of ADs

- A simple AD for a 'Send letter' business



- Must be true before the activity starts

- Will be true after the activity is finished

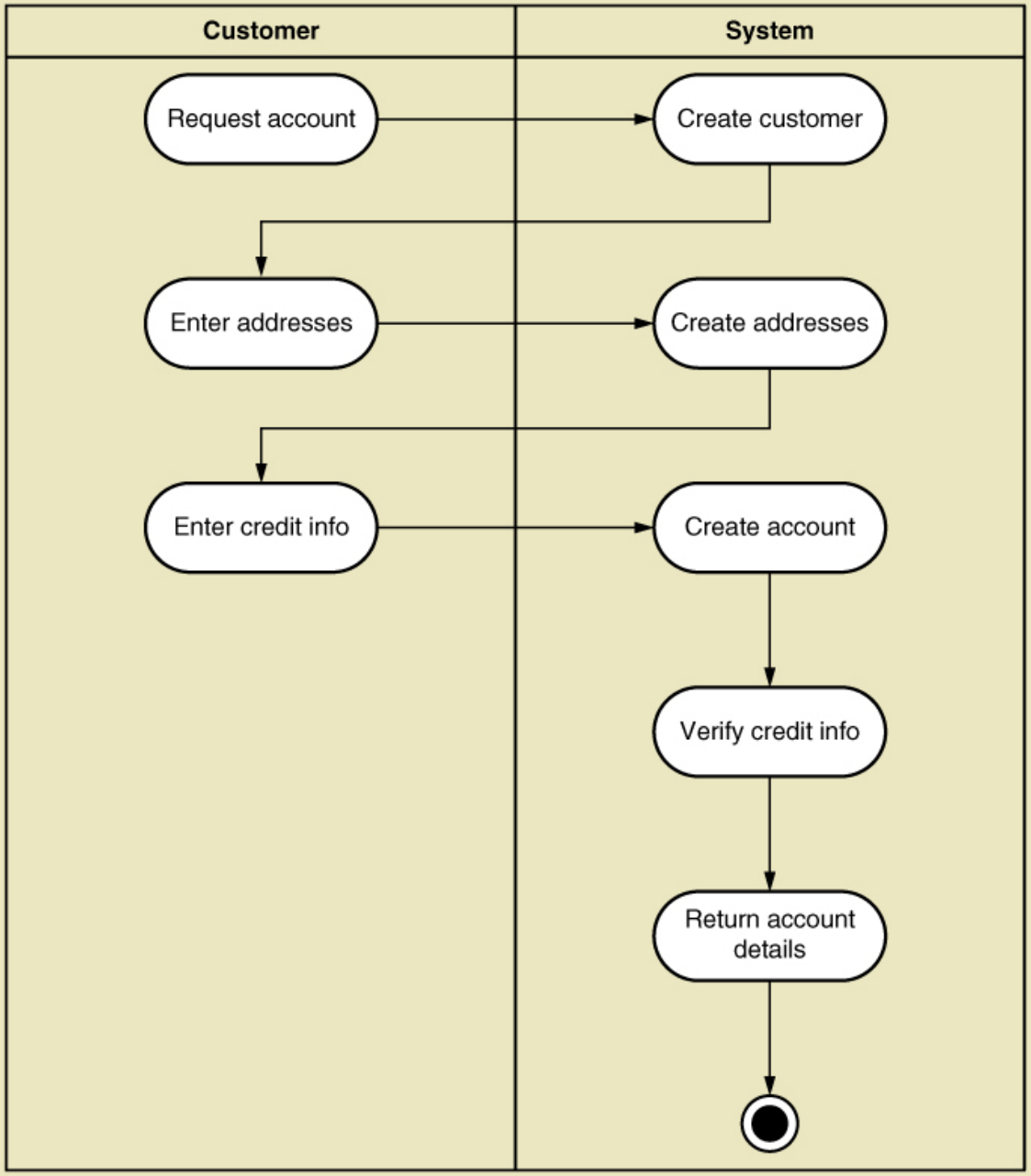
- A single control node where the execution begins when the activity is invoked

- A single or multiple nodes where the activity is terminated

UML Activity Diagram for Use Case

Create Customer Account

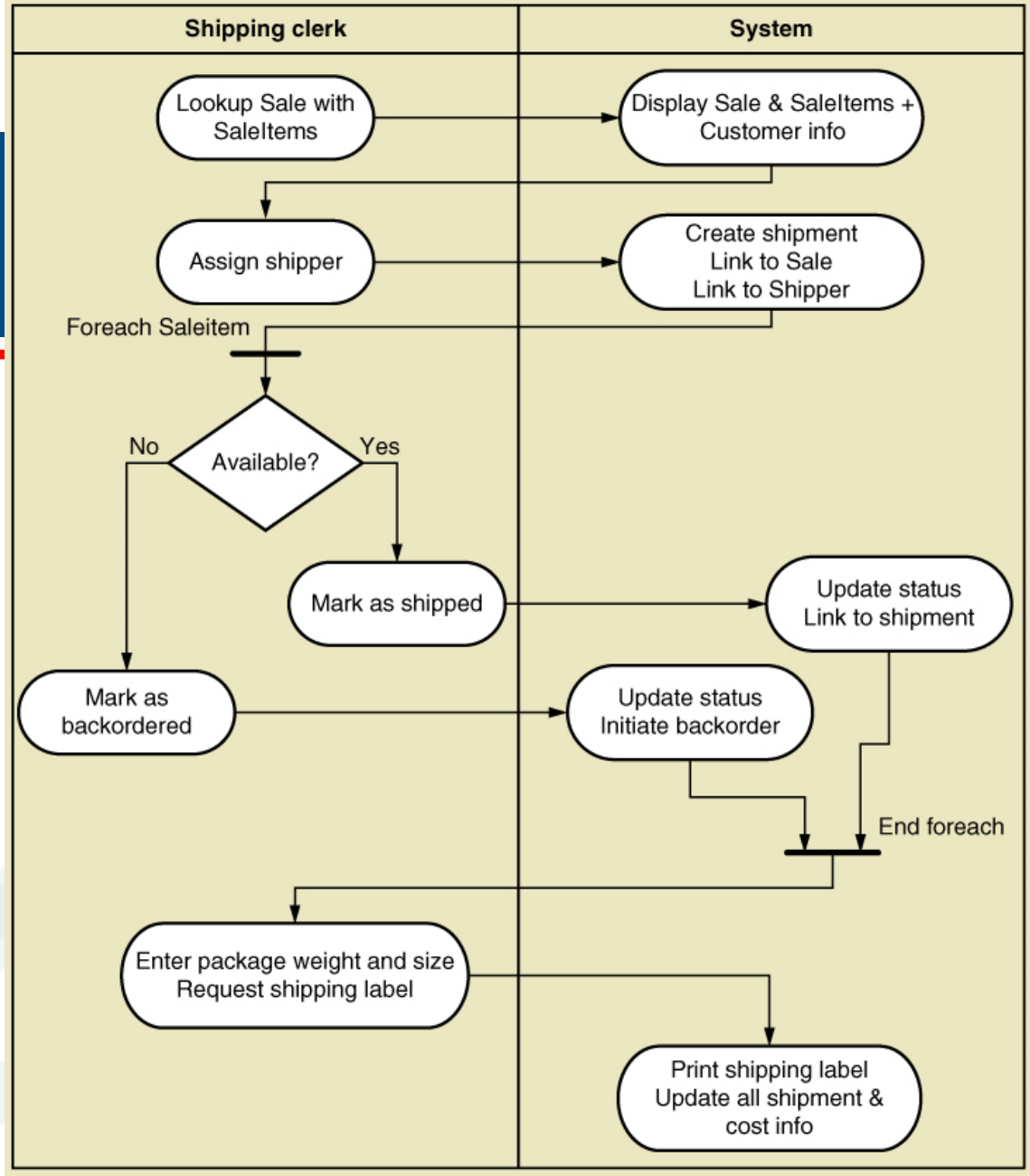
- Note: this shows flow of activities only



(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

Activity Diagram for *Ship Items* Use Case

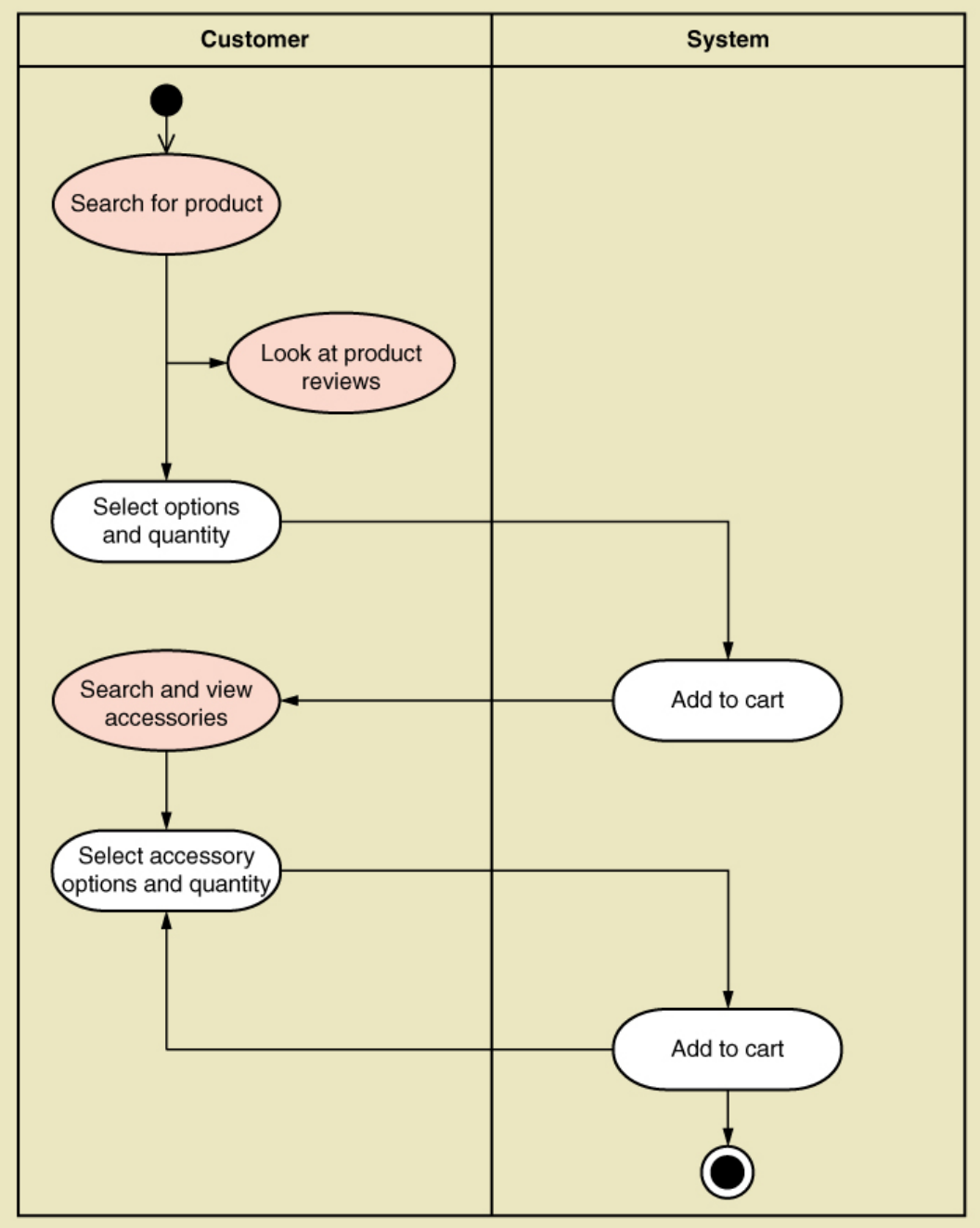
- Note:
  - Synchronization bar for loop
  - Decision diamond



UML Activity Diagram for Use Case

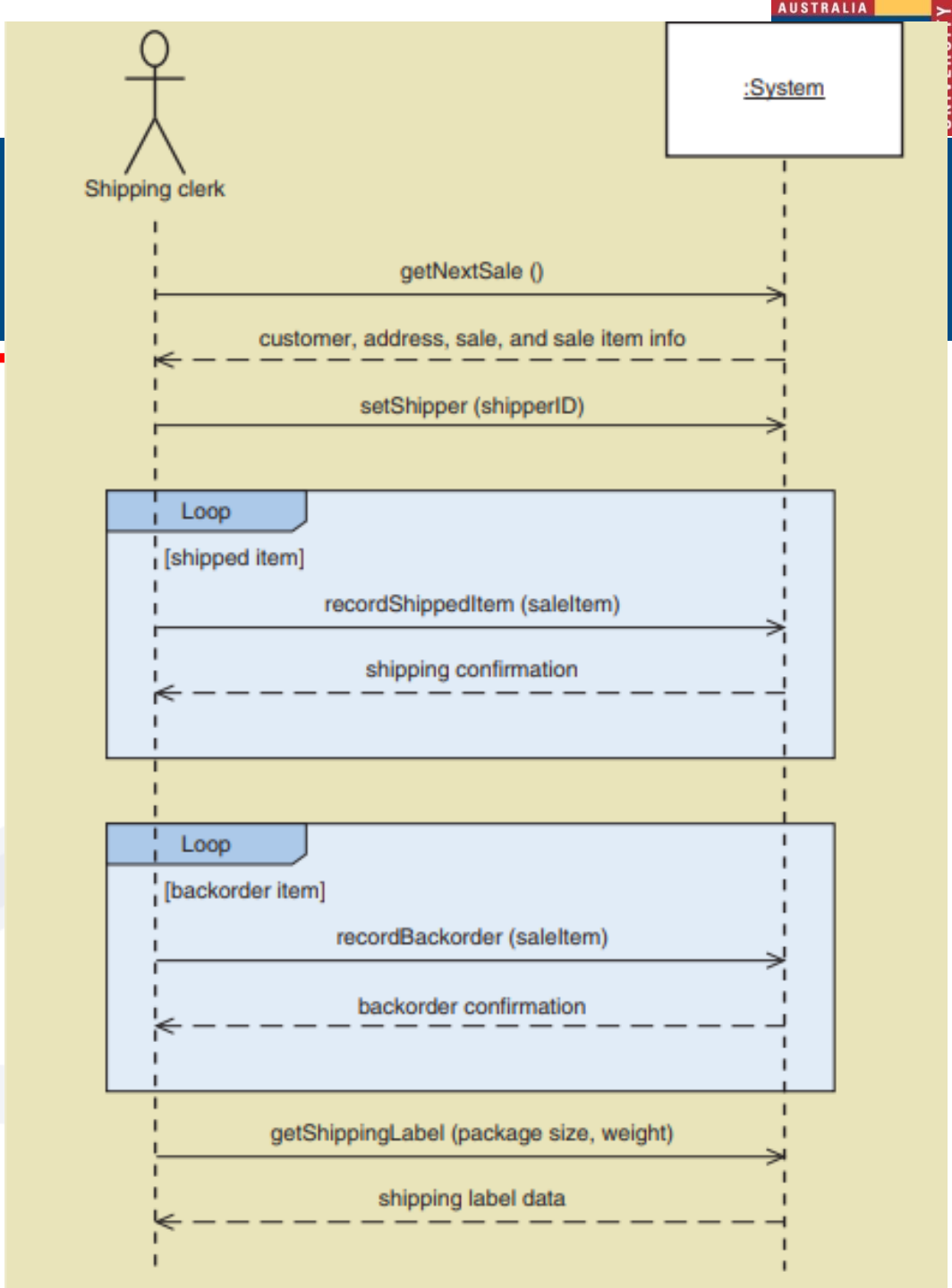
Fill shopping cart

- Note: this shows use case with <<includes>> relationship



(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning)

- SSD for the Ship items use case



CRUD

(For Validating Use Cases)

Use Cases and CRUD

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

- CRUD technique –
  - Create
  - Read/Report
  - Update
  - Delete
- A good cross-check against the existing set of use cases. Used in database context
  - Ensure that all entities in database have a complete "cover" of use cases
- Not for primary identification of use cases

Verifying use cases for Customer

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

Data entity/domain class	CRUD	Verified use case
Customer	Create	Create customer account
	Read/report	Look up customer Produce customer usage report
	Update	Process account adjustment Update customer account
	Delete	Update customer account (to archive)

Sample CRUD Matrix

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

Use case vs. entity/domain class	Customer	Account	Sale	Adjustment
Create customer account	C	C		
Look up customer	R	R		
Produce customer usage report	R	R	R	
Process account adjustment	R	U	R	C
Update customer account	UD (archive)	UD (archive)		

CRUD Analysis Steps

(Systems Analysis and Design in a Changing World, ©2016. Cengage Learning).

1. Identify all types of data (to be discussed details in next modules)
2. For each type of data verify that use cases exist to
   - Create a new instance
   - Update existing instances
   - Reads or reports on information in the class
   - Deletes or archives inactive instances
3. Add new use cases as required. Identify responsible stakeholders
4. Identify which application has responsibility for each action: which to create, which to update, which to use

Summary

- Different models to provide details of use cases are discussed.
- Fully *developed use case descriptions* provide information about each use case, including actors, stakeholders, preconditions, post conditions, the flow of activities and exceptions conditions.
- Some useful advice for creating use cases (I hope)
- Common pitfalls in writing use cases explained
- Guidance on what not to include in a use case

Summary

- The use case diagram is the UML diagram used to show the use cases and the actors
- The use case diagram shows the actors, the automation boundary, the use cases that involve each actor, the <<includes>> <<extends>> and Generalisation relationships.
- A variety of use case diagrams are drawn depending on the presentation needs of the analysis

Summary

- *Activity diagrams* can be used to show the flow of activities for a use case.
- *System sequence diagrams* (SSDs) show the inputs and outputs for each use case as messages
- *CRUD* analysis serves to verify that the new system have use cases to fully process all required actions
- Not all use cases are modelled at a detailed level. Only model when there is complexity and a need to communicate details.
- All of the models must be consistent and integrate together to provide a complete picture of the requirements and specification.

References

- Cockburn, A. (2000) Writing Effective Use Cases. Pre-publication draft.
- Constantine, L.L. (1997) The Case for Essential Use Cases. Object Magazine, May 1997, New York, NY: SIGS Publications.
- OMG (2007). OMG Unified Modeling Language Superstructure, V2.1.2. Needham, MA: Object Management Group, Inc.
- Quatrani, T. (2005) Writing Good Use Cases. Presentation: IBM Software Group
- Wirfs-Brock, R. and Schwartz, J. (2001) The Art of Writing Use Cases. Presentation: Wirfs-Brock and Associates.

Questions

?