Object-Oriented Analysis

Prof. T.H. Tse
Department of Computer Science
Email: thtse@cs.hku.hk
Web: hku.hk/thtse.

Object-Oriented Analysis

- Find and identify business classes
- Organize classes and identify their relationships

Use case modelling

Object modelling

Object modelling .

UML

- ***Unified Modelling Language*** (UML) is a set of modelling conventions used to specify a software system in terms of classes
  - Does ***not*** prescribe a ***method*** for developing systems
  - Only a ***notation*** widely accepted as a standard for object-oriented analysis and design .

UML Diagrams

Structural and behavioural

- Use case diagram
- Deployment diagram
  - Showing execution architecture

Structural

- Class diagram
- Component diagram
- Package diagram

Behavioural

- Interactions
  - Communication diagram
  - Sequence diagram
  - Timing diagram
- State machine
- Activity diagram .

Use Case Modelling

- The process of modelling system scenarios in terms of
  - business events
  - who initiates or participates in the events
  - how the system responds to the events .

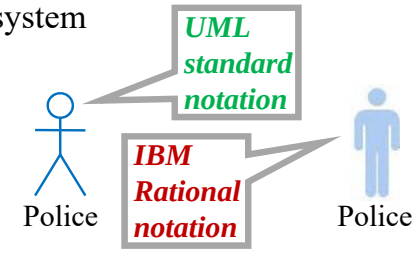
Scenarios and Use Cases

- A ***scenario*** is a behaviourally related ***sequence of events***, automated or manual, for the purpose of completing a business task
- A ***use case*** is a ***collection of related scenarios***, including normal and alternative scenarios

- ***Not*** directly related to object technology .

Actors

- An actor in a use case is an ***external*** party that uses or interacts with the system



- In general, for any use case, there is one ***initiating actor***, and possibly other ***participating actors*** .

Actors

- An actor can be a ***user*** or a ***role***, such as a person or an external system

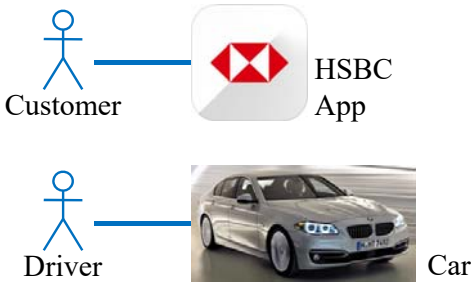
- It can also be a characteristic of the ***environment***, such as time or temperature change

Actors

Examples

Single-User Systems

- End user serves as actor



Actors

Examples

Traditional Multi-User Systems

- End user does not interact directly with system
- Support staff serves as actor

Actors

Examples

New Multi-User Systems

- New system will directly interact with end user
- Eliminates the need for existing support staff

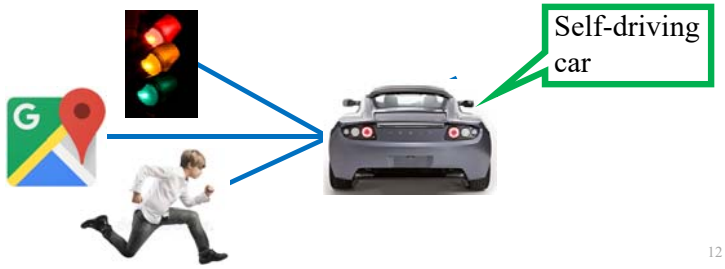
Actors

Examples

上下文？

Context-Sensitive Systems

- New system will directly sense the environment
- Eliminates the need for user interactions



Actors

Examples

"Temporal Event" Use Cases

- A ***temporal event*** is a system event that is triggered by time
  - The actor of a temporal event use case is ***time*** .

Use Case Example

User Requirement 1

- Put an elephant into the refrigerator

How many events?

How many scenarios?

How many use cases? .

Use Case Example

User Requirement 1

- Put an elephant into the refrigerator

User Requirement 2

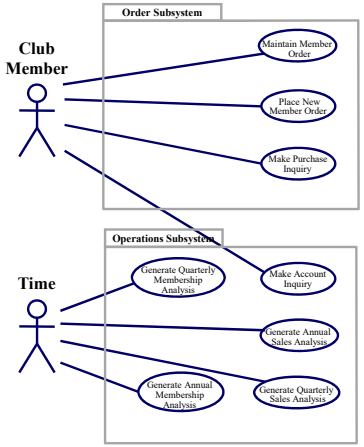
- Put a giraffe into the refrigerator

How many events altogether?

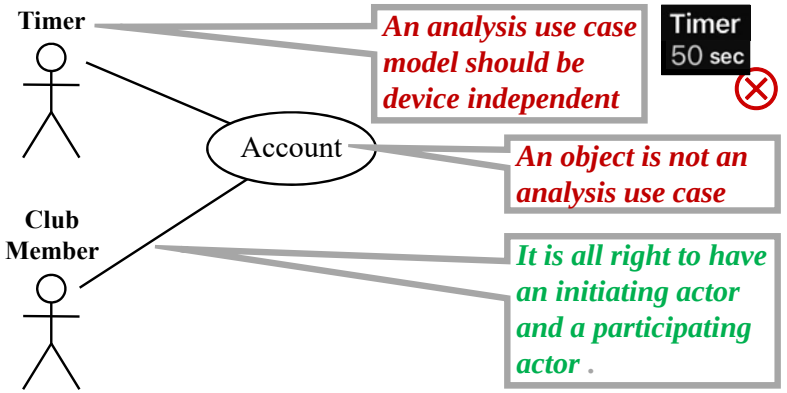
How many scenarios altogether?

How many use cases altogether? .

Use Case Diagrams



We Learn from Mistakes



Use Case Modelling

Benefits

- A basis to help identify classes and their high-level relationships and responsibilities
- A view of system behaviour from an external point of view
- An effective tool for validating requirements
- An effective communication tool
- A basis for test plan
- A basis for users' manual .

With users?

With designers?

With programmers?

Object Modelling

- ***Object modelling*** is a technique for identifying classes within the systems environment and the relationships among the classes
- Should be implementation independent
- Class diagrams rather than object diagrams (which are instance diagrams) .

Object Modelling

- ***Object modelling*** involves
  - A study of existing classes to see if they can be reused or adapted for new uses
  - Defining new or modified classes that will be combined with existing ones for a useful application .

Class Diagrams

- Class diagram graphically depicts the classes and their relationships
- What are the differences between class diagrams and entity-relationship diagrams?
  - Classes encapsulate _____
  - Relationships include _____

    _____ .

Examples of Classes

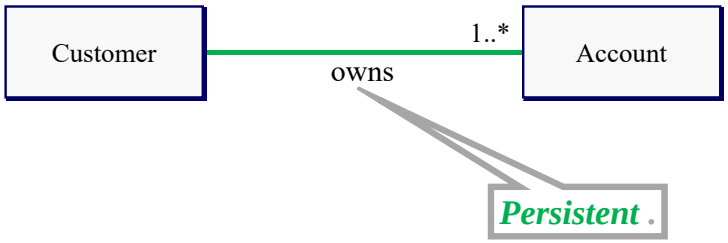
Customer
name age address
changeName changeAge changeAddress

Account
balance
debit credit .

Associations

- A link is a physical or conceptual relation between 2 or more objects
- An association describes a collection of links with common structure and common semantics
  - A link is an instance of an association
- Inherently bi-directional
  - *but* need not be implemented in both directions
- Also known as
  - ☹ relationships
  - ☹ client-server relationships
  - ☹ actor-server relationships
  - ☹ seniority relationships .

Example of Association

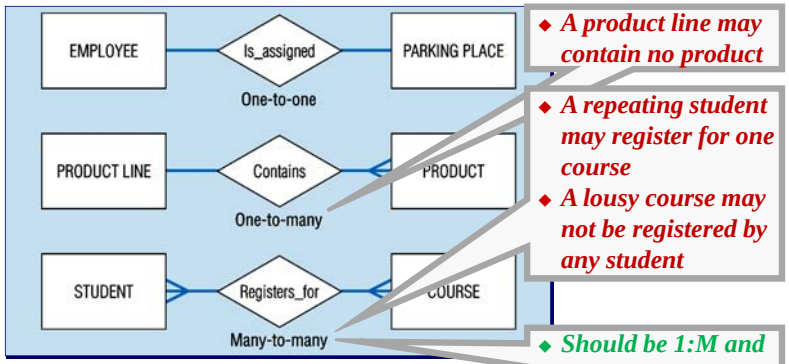


Multiplicity

- The multiplicity between classes *X* and *Y* refers to the number of instances of *X* that will be present for a given instance of *Y*, and vice versa
- Multiplicities are commonly referred to as 1:1, 1:*M*, *M*:1 and *M*:*M* associations
- Also known as
  - ☹ cardinality constraints
  - ☹ instance connections .

Multiplicity

We Learn from Mistakes



J.A. Hoffer, M.B. Prescott, and F.R. McFadden, *Modern Database Management*

- *A product line may contain no product*
- *A repeating student may register for one course*
- *A lousy course may not be registered by any student*
- *Should be 1:M and M:M, where M stands for multiple .*

Sample UML Multiplicity Notation

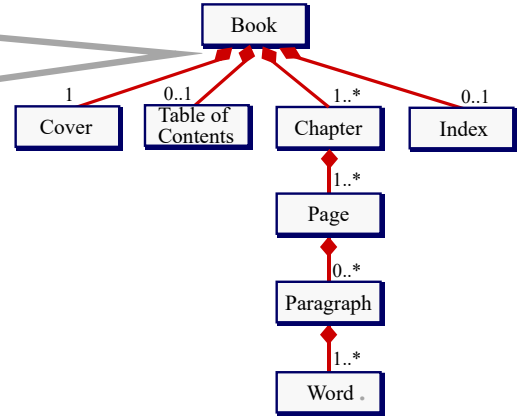
Multiplicity	*Notation*	*Example*
Exactly 1	**1** or ***blank***	Employee — works for — **1** Department Employee — works for — Department
Zero or one	**0..1**	Employee — has — **0..1** Spouse
Zero or more	**0..*** or *****	Customer — has settled — **0..*** Payment Customer — has settled — ***** Payment
One or more	**1..***	University — offers — **1..*** Course
Specific range	**7..9**	Team — has scheduled — **7..9** Game

Aggregations

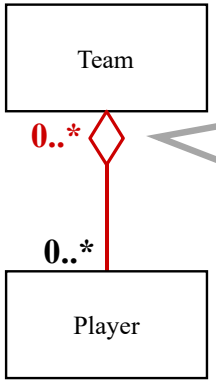
- Aggregation is a kind of association
- Aggregations are also called "*is-part-of* " relationships
- *Composition aggregation* (or simply *composition*) versus *shared aggregation* (or simply *aggregation*) .

Example of Composition Aggregation

Solid diamond indicates *composition aggregation*

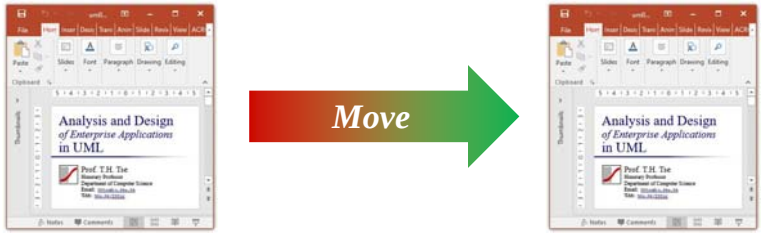


Example of Shared Aggregation



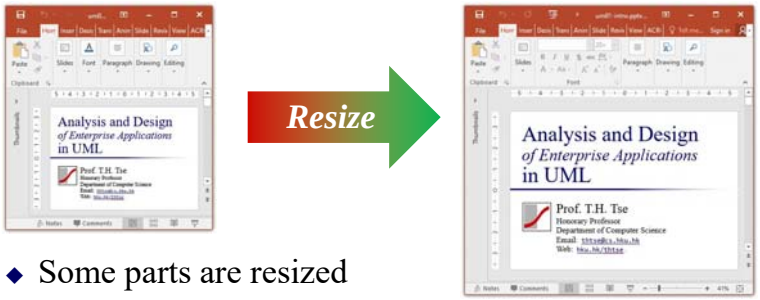
- Hollow diamond indicates *shared aggregation*
- A component may belong to any number of aggregates
  - For example, a player may belong to any number of teams .

Why Do We Need Aggregations?



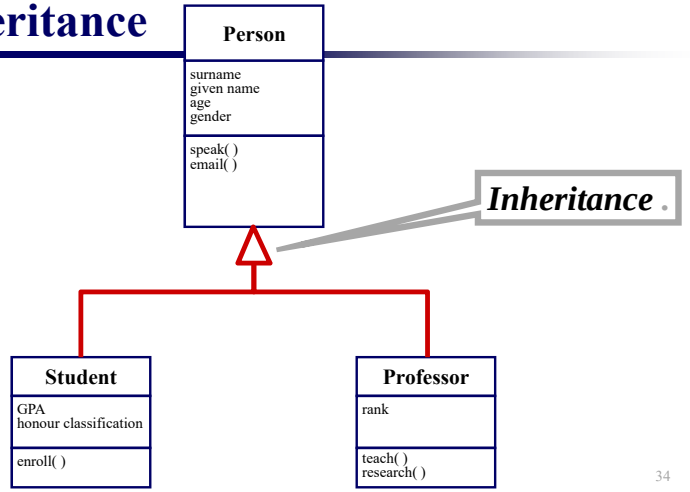
- All the parts move with the whole .

Why Do We Need Aggregations?



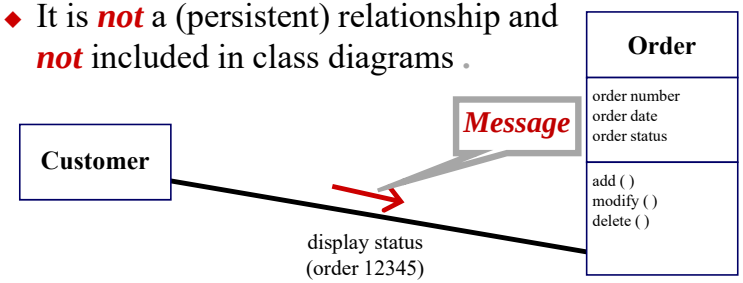
- Some parts are resized
- Others are not
- Need specific methods for each part .

Inheritance



Messages

- A ***message*** is passed when one object invokes one or more of another object's operations to request information or some action
- It is ***not*** a (persistent) relationship and ***not*** included in class diagrams .



Object-Oriented Analysis

- ***Find and identify business classes***
- Organize classes and identify their relationships .

Example

Member Services System

Find and Identify Business Classes

- Step 1. Identify actors and use cases
- Step 2. Construct a use case diagram
- Step 3. For each use case, document normal course of events
- Step 4. For each use case, document alternative courses of events
- Step 5. Identify any use case relationships
- Step 6. Find potential classes
- Step 7. Select proposed classes .

Step 1.

Identify Actors and Use Cases

- Analyse the context of the system
- If an external party initiates the input, it is an actor
- Some inputs are self-explanatory, but others may be misleading
- Confirm your findings with the business analyst .

Step 1.

Identify Actors and Use Cases

Use Case	Actor(s)	Description
Place Member Order	Member	Describes the process of a member submitting an order for SoundStage products .

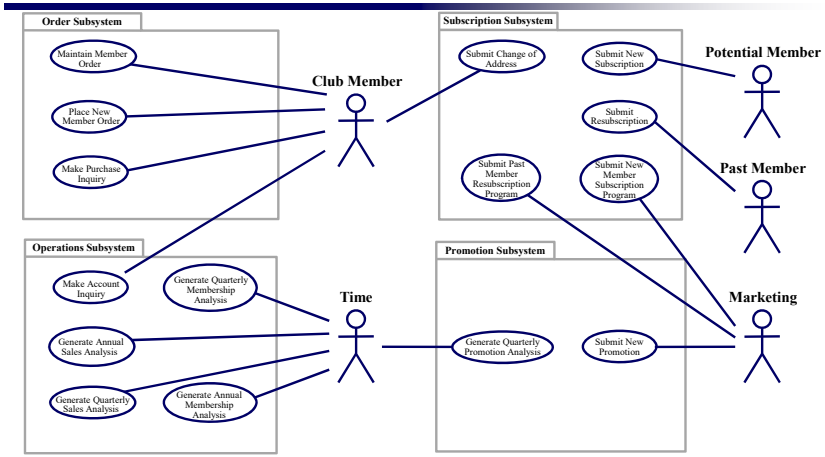
Step 2.

Construct Use Case Diagram

- A ***use case diagram*** graphically depicts the system scope and boundaries
- It represents the relationships between actors and use cases
- Partitioning system behaviour into subsystems is important in the ***development*** strategy: Which use cases will be developed first and by whom .

Step 2.

Construct Use Case Diagram



Step 3. For Each Use Case

Document Normal Course of Events

- The normal course of events is a step-by-step description starting with the actor initiating the use case until the end of the business events
- For ***each*** use case identified, document its normal course of events
- Express in structured English
- At this point, include only the major events occurring most of the time
  - Document exceptional conditions and events later .

Step 3. For Each Use Case

Document Normal Course of Events

Author: S. Shepherd Date: 01/07/2047

Use Case Name	Place Member Order	
Actor(s)	Member, Warehouse	
Description	Describes the process of a member submitting an order for SoundStage products.	
Reference	MSS-1.0	
Normal Course of Events	**Actor Action**	**System Response**
	Step 1. Initiate this use case when a member submits an order.	
		Step 2. Validate member's personal information such as address against what is currently on file.
		Step 3. Check member's credit status with accounts department system to ensure no outstanding payment.
		Step 4. For each product ordered, validate the product number, check availability in inventory, and record the ordered product information such as quantity ordered.
		Step 5. Calculate order subtotal and sales tax.
		Step 6. Verify member's credit card information based on the amount due.
		Step 7. Create packing slip for the member order containing all ordered products available and route it to warehouse.
		Step 8. Generate order confirmation notice indicating the status of the order and send it to the member.
	Step 9. Conclude this use case when the member receives the order confirmation notice .	

"information"

"such as"

Step 3. For Each Use Case

Document Normal Course of Events

Pre-Condition	Member has logged in.
Post-Condition	Member order has been recorded and packing slip has been routed to warehouse.
Assumption	None at this time .

Example of Pre-Condition

- Member has logged in

Compare with

- Step 1. Initiate this use case when a customer logs in and submit an order

The difference is not important because use cases only show scenarios, not implementation .

Example of Post-Condition

- Member order has been recorded

Useful check if there is more than one scenario .

Step 4. For Each Use Case

Document Alternative Courses of Events

- Alternative courses of events are deviations from the normal course
- A use case can have multiple alternative courses
- They are documented separately .

Step 4. For Each Use Case

Document Alternative Courses of Events

Alternative Courses	**Alt. Step 2A.** If member has indicated an address change in the order, update the member's record. **Alt. Step 3A.** If accounts department system returns a credit status that the member is in arrears, send order rejection notice to the member. **Alt. Step 4A.** If product number is invalid, send notification to member requesting them to resubmit product number. If the ordered product is not available, record the product information and mark as "back order". **Alt. Step 6.** If member's credit card information is invalid or if member is in arrears, send credit problem notice to the member. Modify the order's status to "on hold pending payment" .
Pre-Condition	Member has logged in.
Post-Condition	Member order has been recorded and packing slip has been routed to warehouse.
Assumption	None at this time.

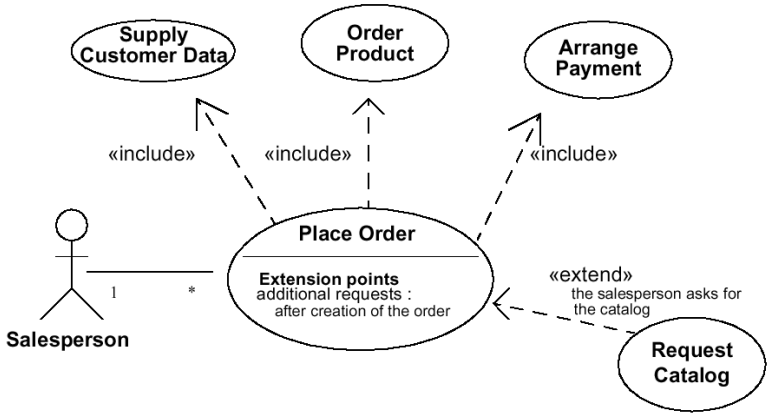
Step 5.

Identify Any Use Case Relationships

- Part of the use case diagram
- Only useful for ***excessively complex*** use cases
- Hence, ***more useful in design*** than in analysis .

Use Case Relationships

Example



Use Case Relationships

Association

- Instances of the actor and instances of the use case communicate with each other
- The only relationship between actors and use cases

Include

- An include relationship ***from*** use case ***A*** ***to*** use case ***B***

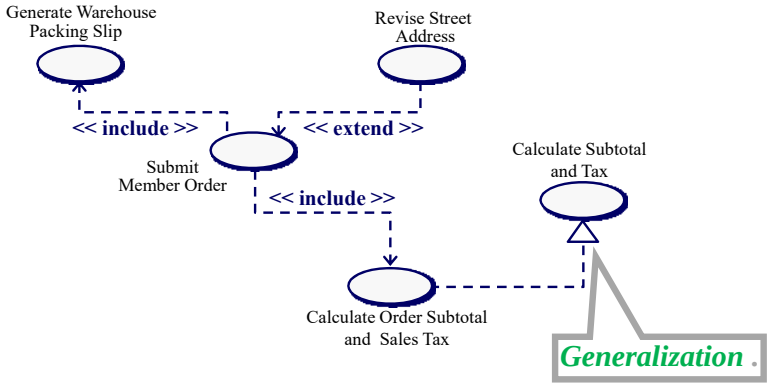
  indicates that an instance of *A* will also contain the behaviour as specified by *B* .

Use Case Relationships

Extend

- An extend relationship ***from*** use case ***B*** ***to*** use case ***A*** indicates that an instance of *A* may be augmented ***(subject to conditions specified in the extension)*** by the behaviour specified by *B* .

Use Case Relationships

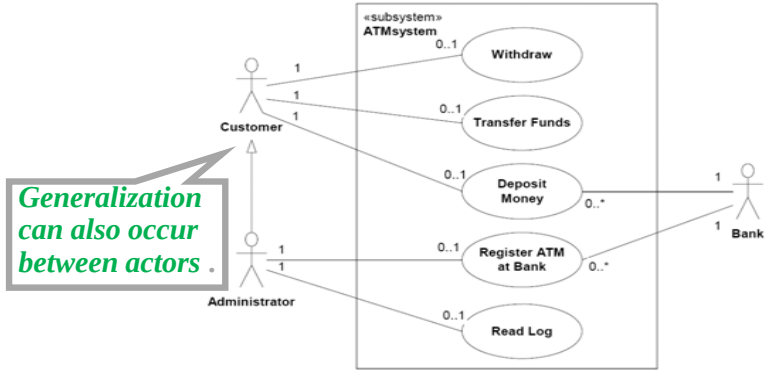


Use Case Relationships

Generalization

- A generalization ***from*** use case ***A*** ***to*** use case ***B*** indicates that *A* inherits *B* .

Use Case Relationships



Example

Include and Generalization

Author: S. Shepherd Date: 01/07/2047

Use Case Name	Place Member Order	
Actor(s)	Member, warehouse	
Description	Describes the process of a member submitting an order for SoundStage products.	
Reference	MSS-1.0	
Normal Course of Events	**Actor Action** **Step 1**. Initiate this use case when a member submits an order. **Step 9**. Conclude this use case when the member receives the order confirmation notice.	**System Response** **Step 2**. Validate member's personal information such as address against what is currently on file. **Step 3**. Check member's credit status with accounts department system to ensure no outstanding payment. **Step 4**. For each product ordered, validate the product number, check availability in inventory, and record the ordered product information such as quantity ordered. **Step 5.** **Invoke use case Calculate Order Subtotal and Sales Tax.** **Step 6**. Verify member's credit card information based on the amount due. **Step 7**. **Invoke use case Generate Warehouse Packing Slip.** **Step 8**. Generate order confirmation notice indicating the status of the order and send it to the member.

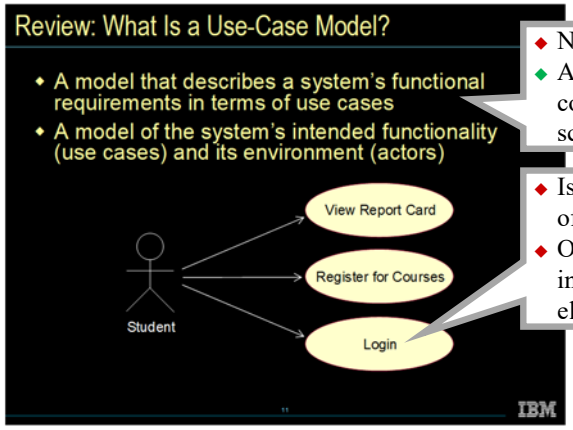
Example

Extend

Alternative Courses	**Alt. Step 2A.** **⟨Extension Point⟩ If member has indicated an address change in the order, invoke use case *Revise Street Address*.** **Alt. Step 3A.** If accounts department system returns a credit status that the member is in arrears, send order rejection notice to the member. **Alt. Step 4A.** If product number is invalid, send a notification to the member requesting them to resubmit the product number. If the ordered product is not available, record the product information and mark as "back order". **Alt. Step 6**. If member's credit card information is invalid or if member is in arrears, send credit problem notice to member. Modify the order's status to "on hold pending payment."
Pre-Condition	Member has logged in.
Post-Condition	Member order has been recorded and packing slip has been routed to warehouse.
Assumption	None at this time.

Use Case Modelling (Steps 1 to 5)

We Learn from Mistakes

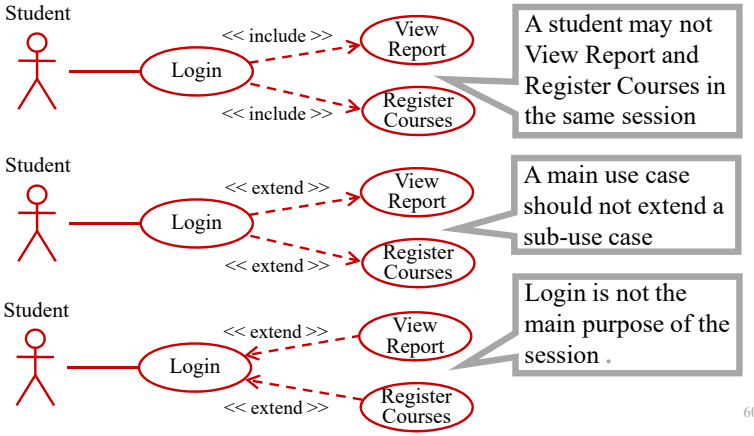


- No
- A use case is a collection of related scenarios

- Is login a collection of related scenarios?
- Only if students log in and do nothing else!

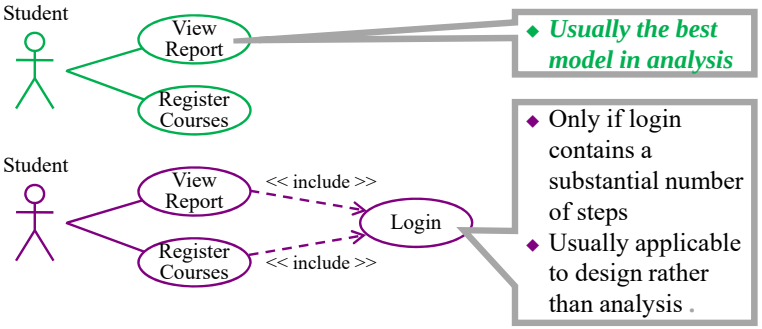
Use Case Modelling (Steps 1 to 5)

We Learn from Mistakes



Use Case Modelling (Steps 1 to 5)

Recommended Examples



- ***Usually the best model in analysis***
- Only if login contains a substantial number of steps
- Usually applicable to design rather than analysis .

Use Case Modelling (Steps 1 to 5)

Further Mistakes to Avoid

- (Don't) confuse use case diagrams with use case descriptions
- (Don't) confuse use cases with system functions
- (Don't) use a system perspective
- (Don't) avoid details in user requirements ...

Use case diagrams are not flowcharts

Use cases are collections of scenarios

Analysis use cases describe user requirements

Use Case Modelling (Steps 1 to 5)

Further Mistakes to Avoid

- (Don't) describe only user interactions and ignore system responses
- (Don't) forget alternative courses of events
- (Don't) write in passive voice

System responses may be part of the user requirements

Use present tense, active voice ...

Use Case Modelling (Steps 1 to 5)

Further Mistakes to Avoid

- (Don't) implement from use case descriptions
- (Don't) aim for functional decomposition using <<include>> or <<extend>>
- (Don't) aim at perfect use case descriptions

Use cases are not object-oriented

Use cases are not for implementation

Use cases are not for implementation .

Step 6.
Find Potential Classes

- Review each use case and highlight nouns that correspond to business entities

Recall Step 3. For Each Use Case
Document Normal Course of Events

Author: S. Shepherd — Date: 01/07/2047

Use Case Name	Place Member Order	
Actor(s)	Member, warehouse	
Description	Describes the process of a member submitting an order for SoundStage products.	
Reference	MSS-1.0	
Normal Course of Events	**Actor Action** **Step 1**. Initiate this use case when a member submits an order. **Step 9**. Conclude this use case when the member receives the order confirmation notice .	**System Response** **Step 2**. Validate member's personal information such as address against what is currently on file. **Step 3**. Check member's credit status with accounts department system to ensure no outstanding payment. **Step 4**. For each product ordered, validate the product number, check availability in inventory, and record the ordered product information such as quantity ordered. **Step 5.** Calculate order subtotal and sales tax. **Step 6**. Verify member's credit card information based on the amount due. **Step 7**. Create packing slip for the member order containing all ordered products available and route it to warehouse. **Step 8**. Generate order confirmation notice indicating the status of the order and send it to the member.

Step 6. Find Potential Classes
Highlight Nouns

Author: S. Shepherd — Date: 01/07/2047

Use Case Name	Place Member Order	
Actor(s)	Member, warehouse	
Description	Describes the process of a member submitting an order for SoundStage products.	
Reference	MSS-1.0	
Normal Course of Events	**Actor Action** **Step 1**. Initiate this use case when a member submits an order. **Step 9**. Conclude this use case when the member receives the order confirmation notice.	**System Response** **Step 2**. Validate member's personal information such as address against what is currently on file. **Step 3**. Check member's credit status with accounts department system to ensure no outstanding payment. **Step 4**. For each product ordered, validate the product number, check availability in inventory, and record the ordered product information such as quantity ordered. **Step 5.** Calculate order subtotal and sales tax. **Step 6**. Verify member's credit card information based on the amount due. **Step 7**. Create packing slip for the member order containing all ordered products available and route it to warehouse. **Step 8**. Generate order confirmation notice indicating the status of the order and send it to the member.

Recall Step 4. For Each Use Case
Document Alternative Courses of Events

Alternative Courses	**Alt. Step 2A.** If member has indicated an address change in the order, update the member's record. **Alt. Step 3A.** If accounts department system returns a credit status that the member is in arrears, send order rejection notice to the member. **Alt. Step 4A.** If product number is invalid, send notification to member requesting them to resubmit product number. If the ordered product is not available, record the product information and mark as "back order". **Alt. Step 6.** If member's credit card information is invalid or if member is in arrears, send credit problem notice to member. Modify the order's status to "on hold pending payment".
Pre-Condition	Member has logged in.
Post-Condition	Member order has been recorded and packing slip has been routed to warehouse.
Assumption	None at this time .

Step 6. Find Potential Classes
Highlight Nouns

Alternative Courses	**Alt. Step 2A.** If member has indicated an address change in the order, update the member's record. **Alt. Step 3A.** If accounts department system returns a credit status that the member is in arrears, send order rejection notice to the member. **Alt. Step 4A.** If product number is invalid, send notification to member requesting them to resubmit product number. If the ordered product is not available, record the product information and mark as "back order". **Alt. Step 6.** If member's credit card information is invalid or if member is in arrears, send credit problem notice to the member. Modify the order's status to "on hold pending payment".
Pre-Condition	Member has logged in.
Post-Condition	Member order has been recorded and packing slip has been routed to warehouse.
Assumption	None at this time.

Find Potential Classes

Potential Class		
Accounts Department System		
Address		
Address Change		
Amount Due		
Availability		
Credit Problem Notice		
File		
Inventory		
Member		
Member Order		
Member's Credit Card Information		
Member's Credit Status		
Member's Personal Information		
Member's Record		
Notification		
Order		
Order Confirmation Notice		
Order Rejection Notice		
Order Subtotal		
Ordered Product Information		
Outstanding Payment		
Packing Slip		
Product		
Product Number		
Product Ordered		
Quantity Ordered		
Sales Tax		
Status of Order		
Warehouse .		

Step 7.
Select Proposed Classes

- Not all the of the nouns represent good business classes
- Remove the nouns that represent:
  - Synonyms
  - Nouns outside the scope of the system
  - Nouns that are roles without unique behaviour or are external roles
  - Unclear nouns that need focus
  - Nouns that are really actions or attributes .

Select Proposed Classes

Proposed class .

Potential Class		Reason
Accounts Department System	×	Actor
Address	×	Attribute of Member
Address Change	×	Attribute of Member
Amount Due	×	Attribute of Member Order
Availability	×	Vague attribute of Product
Credit Problem Notice	×	Potential interface to be considered in design
File	×	Vague alias of Member
Inventory	×	Vague alias of Product
Member	✓	
Member Order	✓	
Member's Credit Card Information	×	Vague attribute of Member
Member's Credit Status	×	Attribute of Member
Member's Personal Information	×	Vague attribute of Member
Member's Record	×	Alias of Member
Notification	×	Potential interface to be considered in design
Order	×	Alias of Member Order
Order Confirmation Notice	×	Attribute of Member Order
Order Rejection Notice	×	Potential interface to be considered in design
Order Subtotal	×	Attribute of Member Order
Ordered Product Information	×	Vague attribute of Product
Outstanding Payment	×	Attribute of Member
Packing Slip	×	Potential interface to be considered in design
Product	✓	
Product Number	×	Attribute of Product and Product Ordered
Product Ordered	✓	
Quantity Ordered	×	Attribute of Product Ordered
Sales Tax	×	Attribute of Member Order
Status of Order	×	Attribute of Member Order
Warehouse	×	Actor .

Actor

Attribute

Interface

Synonym

Select Proposed Classes

Potential Class
Member
Member Order
Product
Product Ordered

Select Proposed Classes

- Some classes have been "pre-cooked" ***before the lecture*** to save time
- They will be useful later .

Potential Class
Member
Member Order
Product
Product Ordered
Plus
Agreement
Audio Title
Club Member
Game Title
Merchandise
Past Member
Potential Member
Promotion
Return
Title
Transaction
Video Title

Object-Oriented Analysis

- Find and identify the business classes
- ***Organize classes and identify their relationships*** .

Organize Classes and Identify their Relationships

- A class diagram is used to graphically depict classes and their relationships
  - Relationships include associations (multiplicities), inheritance, and aggregations .

Organize Classes and Identify their Relationships

- Step 1. Identify associations and multiplicities
- Step 2. Identify inheritance
- Step 3. Identify aggregations
- Step 4. Prepare a class diagram

> - ***Iterations of these 4 steps***
> - ***Not necessarily in sequence*** .

Organize Classes and Identify their Relationships

Step 1. Identify Associations and Multiplicities

- Recall that a relationship between 2 objects/classes is what one object/class "needs to know" about the other
- Once an association has been identified, the multiplicities should also be defined .

Organize Classes and Identify their Relationships

Step 2. Identify Inheritance

- ☺ Consider situations where "class *X* ***is a kind of*** class *Y* "
- ☹ Consider common attributes and behaviour across classes ***??***
- ☹ Consider reuse of program code ***??***

> ***Do not tackle design issues during analysis*** .

Organize Classes and Identify their Relationships

Step 3. Identify Aggregations

- ☺ Consider situations where "class X ***is part of*** class Y"
- ☹ Aggregation ***does not imply*** inheritance
  - ☹ Part of an object does not inherit attributes or behaviour from the whole
- ☺ Aggregation ***propagates*** behavior
  - ☺ ***Selected*** behaviour applied to the whole is applied to the part .

Aggregations Propagate Behaviour

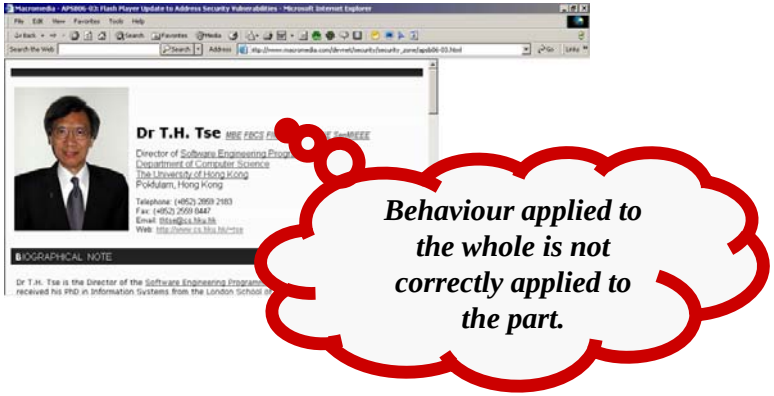
Recall

Aggregations Propagate Behaviour

Example of Failure

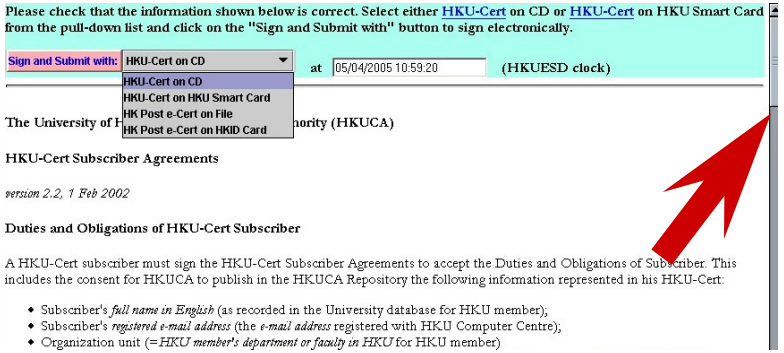
Aggregations Propagate Behaviour

Example of Failure



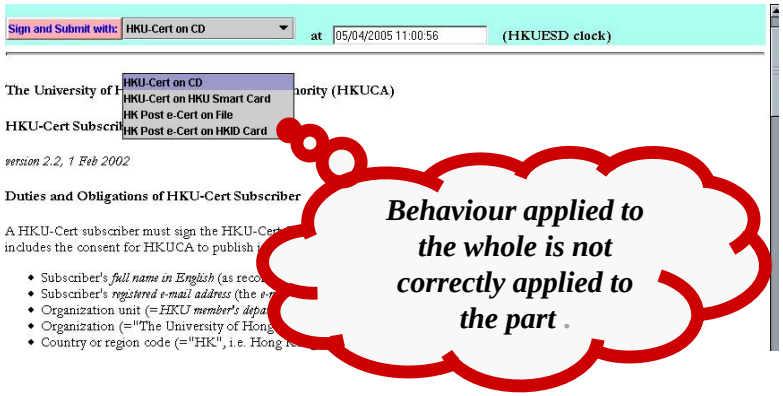
Aggregations Propagate Behaviour

Example of Failure



Aggregations Propagate Behaviour

Example of Failure



Organize Classes and Identify their Relationships

Step 4. Prepare a class diagram

- Construct a class diagram with class names, attributes, and relationships

Methods ? .

Member Services System

Class Diagram