ELSEWEDY UNIVERSITY OF TECHNOLOGY
- POLYTECHNIC OF EGYPT -

Software Engineering

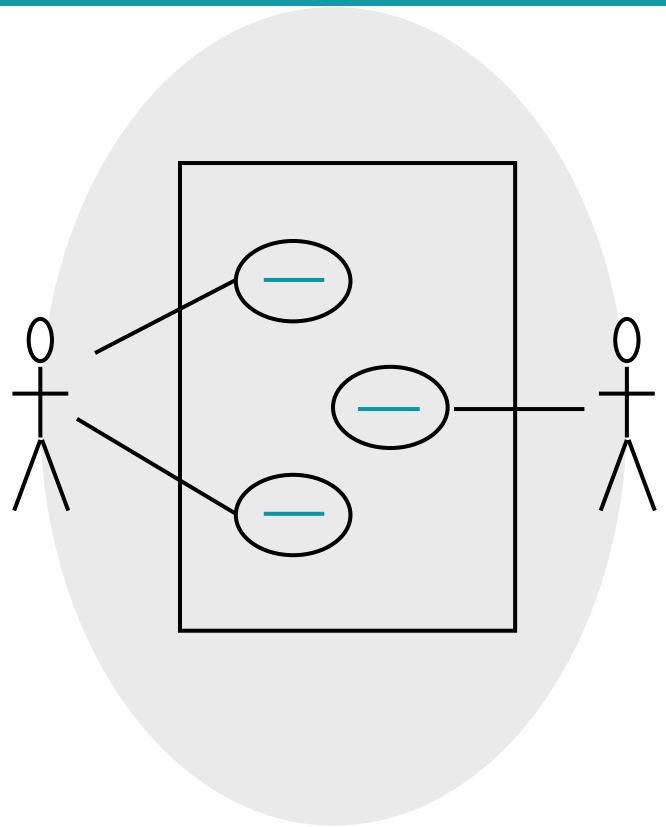
Lecture 3: UML

UNIFIED MODELING LANGUAGE™

© Fall 2024 – Dr. Alyaa Hamza
Alyaa.hamza@sut.edu.eg

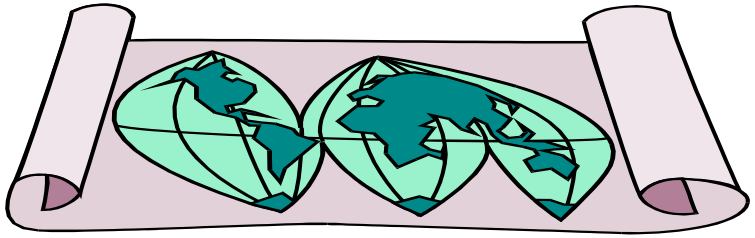
Objectives

- Understand where UML has come from
- Introduce the key UML Diagram Types
- Understand what a Use Case is
- Know how to model Use Case diagrams
- Know how to write a Use Case Description
- Recognise an Object Sequence Diagram
- Know what Robustness Analysis is



Why Build Models?

- Cheaper than building the real thing
- To provide a clear abstraction of concepts
- To gain understanding, before building
- To provide a common vision for diverse groups (users and developers)
- To manage, scope and document complexity



Analysis and Design

- The same ‘language’ is used to express Analysis and Design
- Avoids a “semantic gap”

Analysis

- **The ‘what’**
- **Defining the problem space**
- **A conceptual model to facilitate understanding**
- **Unaffected by software considerations**
- **Unaffected by performance considerations**

Design

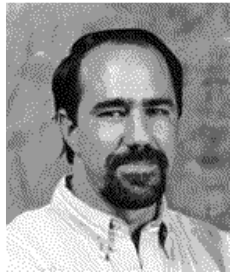
- **The ‘how’**
- **Defining the solution space**
- **A specification model to facilitate implementation**
- **Improving the software**
- **Improving the performance**

UML Major Contributors

JAMES RUMBAUGH

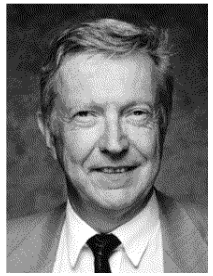
"Object-Oriented Modelling and Design" (1991)

GRADY BOOCH



"Object-Oriented Design with Applications" (1993)

IVAR JACOBSON



"Object-Oriented Software Engineering: A Use Case Driven Approach" (1992)

Why Is UML Important?

- Represents a convergence of Ideas
- An end to the “Methods Debate”?
- A modelling standardization
  - Semantics
  - Definition
  - Notation

Why Is UML Important?

- Use graphical notation: more clearly than natural language (imprecise) and code (too detailed).
- Help acquire an overall view of a system.
- UML is not dependent on any one language or technology.
- UML moves us from fragmentation to standardization.

تجزئة

توحيد

About UML

UML Specification
The Object Management Group (OMG) Version 2.2, February 2009

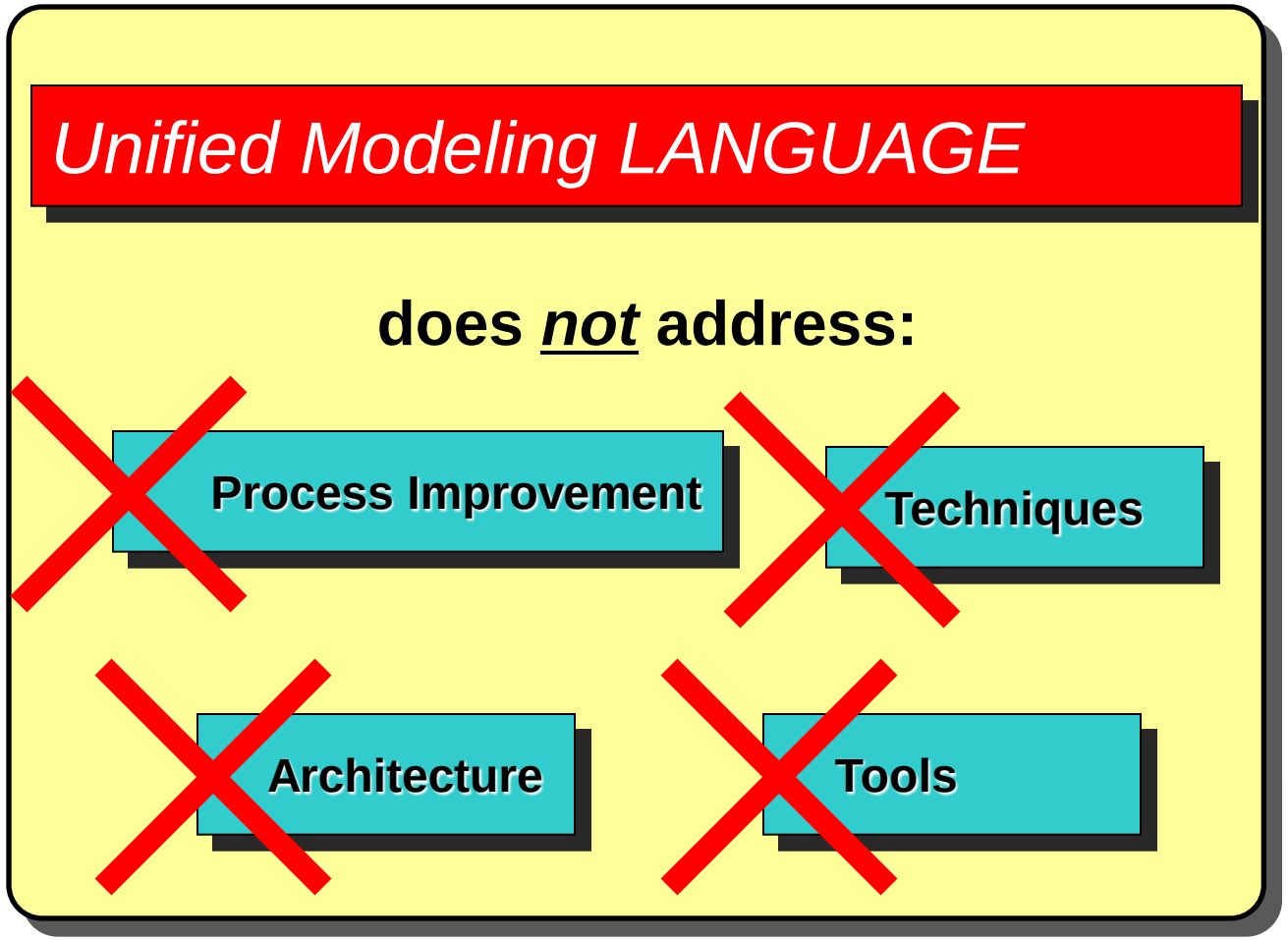
The Unified Modelling Language is a visual language for:
- specifying,
- constructing
- documenting

the artefacts of systems.

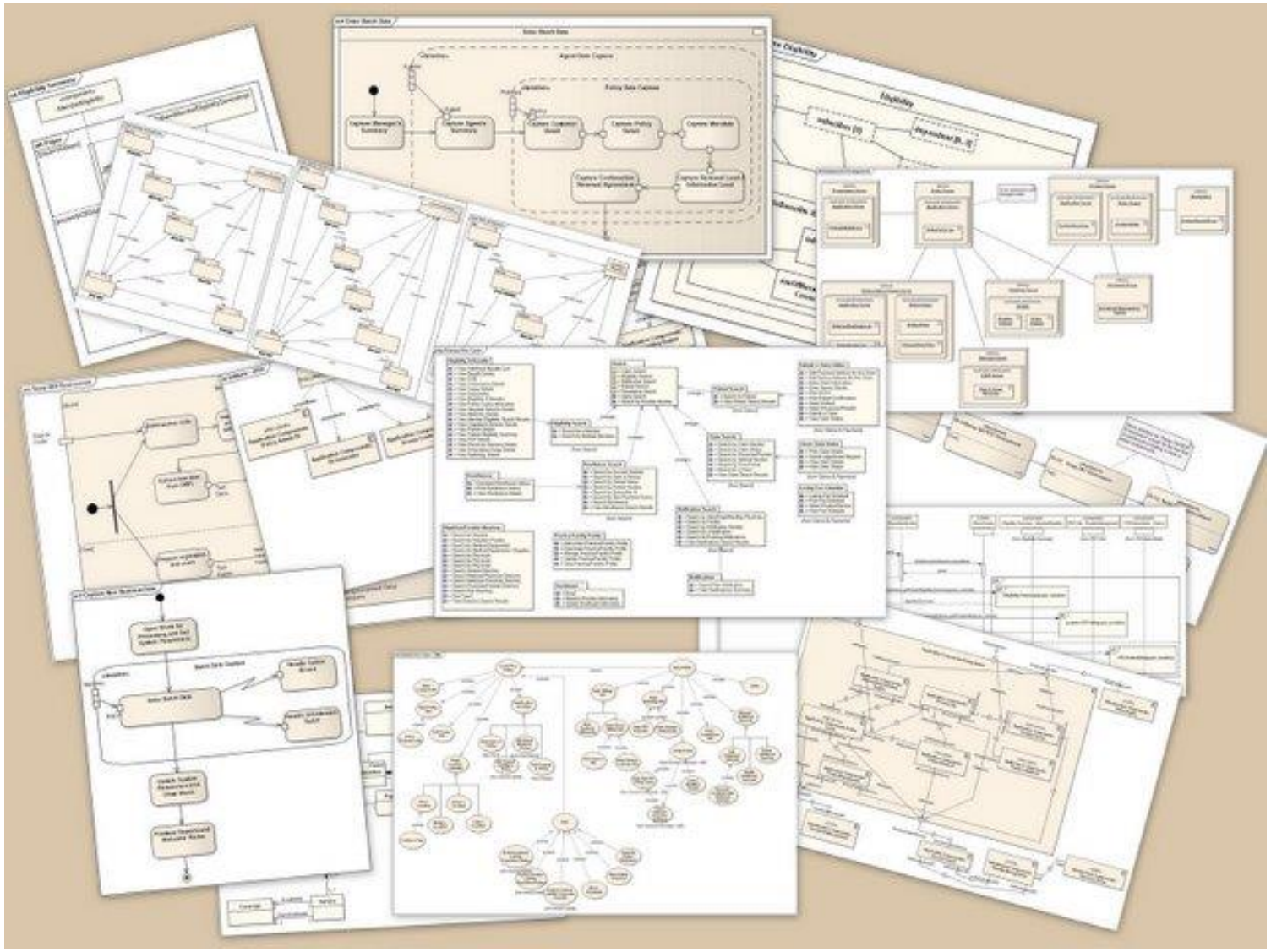
توحيد
بناء
توثيق

Can be applied to all application domains and implementation platforms

UML is not a Methodology!



UML – the Diagrams



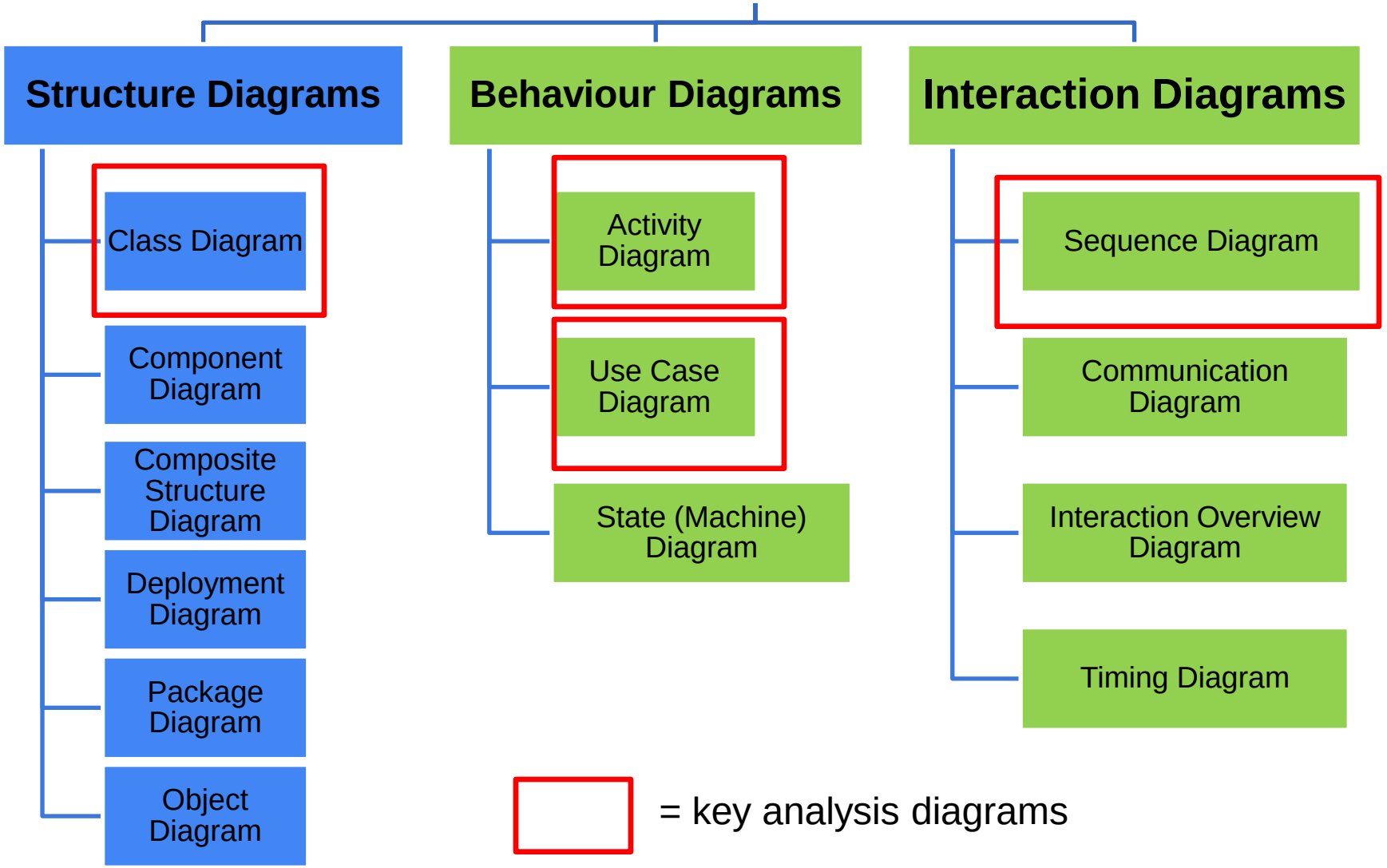
UML – the Diagrams

important

UML Diagrams show ...

- **Activities**
- **Actors**
- **Business processes**
- **Database schemas**
- **Logical components**
- **Physical components**
- **Programming language and software components**

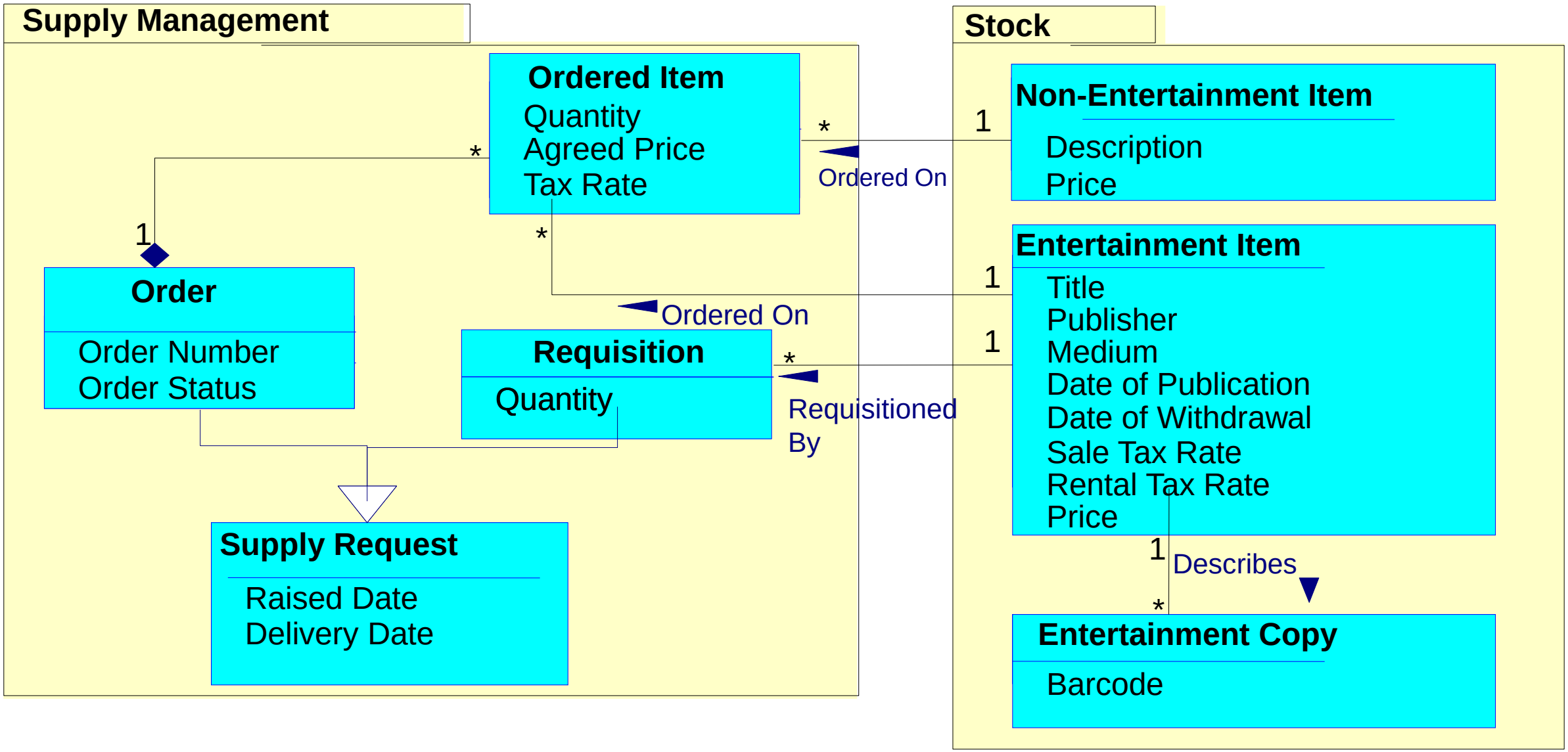
Key UML Diagrams



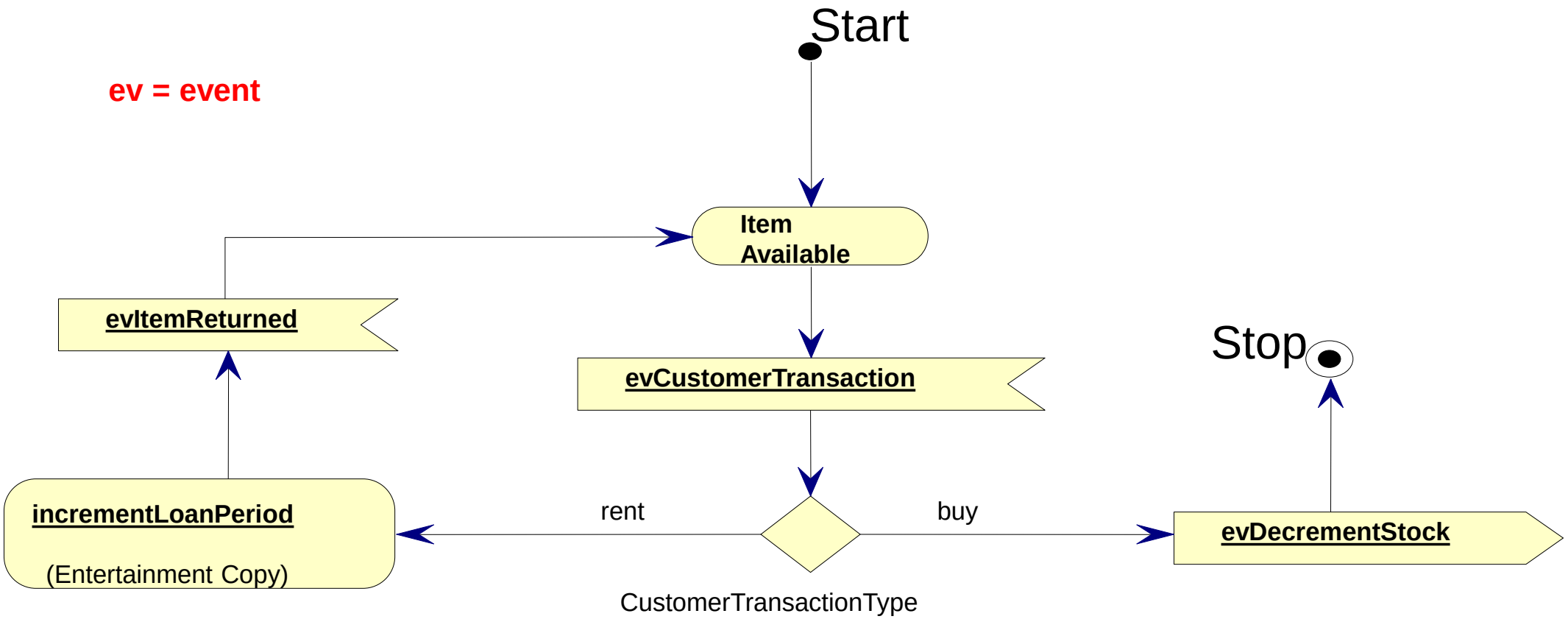
UML – the key diagrams

- **Structure Diagrams**
  - Class Diagram: shows the logical Classes, associations, attributes
  - Component Diagram: shows how software is split into components
  - Composite Structure Diagram: describes the internal structure of a Class
  - Deployment Diagram: describes the hardware used, and what is deployed on the hardware
  - Package Diagram: shows the elements of the system as logical groupings
  - Object Diagram: shows how objects (instances) are interrelated
- **Behaviour Diagrams**
  - Activity Diagram: describes step by step workflow (swimlanes)
  - Use Case Diagram shows actors, goals and processes
  - State (machine) Diagram describes the states and state transitions of the system
- **Interaction Diagrams**
  - Sequence Diagram: shows how objects communicate in terms of a sequence of messages
  - Communication Diagram: interaction between objects in the system, as messages
  - Interaction Overview Diagram: shows interactions between several communication diagrams
  - Timing Diagram: an Interaction diagram focusing on timing constraints

UML Class Diagram

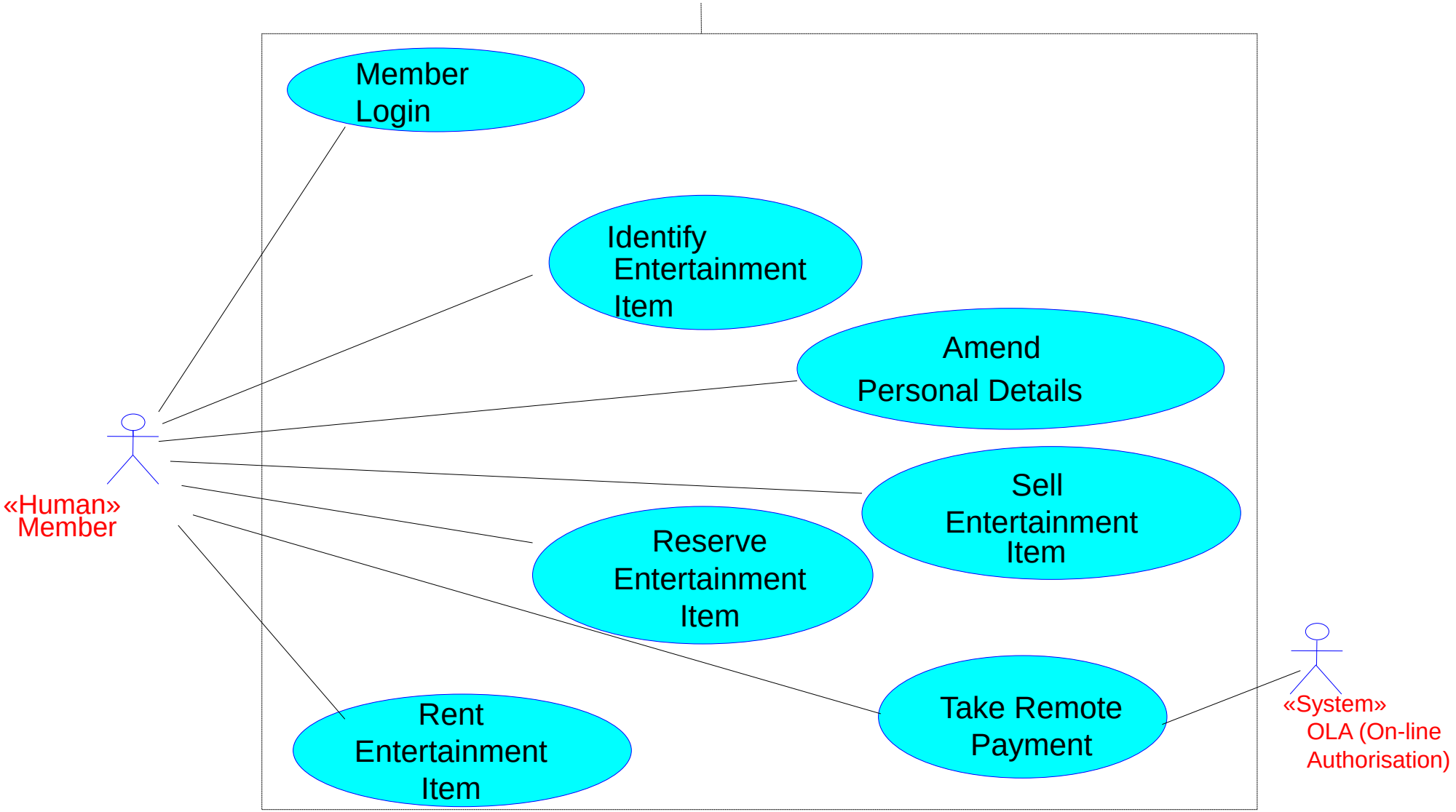


UML Activity Diagram



This diagram may be used to describe business processes (using swim lanes), individual Use Cases or the flow through an individual Class operation.

UML Use Case


Member Login
Identify Entertainment Item
Amend Personal Details
Sell Entertainment Item
Reserve Entertainment Item
Rent Entertainment Item
Take Remote Payment
«Human» Member
«System» OLA (On-line Authorisation)

UML Sequence Diagram

Rent Entertainment Copy to Member

- If Transaction is not started
  - Start Transaction
- End If
- Perform certification check

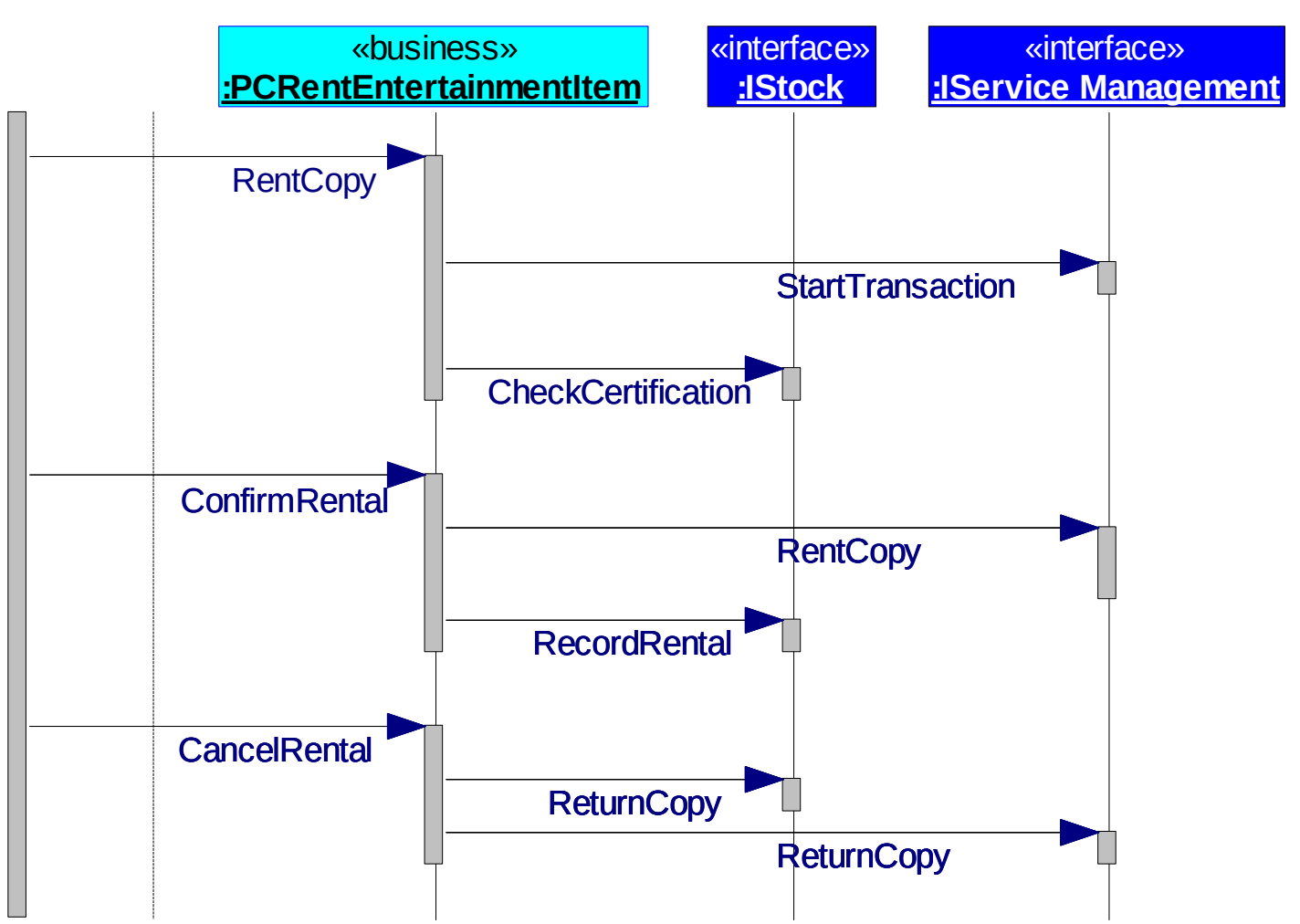
If Member is permitted to rent

- Confirm the Rental of the copy
  - Rent the item for the agreed period
  - Remove item from stock

Else

- Cancel the Rental of the copy
  - Re-stock the item
  - Remove item from transaction

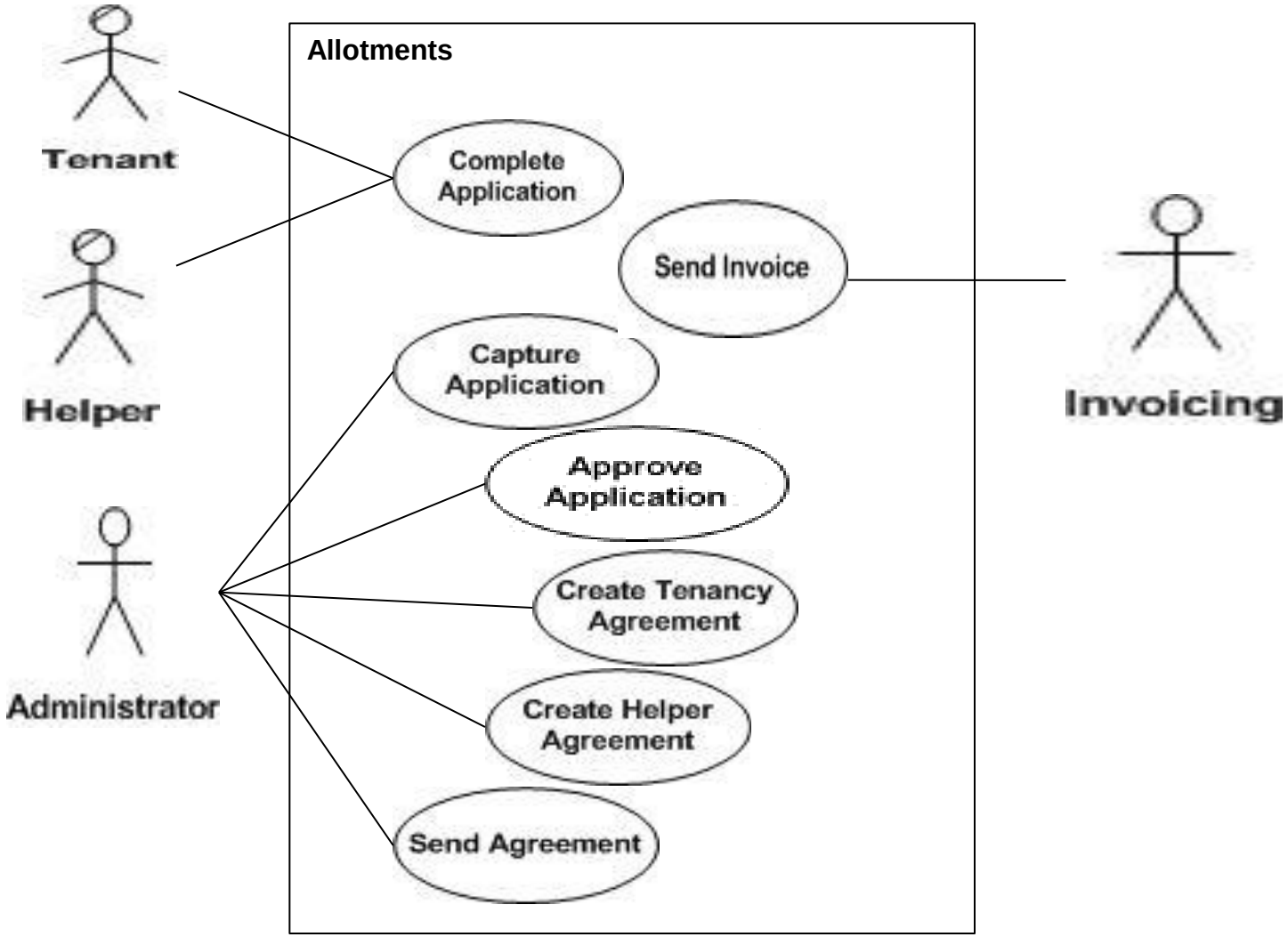
End If



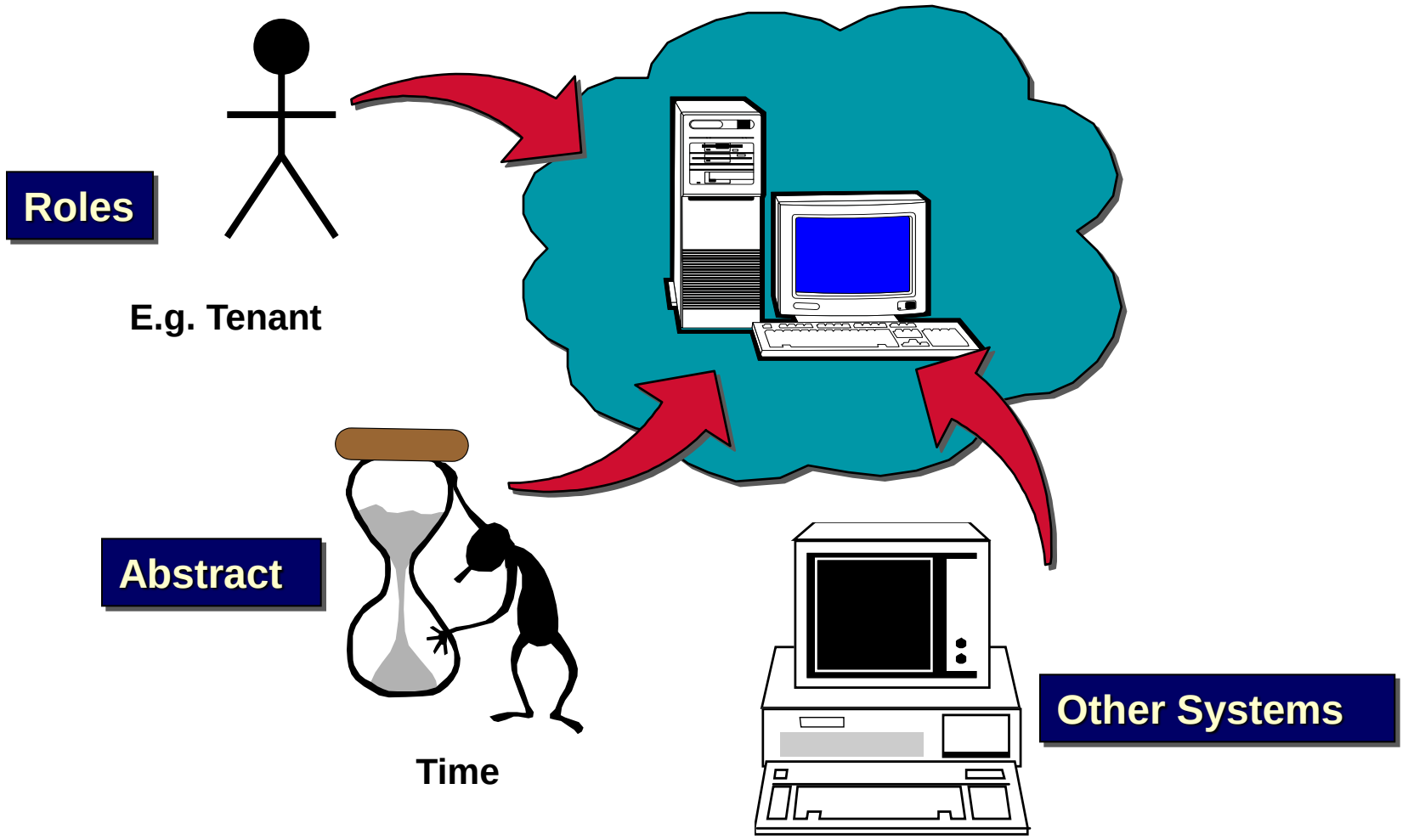
Use Cases

- **A natural unit of work**
- **Each “bubble” typically one person, one place, one time**
- **Overview diagram of many Use Cases give Scope**
- **Incorporate requirements**
- **Useful for testing too**

UML Use Case Diagram

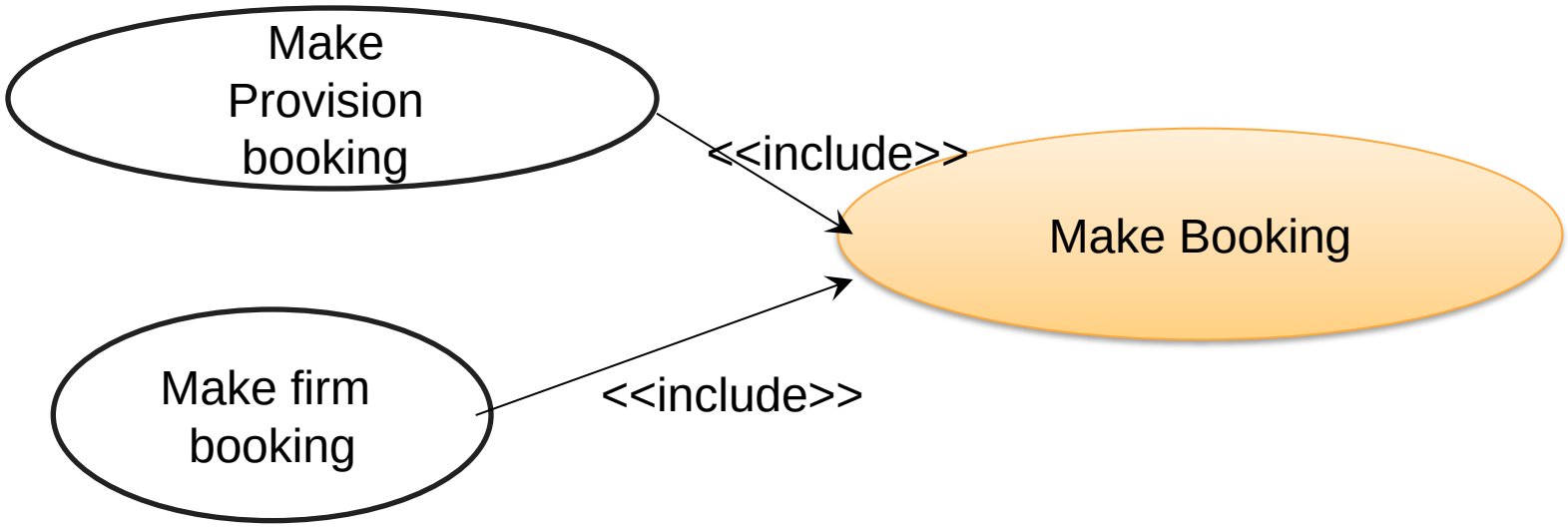


What is an Actor?



Relationship between use cases

- Use cases can hold some relationships between each other, like include, extend



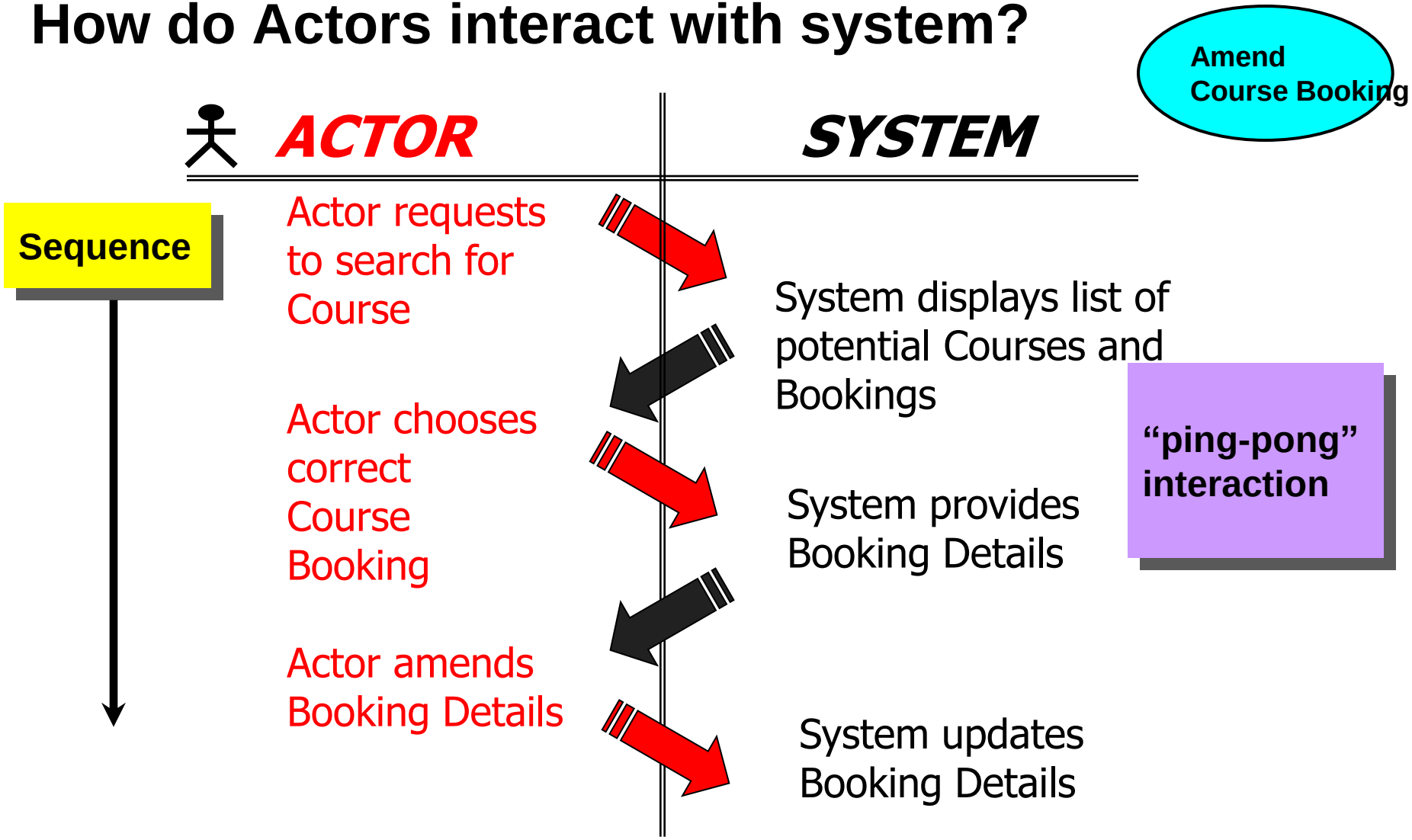
How to Identify Use Cases?

- **Identify candidate system actors**
- **Identify candidates Use Cases**
  - **probably from the Business Processes in the business process model chosen for system automation**
- **Scope units of interaction (Use Cases)**
  - **start point (look for actor and initial event)**
  - **end point (look for beneficial result for actor)**

Two versions of the process model
As is and 2 B

Use Case Specification

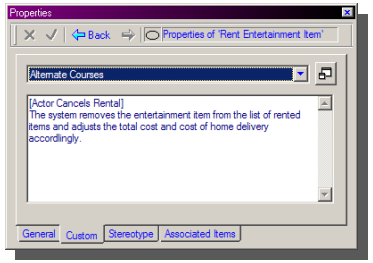
How do Actors interact with system?



Writing the Use Case Description

- Try to use ‘structured text’ ie
  - Simple statement:
    - ‘Actor does something’
    - ‘System does something’
  - Selection:
    - IF a condition is met
      *{simple statements or the name of an Alternate Course}
      Else
      *…
    - Select from following:
      Case: condition 1
      *…
      Case: Condition 2
      *…
      etc
  - Iteration:
    - While some condition is met …
      *…

Use Case Specification



Prototype

Name

Brief Description

Triggers

Pre Condition(s)

Post Condition(s)

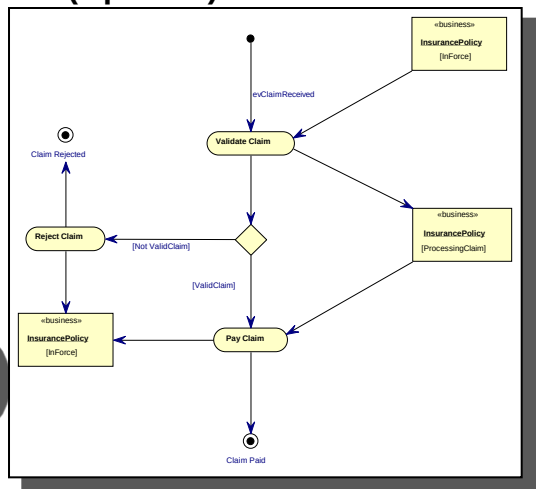
Inputs

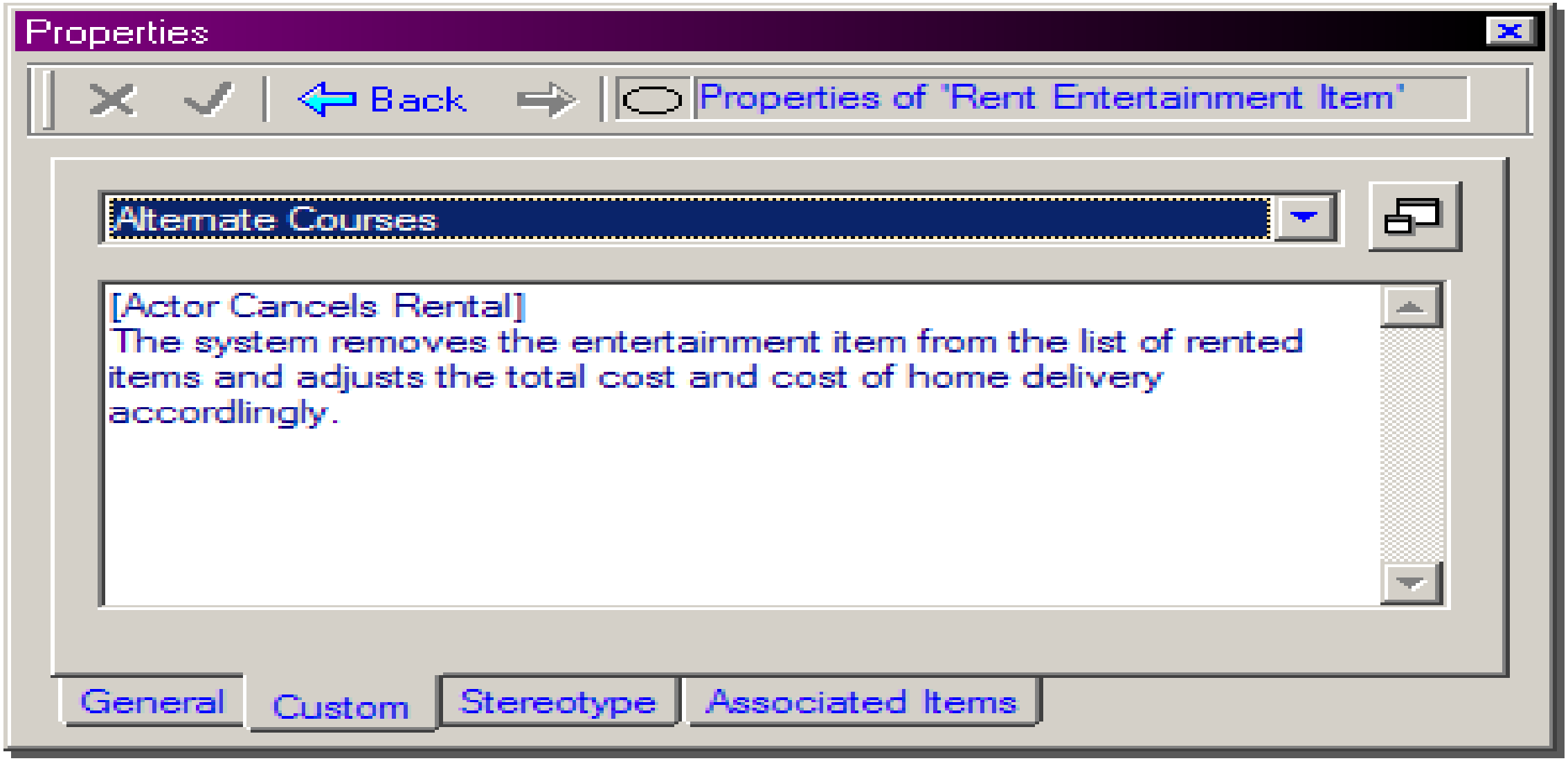
Outputs

Alternate Courses

Use Case Description

Activity Diagram
(Optional)




Properties
Back
Properties of "Rent Entertainment Item"
Alternate Courses
[Actor Cancels Rental]
The system removes the entertainment item from the list of rented items and adjusts the total cost and cost of home delivery accordingly.
General
Custom
Stereotype
Associated Items

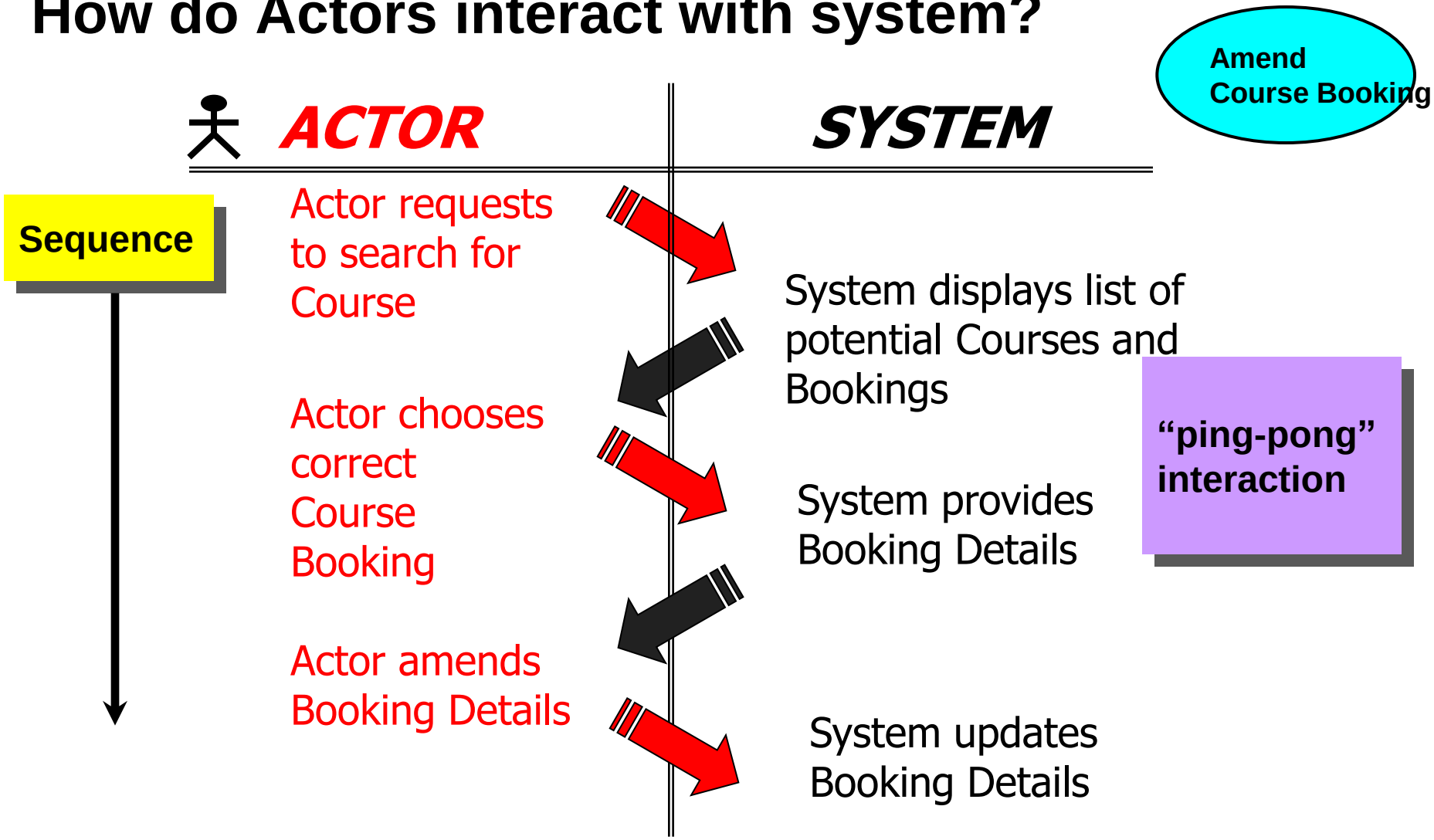
Prototype

A Use Case Description

Use case	Search books	
Actor	student	
trigger	Student needs to look for books	
Pre condition	System is running student is logged in	
Post condition	Student see the selected book list	
input	Category name	
output	none	
Main course	1 Actor	Enters the category name
	2 System	Accept the category name
	3 Actor	Press search
	4 System	Looks for books with this category and display their details and number of copies available
	5 Actor	Observe the result and terminate the search form (OK button)
	6 System	Ends the dialog and return control to the actor again
alternative	1.1 category name is wrong or empty book list, the system send s a message and ask the actor to retype category name	

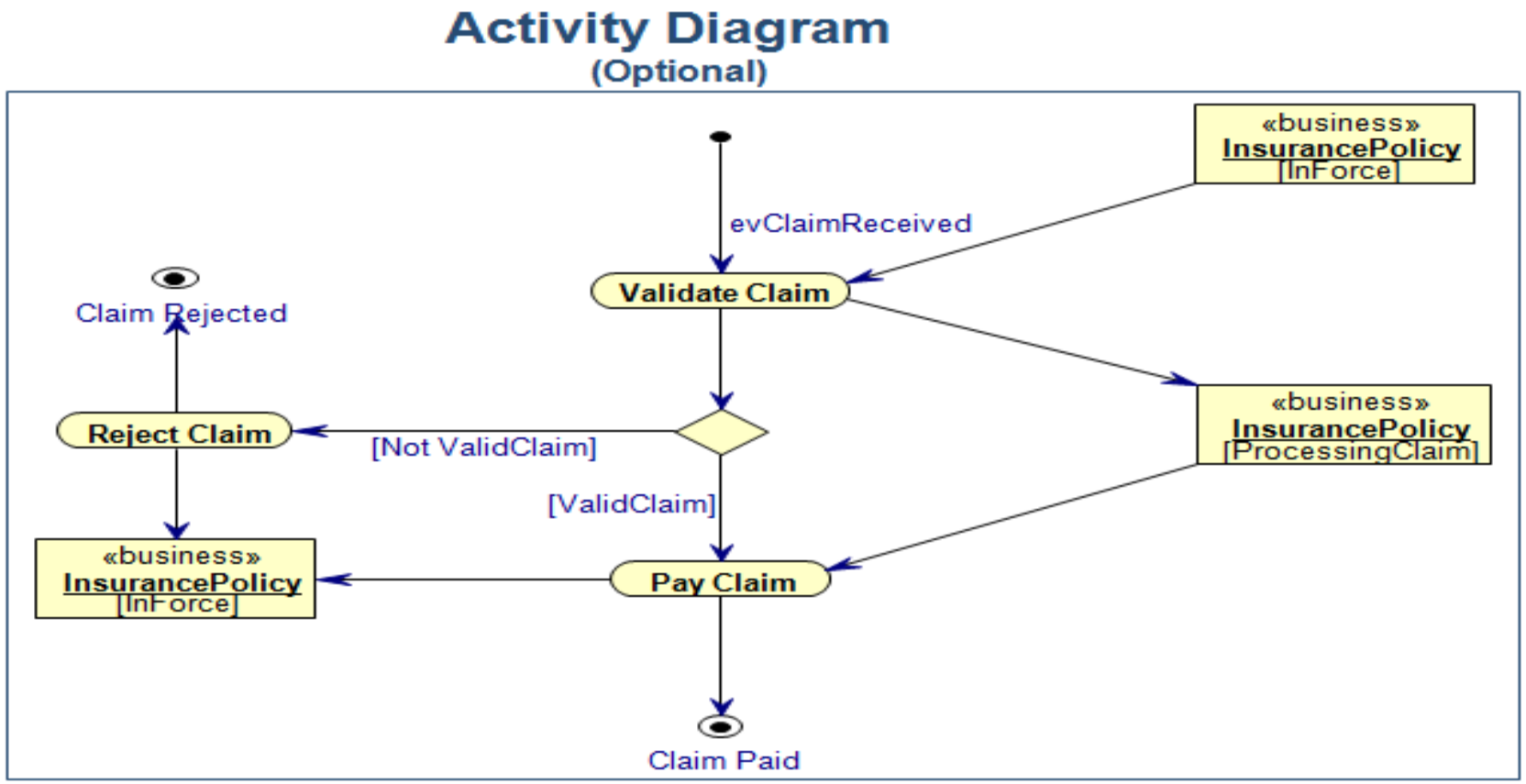
Use Case Specification

How do Actors interact with system?

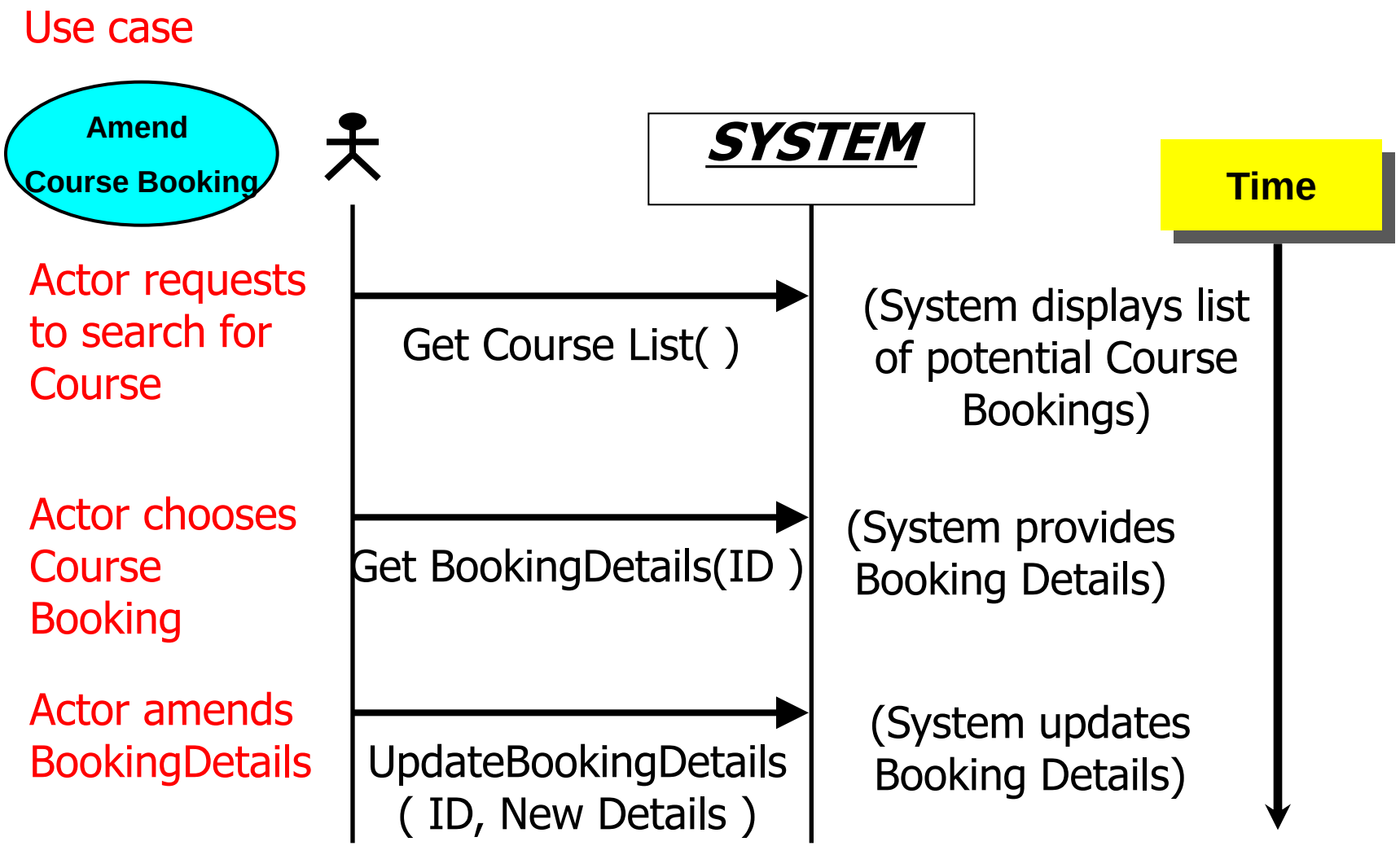


Use Case Description with Activity Diagram

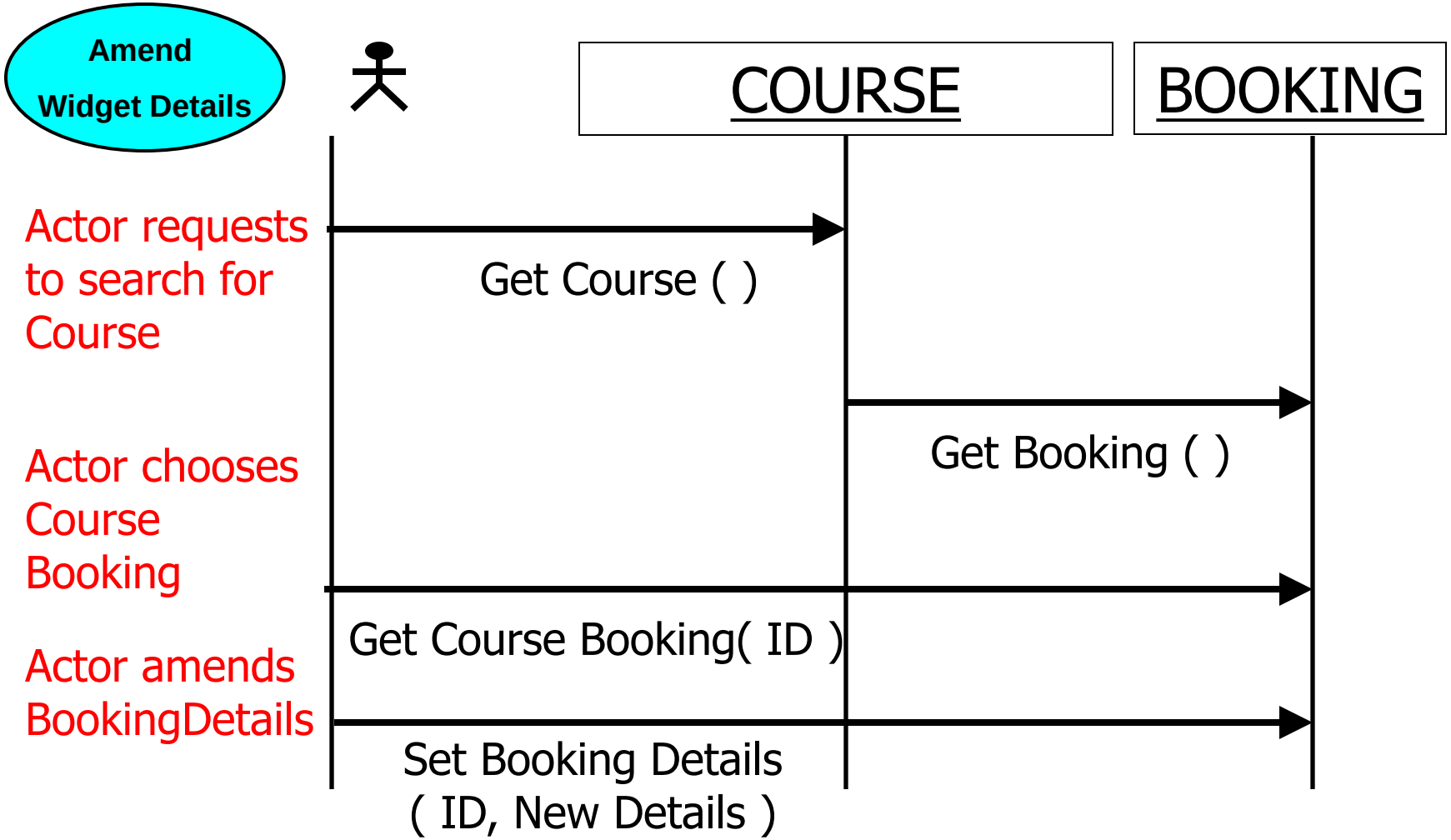
- Where there are many options/paths through the Use Case, an Activity Diagram can aid understanding



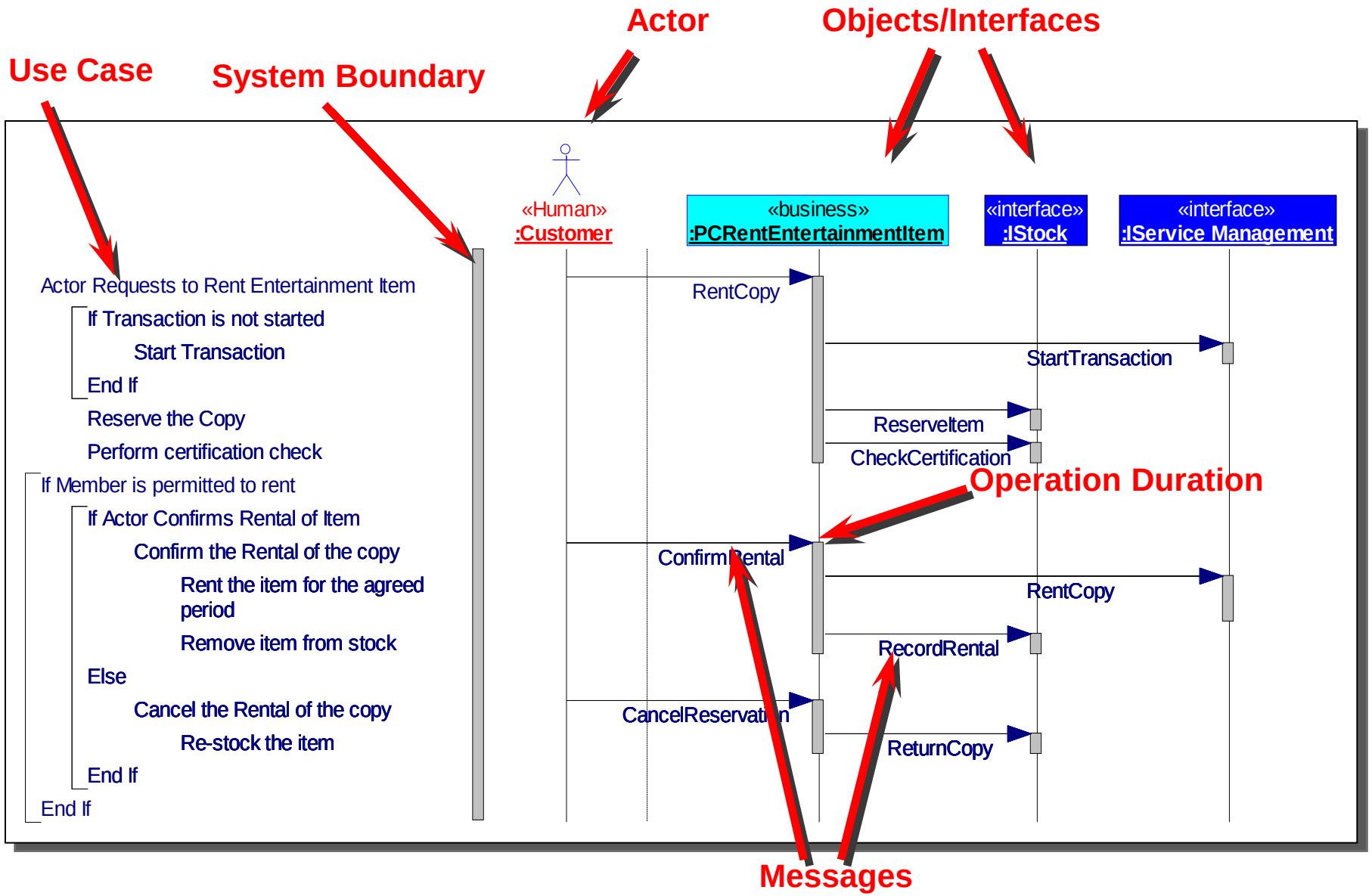
Interactions Become Messages



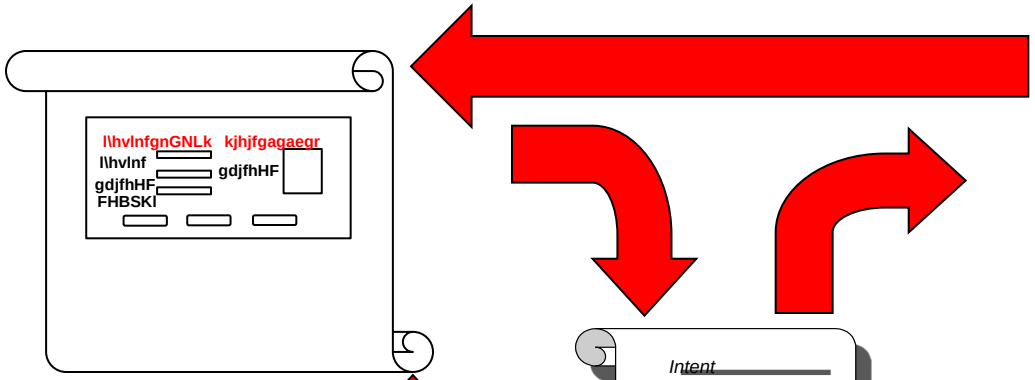
Object Interactions



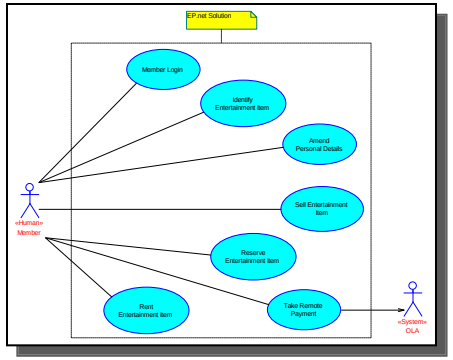
Object Sequence Diagram



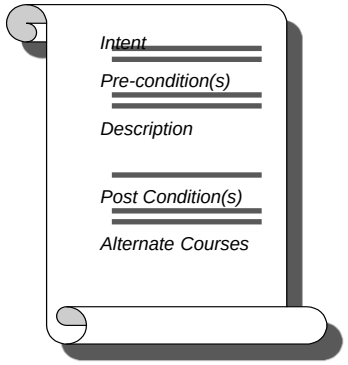
The Modelling Micro-iterations



Prototype

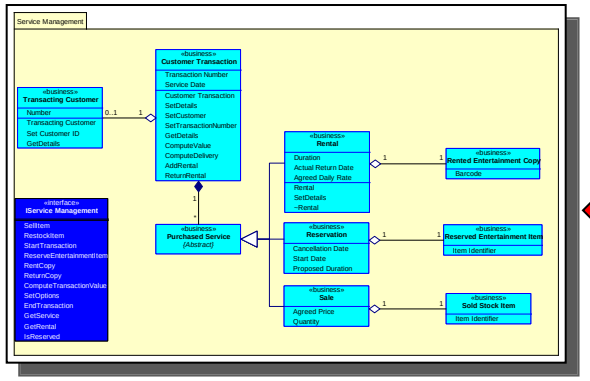


Use Case Diagram

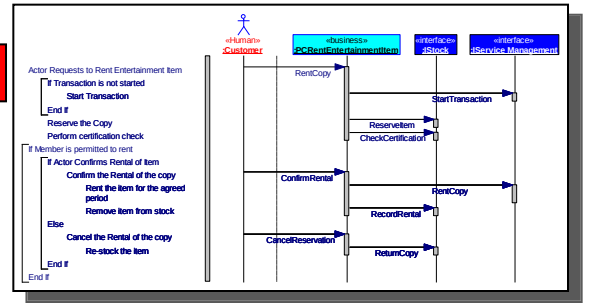


Use Case Description

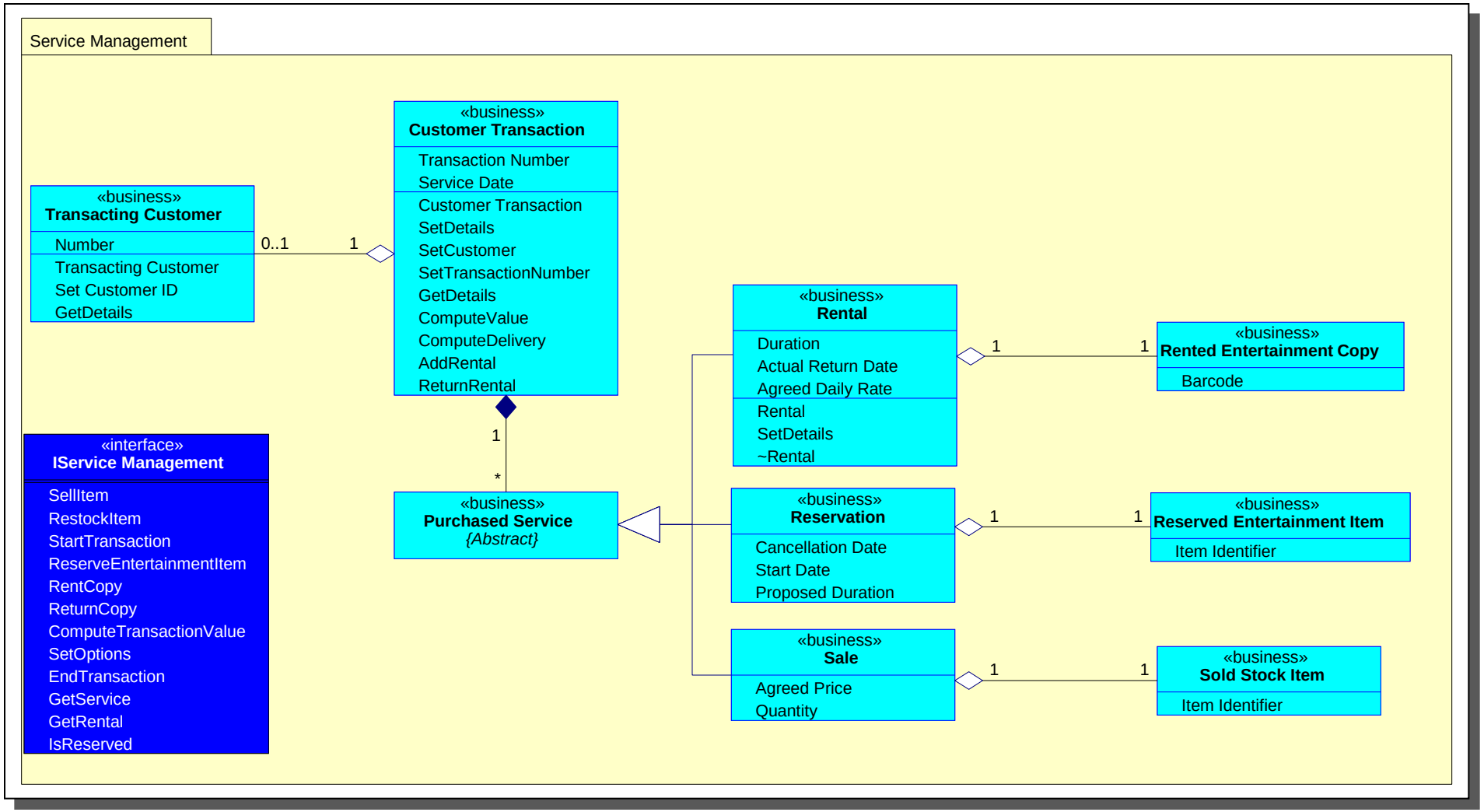
Class Model



Object Sequence Diagram



Class model


Service Management
«business»
Customer Transaction
Transaction Number
Service Date
Customer Transaction
SetDetails
SetCustomer
SetTransactionNumber
GetDetails
ComputeValue
ComputeDelivery
AddRental
ReturnRental
«business»
Transacting Customer
Number
Transacting Customer
Set Customer ID
GetDetails
0..1
1
1
*
«interface»
IService Management
SellItem
RestockItem
StartTransaction
ReserveEntertainmentItem
RentCopy
ReturnCopy
ComputeTransactionValue
SetOptions
EndTransaction
GetService
GetRental
IsReserved
«business»
Purchased Service
{Abstract}
«business»
Rental
Duration
Actual Return Date
Agreed Daily Rate
Rental
SetDetails
~Rental
«business»
Reservation
Cancellation Date
Start Date
Proposed Duration
«business»
Sale
Agreed Price
Quantity
1
1
1
1
1
1
«business»
Rented Entertainment Copy
Barcode
«business»
Reserved Entertainment Item
Item Identifier
«business»
Sold Stock Item
Item Identifier

Exercise

- ✓ **Create a use case for SUT LMS**
- ✓ **Write down a use case description for one of the use cases “enroll in a training course run by the user ”**

Summary

- Object-orientation offers a model-based approach to understanding the problem and structuring the solution

- The benefits of object-orientations include:
  - Flexibility in development
  - Ease of maintenance and the ability to extend system
  - Common language and model used throughout lifecycle

- The Unified Modelling Language (UML) is the standard graphical and textual Object-orientation modelling notation