Software Design and Analysis
CS-3004
Lecture#03

Dr. Javaria Imtiaz,
Mr. Basharat Hussain,
Mr. Majid Hussain

quest lab

Today’s Agenda

- Requirement Analysis (SDLC)
- What is Use Case?
- Identification of Use Cases
- Use Case Diagrams
- Components of Use Case Diagram
- Examples

Requirements Analysis and Use-Case Diagrams

How are design ideas communicated in a team environment?

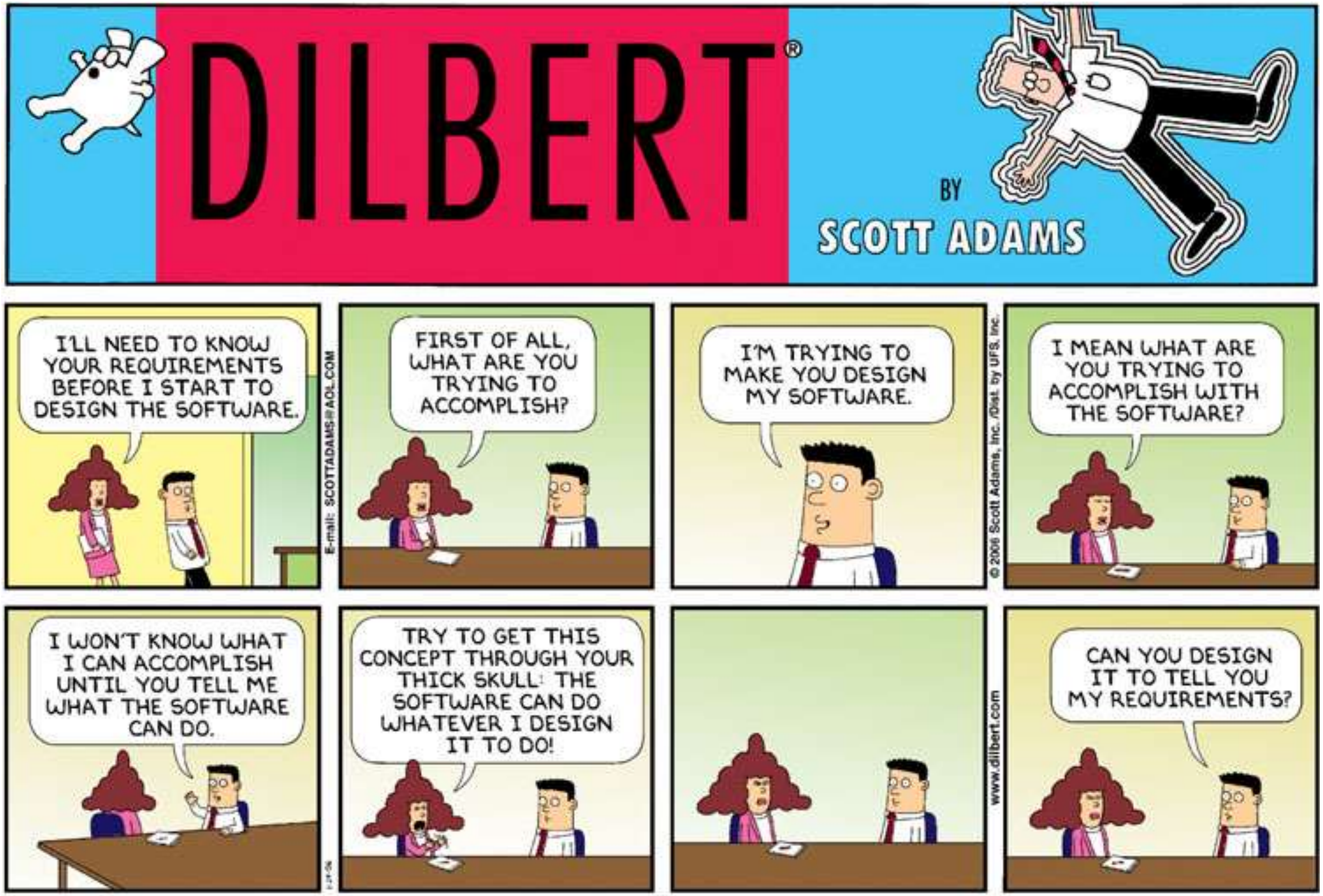
- If the software is large scale, employing perhaps dozens of developers over several years, it is important that all members of the development team communicate using a common language.
- This isn't meant to imply that they all need to be fluent in English or C++, but it does mean that they need to be able to describe their software's operation and design to another person.
- That is, the ideas in the head of say the analyst have to be conveyed to the designer in some way so that he/she can implement that idea in code.
- Just as mathematicians use algebra and electronics engineers have evolved circuit notation and theory to describe their ideas, software engineers have evolved their own notation for describing the architecture and behaviour of software system.
- That notation is called UML. The Unified Modelling Language. Some might prefer the title Universal Modelling language since it can be used to model many things besides software.

What is UML ?

- UML is not a language in the same way that we view programming languages such as ‘C++’, ‘Java’ or ‘Basic’.
- UML is however a language in the sense that it has syntax and semantics which convey meaning, understanding and constraints (i.e. what is right and wrong and the limitations of those decisions) to the reader and thereby allows two people fluent in that language to communicate and understand the intention of the other.
- UML represents a collection of graphical notations to capture requirements and design alternatives.
- UML is to software engineers what building plans are to an architect and an electrical circuit diagrams is to an electrician.
- UML is suitable for large scale projects

Phases of System Development

- Requirement Analysis
  - The functionality users require from the system
  - **Use-case model**
- OO Analysis
  - Discovering classes and relationships
  - Class diagram
- OO Design
  - Result of Analysis expanded into technical solution
  - Sequence diagram, state diagram, etc.
  - Results in detailed specs for the coding phase
- Implementation (Programming/coding)
  - Models are converted into code
- Testing
  - Unit tests, integration tests, system tests and acceptance tests.


DILBERT®
BY SCOTT ADAMS
I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?
E-mail: SCOTTADAMS@AOL.COM
© 2006 Scott Adams, Inc. /Dist. by UFS, Inc.
www.dilbert.com

Definition of a Use-Case

- A process or procedure, describing a *user's interaction* with the system (e.g. library) for a *specified, identifiable purpose.* (e.g. borrowing a book).

- As such, each Use-Case describes a step-by-step sequence of operations, iterations and events that document

  - The Interaction taking place.
  - The Measurable Benefits to the user interacting with the system.
  - The Effect of that Interaction on the system.

- It is important to document these use-cases as fully as possible as each use-case captures some important functionality that our system will have to provide in the automated version of the library.

Flex

Register Course

ATM

Withdraw Money

Bank

Apply for Loan

Online Shopping Management system

Make Purchase

Library

Issue Book

OSM

Make Payment

HMS

Order Food

FMS

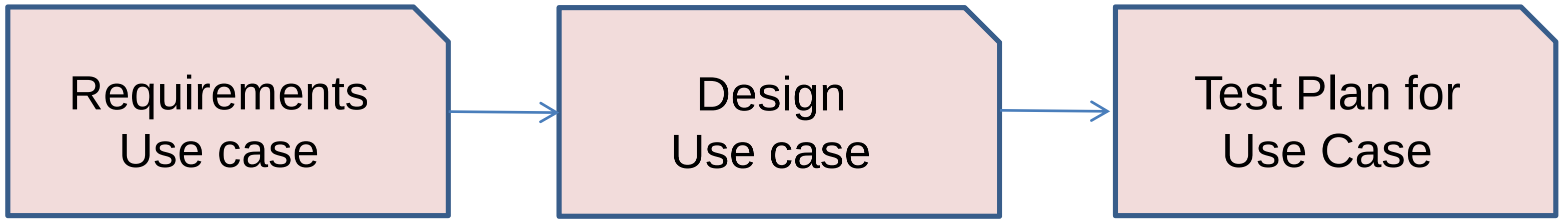
Book Online Ticket

Use Cases in Iterative Development

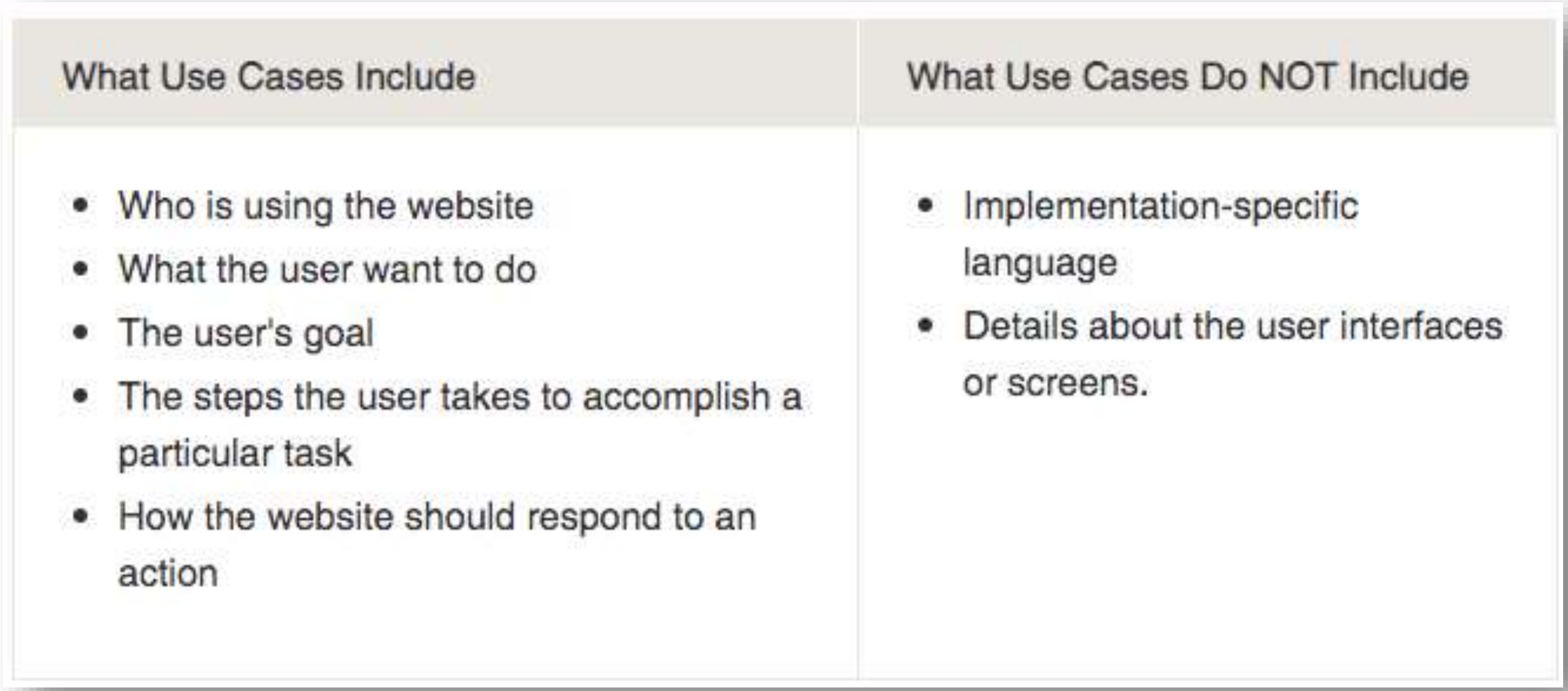
- Functional requirements are primarily captured in use cases
- Use cases drive the iteration planning and work Easy for users to understand
- Influence user manual/documentation
- Functional or system testing corresponds to the scenarios of use cases
- Independent of implementing technology UI shortcuts for most common scenarios

Benefits – Development Life Cycle

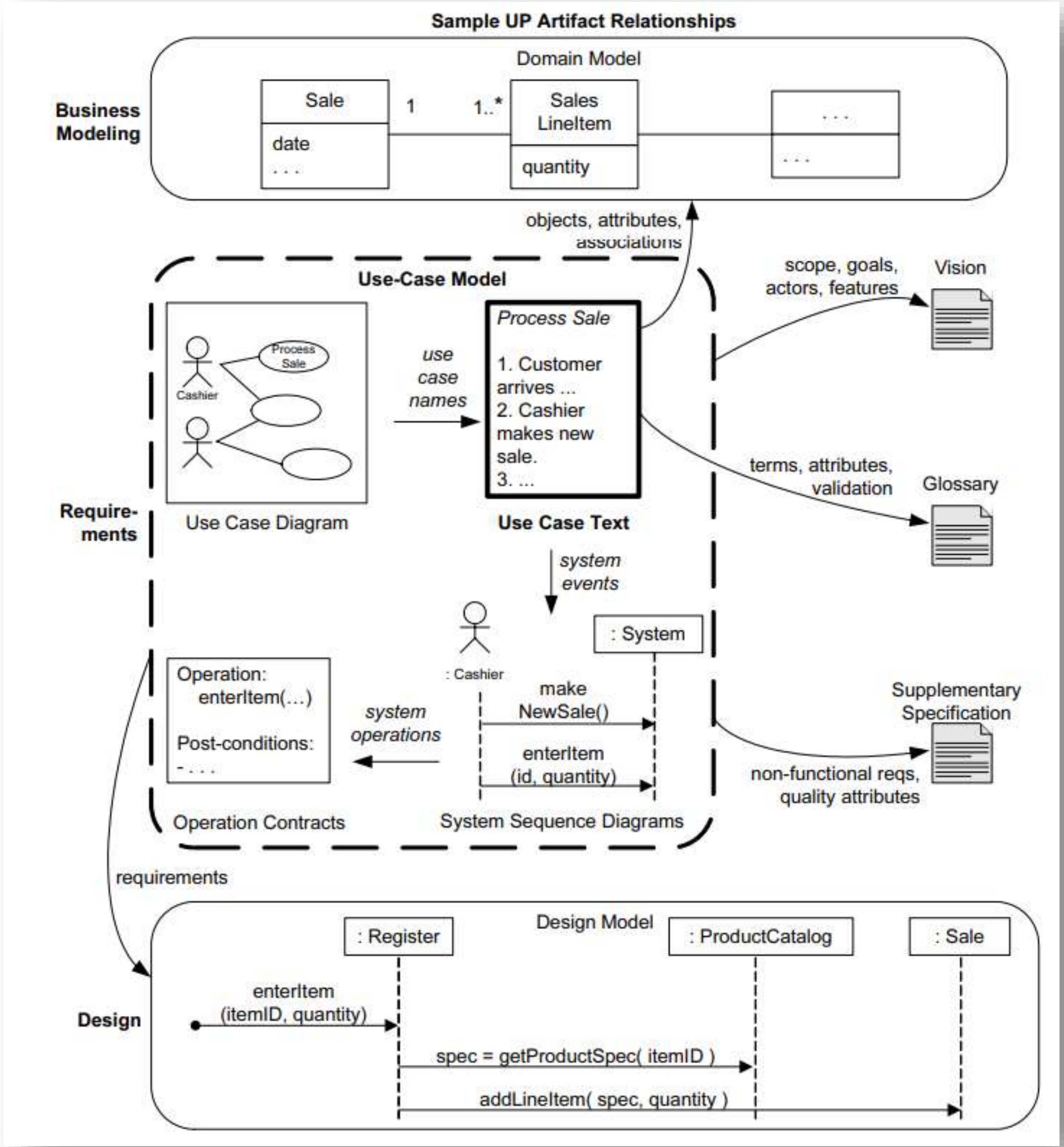
- Use cases started for requirements become the vehicle for the following activities:
  - Detailing them to show the corresponding design work
  - Development of test plans for acceptance testing
  - Creation of interaction design specifics, simply by adding detail to the use cases about "how" the actions will be done
  - The basis for discovering a great OO design, based on the actors and their actions



What ‘should’ and what ‘shouldn’t ‘ be included in use cases?



What Use Cases Include	What Use Cases Do NOT Include
• Who is using the website • What the user want to do • The user's goal • The steps the user takes to accomplish a particular task • How the website should respond to an action	• Implementation-specific language • Details about the user interfaces or screens.


Sample UP Artifact Relationships
Business Modeling
Domain Model
Sale
date
. . .
1
1..*
Sales LineItem
quantity
. . .
. . .
objects, attributes, associations
Use-Case Model
Process Sale
Cashier
Use Case Diagram
use case names
Process Sale
1. Customer arrives ...
2. Cashier makes new sale.
3. ...
Use Case Text
scope, goals, actors, features
Vision
terms, attributes, validation
Glossary
Requirements
system events
: Cashier
: System
make NewSale()
enterItem (id, quantity)
System Sequence Diagrams
Operation: enterItem(…)
Post-conditions: - . . .
Operation Contracts
system operations
Supplementary Specification
non-functional reqs, quality attributes
requirements
Design Model
: Register
: ProductCatalog
: Sale
Design
enterItem (itemID, quantity)
spec = getProductSpec(itemID)
addLineItem(spec, quantity)

How to Find Use Cases?

- Choose the system boundary
  - what you are building?
  - who will be using the system?
  - what else will be used that you are not building?
- Find primary actors and their goals
  - brainstorm the primary actors first who starts and stops the system?
  - who gets notified when there are errors or failures?
- Define use cases that satisfy user goals
  - prepare an actor-goal list (and not actor-task list) in general, one use case for each user goal
  - name the use case similar to the user goal

What Tests Can Help Find Useful Use Cases?

- Which of these are valid use cases?
  - Negotiate a Supplier Contract
  - Handle Returns
  - Log in
  - Move Piece on the Game Board

What Tests Can Help Find Useful Use Cases?

- Which of these are valid use cases?
  - Negotiate a Supplier Contract
  - Handle Returns
  - Log in
  - Move Piece on the Game Board

All of these can be use cases at different levels, depending on the system, boundary, actors, and goals

What Tests Can Help Find Useful Use Cases?

- Rather than asking
  - ”What is a valid use case?” More practical question:
  - “What is a useful level of focus to express use cases for application requirements analysis?”
- Rules of thumb
  - The Boss Test
  - The EBP Test
  - The size test

What Tests Can Help Find Useful Use Cases?

- **The boss test**

  "What have you been doing all day?"

  - Your reply "logging in!"
    - Is your boss happy? No value? No good use case!

- **The Elementary Business Process (EBP) test**

  a task performed by one person in one place at one time, in response to a business event, which adds measurable business value and leaves data in a consistent state

  Good Examples: Approve Credit or Price Order

  Bad Examples: delete a line item or print the document

- **The size test**

  - Just a single step in a sequence of others -> not good!

Applying Tests

- Negotiate a supplier contract
  - Much broader than EBP, rather a business use case
- Handle returns
  - OK with the Boss. EBP. Size is good.
- Log in
  - Boss is not happy is this is all you do all day!
- Move piece on game board
  - Single step – fails the size test.

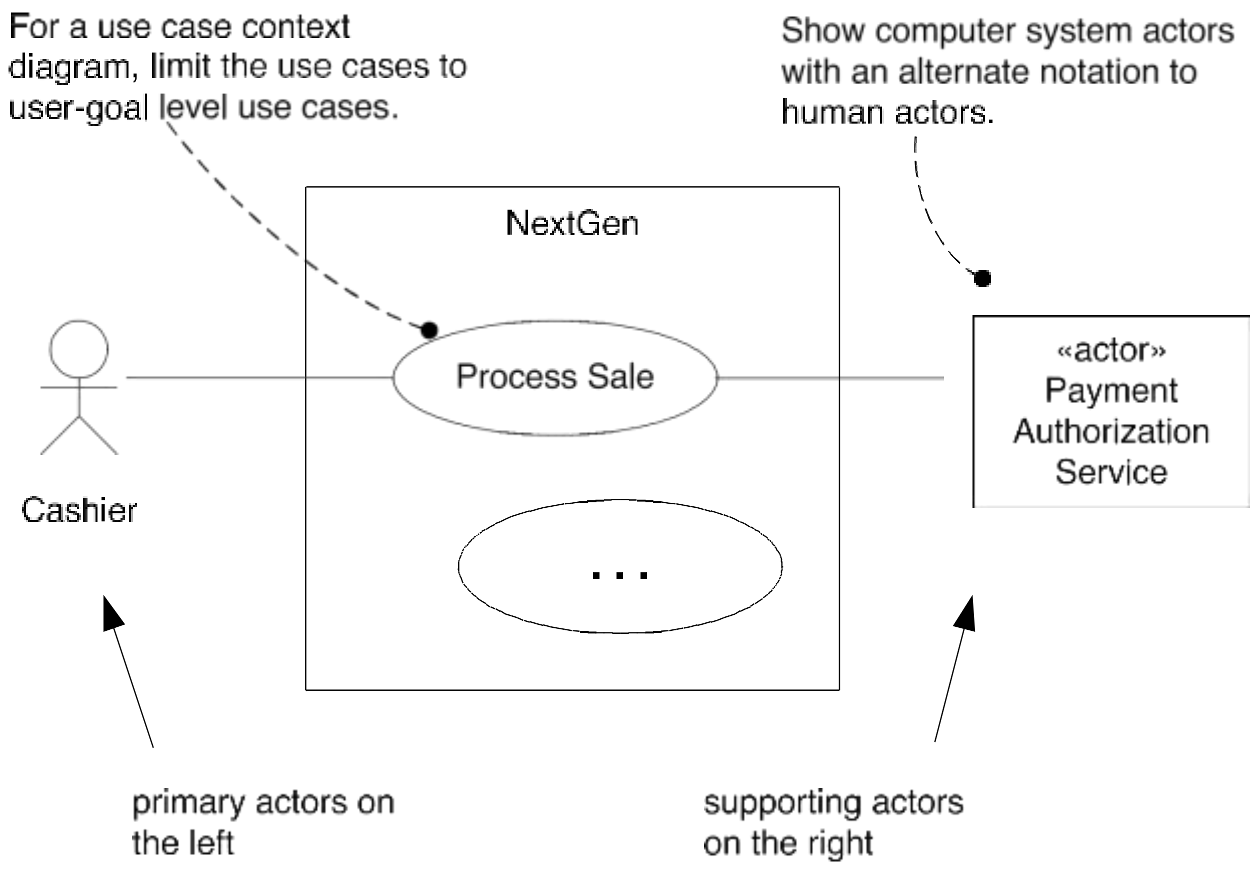
USE CASE DIAGRAM

quest lab

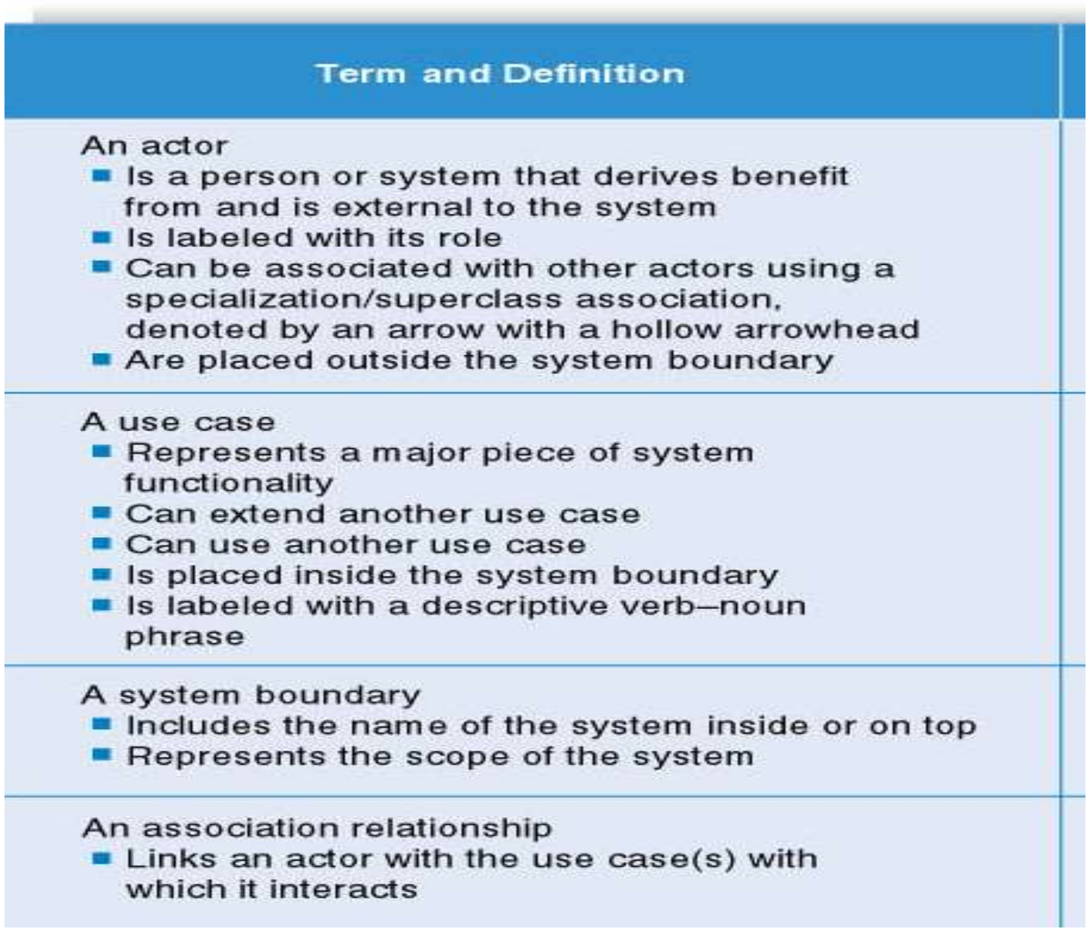
Use Case Diagram Concepts

- Summarizes all use cases (for the part of the system being modeled) together in one picture.
- Typically drawn early in the SDLC
- Shows the associations between actors and use cases

Use Case (Context) Diagrams: Suggested Notation

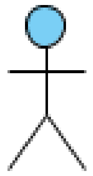

For a use case context diagram, limit the use cases to user-goal level use cases.
Show computer system actors with an alternate notation to human actors.
NextGen
Process Sale
. . .
Cashier
«actor» Payment Authorization Service
primary actors on the left
supporting actors on the right

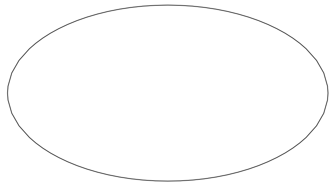
Syntax for Use Case Diagram



Term and Definition
An actor ■ Is a person or system that derives benefit from and is external to the system ■ Is labeled with its role ■ Can be associated with other actors using a specialization/superclass association, denoted by an arrow with a hollow arrowhead ■ Are placed outside the system boundary
A use case ■ Represents a major piece of system functionality ■ Can extend another use case ■ Can use another use case ■ Is placed inside the system boundary ■ Is labeled with a descriptive verb—noun phrase
A system boundary ■ Includes the name of the system inside or on top ■ Represents the scope of the system
An association relationship ■ Links an actor with the use case(s) with which it interacts

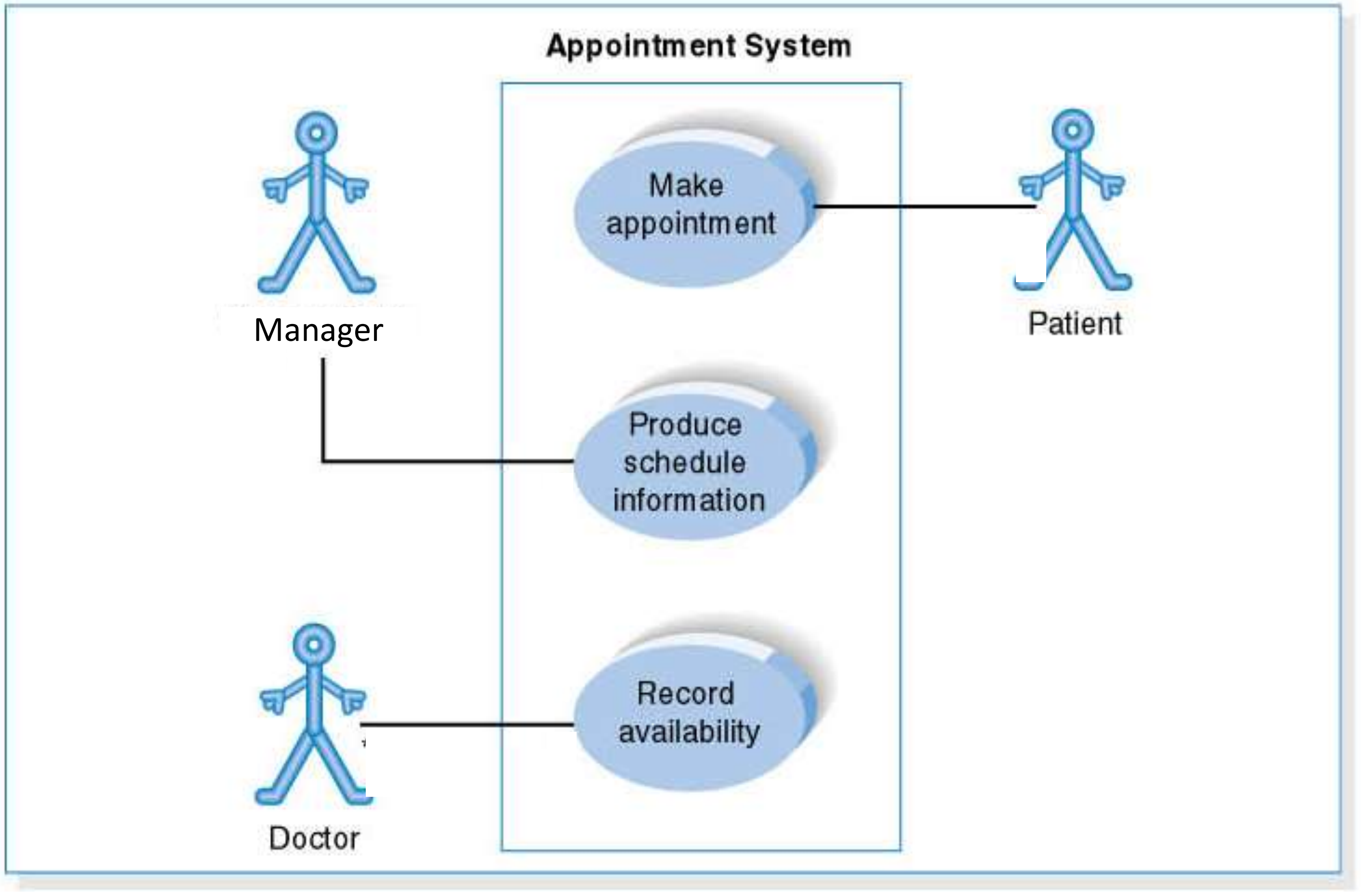
Notations





Name of the system

Use Case Diagram for Appointment System



Revision

- Why to create a use case diagram?
- Major components of Use case diagram
- Boundary
- Actor, types, place
- What is use case? use cases are not diagrams, they are text
- What are the three test cases use to identify the valid or invalid use cases?

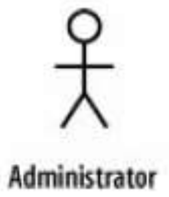
Example

Requirement 1

The content management system shall allow an *administrator* to create a new blog account, provided the personal details of the new blogger are verified using the author credentials database.

The requirement indicates that that the ***Administrator*** interacts with the system to create a new blogger's account

The ***Administrator*** *interacts with* the system and is *not part of* the system; thus, the Administrator Is an ***Actor***.



It's actually worth being very careful when naming your actors. The best approach is to use a name that can be understood by both your customer *and* your system designers.

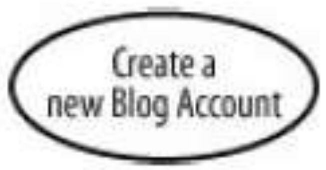
Example

- What can be the use case in our Requirement 1?

Requirement 1
The content management system shall allow an *<u>administrator</u>* to create a new blog account, provided the personal details of the new blogger are verified using the author credentials database.

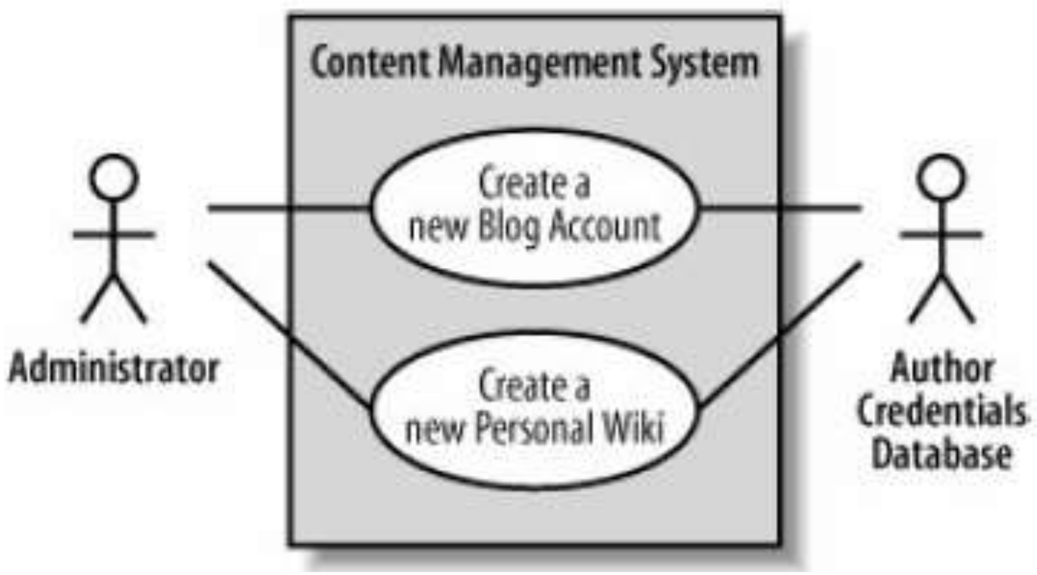
to create a new blog account

<u>UML Representation</u>

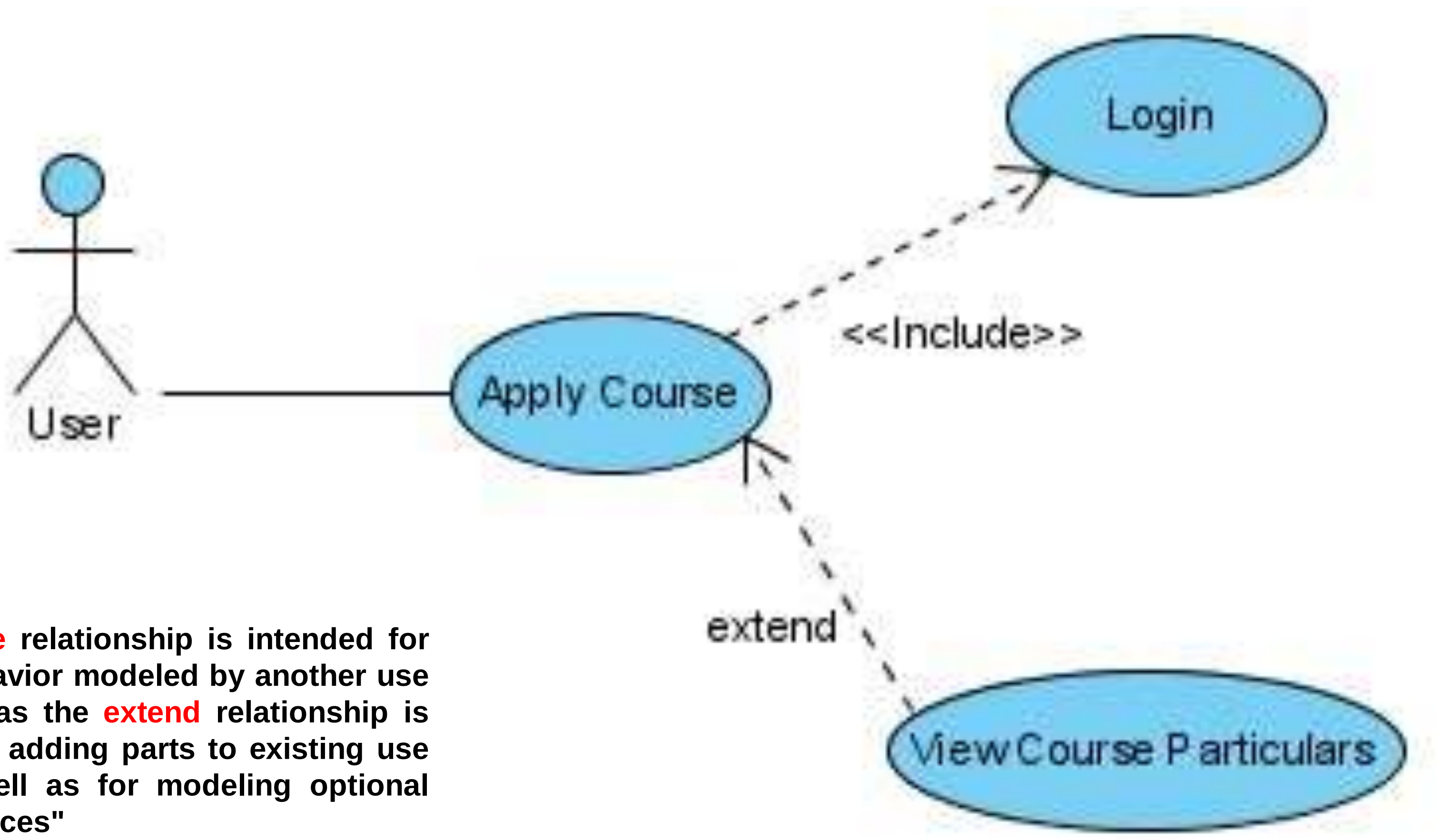


Example (Cont)

Requirement 2
The content management system shall allow an administrator to create a new personal Wiki, provided the personal details of the applying author are verified using the Author Credentials Database.

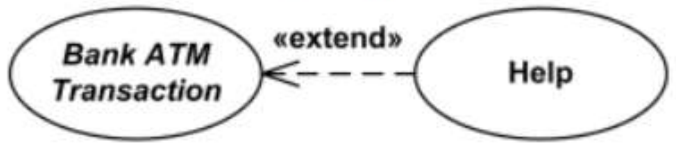
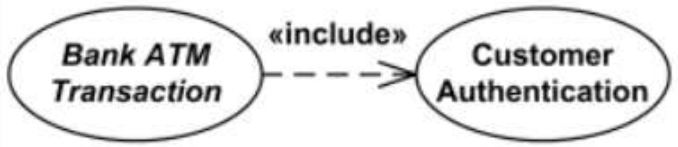

Content Management System
Create a new Blog Account
Create a new Personal Wiki
Administrator
Author Credentials Database

Example – Include and Extend



"The include relationship is intended for reusing behavior modeled by another use case, whereas the extend relationship is intended for adding parts to existing use cases as well as for modeling optional system services"

Include and Extend Relationships

Extend	Include
	 Bank ATM Transaction «include» Customer Authentication
Base use case is complete (concrete) by itself, defined independently.	Base use case is incomplete (**abstract use case**).
Extending use case is optional, supplementary.	Included use case required, not optional.
Has at least one explicit extension location.	No explicit inclusion location but is included at some location.
Could have optional extension condition.	No explicit inclusion condition.

Include and Extend

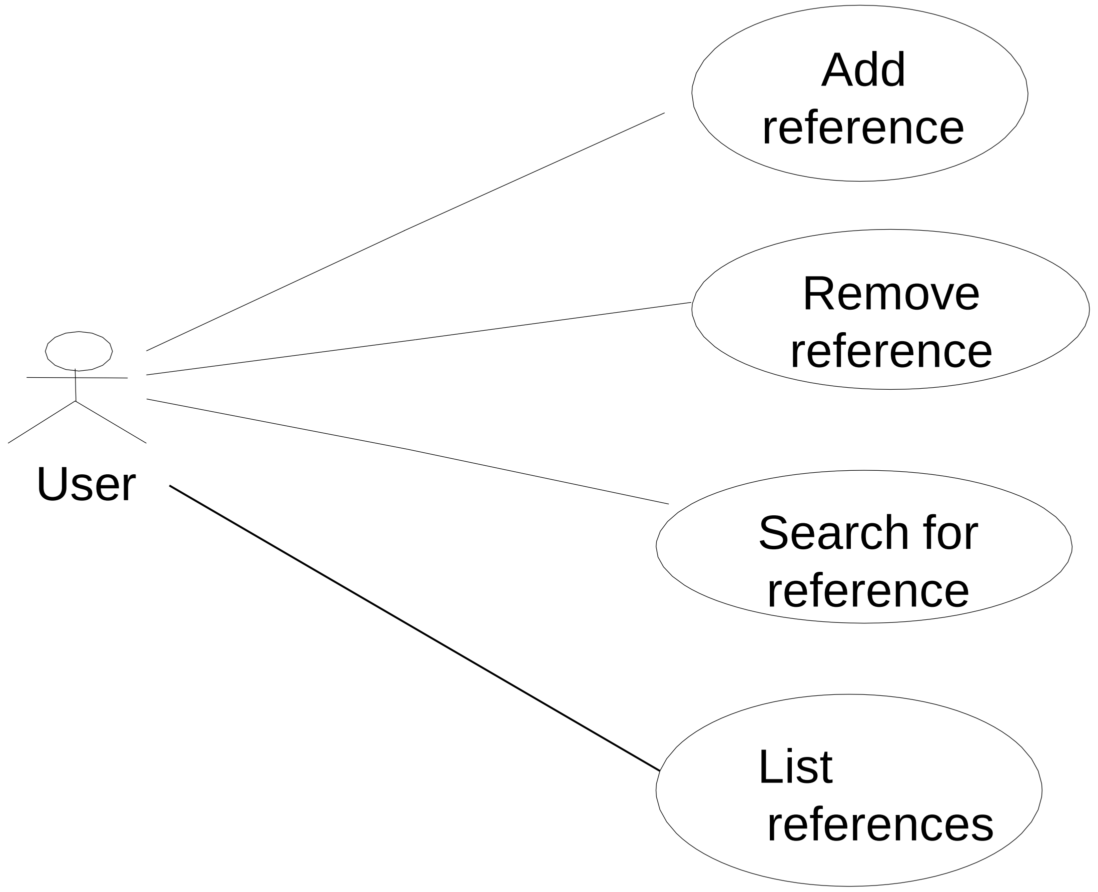
- The use of include and extend is discouraged simply because they add unnecessary complexity to a Use Case diagram.
- Since the primary purpose of use cases is to show user centered functionality, the precedence of use cases takes little importance.

Use Case Diagram– Guidelines & Caution

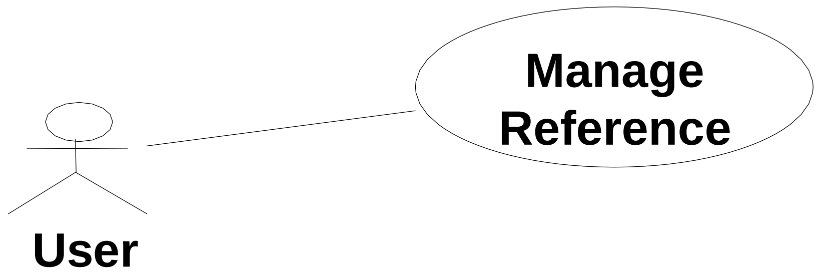
- Actors should be Use cases should ideally **begin with a verb** – i.e generate report.
- Use cases **should NOT be open ended** – i.e Register (instead should be named as Register New User)

- Avoid showing communication between actors.
  - named as singular. i.e., student and NOT students. NO names should be used – i.e John, Sam, etc.
  - Do NOT show behavior in a use case diagram; instead only depict only system functionality.
  - Use case diagram does not show sequence – unlike DFDs.

Granularity of Use Cases

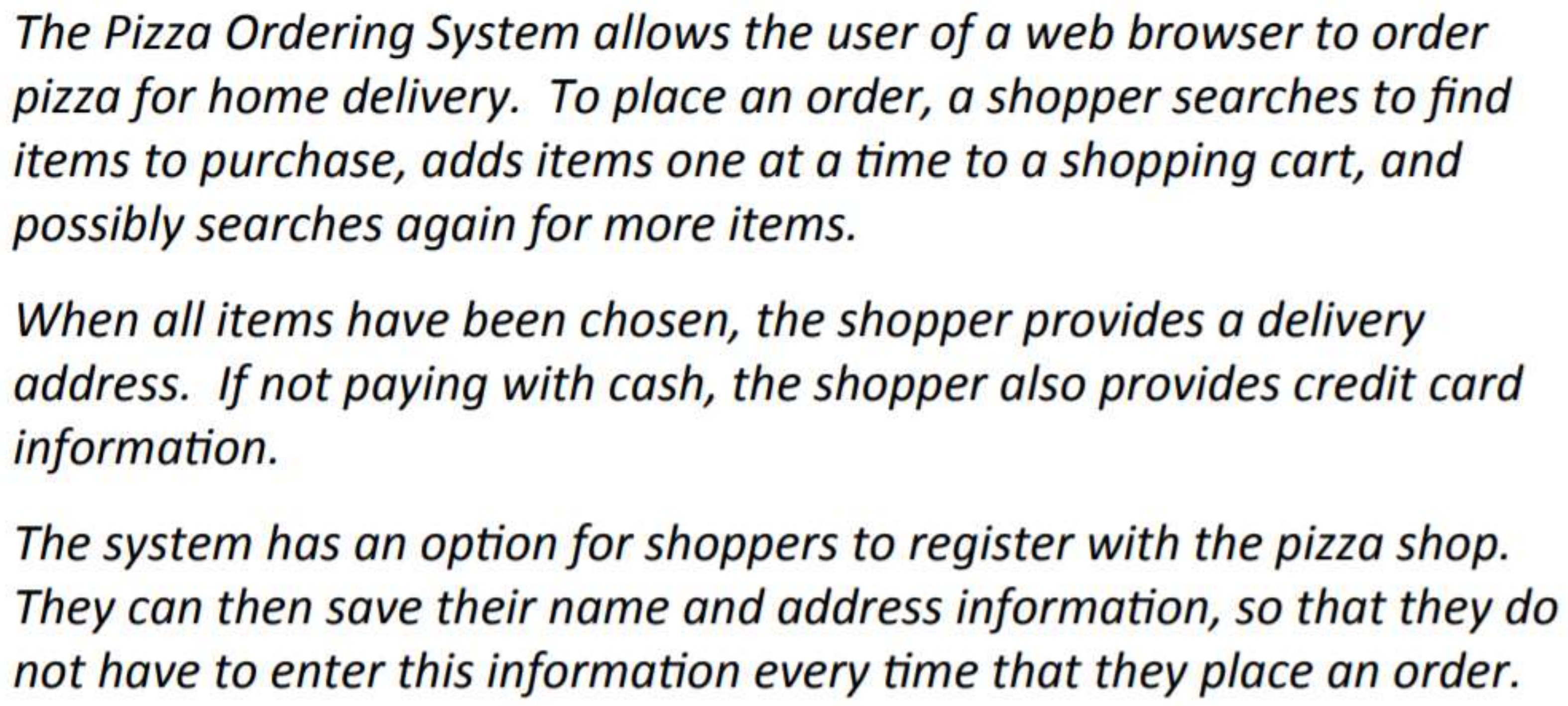


Granularity of Use Cases



Generally, a use case should embody sufficient levels of granularity without which the use case may not be rendered as useful.

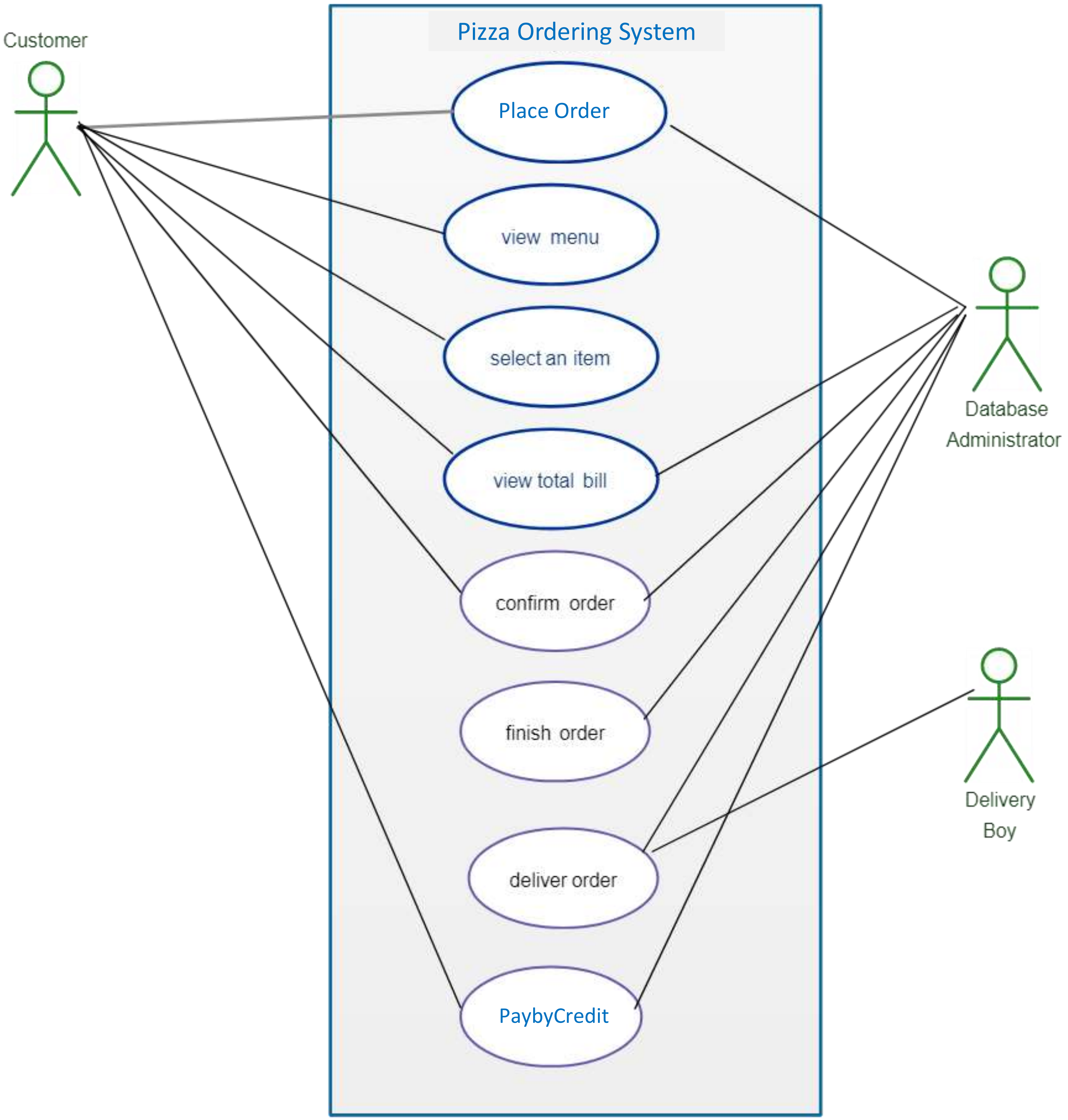
Pizza Ordering System



The Pizza Ordering System allows the user of a web browser to order pizza for home delivery. To place an order, a shopper searches to find items to purchase, adds items one at a time to a shopping cart, and possibly searches again for more items.

When all items have been chosen, the shopper provides a delivery address. If not paying with cash, the shopper also provides credit card information.

The system has an option for shoppers to register with the pizza shop. They can then save their name and address information, so that they do not have to enter this information every time that they place an order.


Pizza Ordering System
Customer
Place Order
view menu
select an item
view total bill
confirm order
finish order
deliver order
PaybyCredit
Database Administrator
Delivery Boy

Revision

- How to find Use Cases
  - **Choose a system boundary** : Everything outside SuD, easy to find once external actors are identifies
  - **Identify Actors:** Actual Person, Something you cannot change in a SuD
  - **Identify User Goals**
  - **Define Use Cases**

Library Management System

- Any library member should be able to search books by their title, author, subject category as well by the publication date. Each book will have a unique identification number and other details including a rack number which will help to physically locate the book. There could be more than one copy of a book, and library members should be able to check-out and reserve any copy. We will call each copy of a book, a book item. The system should be able to retrieve information like who took a particular book or what are the books checked-out by a specific library member. There should be a maximum limit (5) on how many books a member can check-out. There should be a maximum limit (10) on how many days a member can keep a book. The system should be able to collect fines for books returned after the due date. Members should be able to reserve books that are not currently available. The system should be able to send notifications whenever the reserved books become available, as well as when the book is not returned within the due date. Each book and member card will have a unique barcode. The system will be able to read barcodes from books and members' library cards.

Requirements

1. Any library member should be able to search books by their title, author, subject category as well by the publication date.

2. Each book will have a unique identification number and other details including a rack number which will help to physically locate the book.

3. There could be more than one copy of a book, and library members should be able to check-out and reserve any copy. We will call each copy of a book, a book item.

4. The system should be able to retrieve information like who took a particular book or what are the books checked-out by a specific library member.

5. There should be a maximum limit (5) on how many books a member can check-out.

6. There should be a maximum limit (10) on how many days a member can keep a book.

7. The system should be able to collect fines for books returned after the due date.

8. Members should be able to reserve books that are not currently available.

9. The system should be able to send notifications whenever the reserved books become available, as well as when the book is not returned within the due date.

10. Each book and member card will have a unique barcode. The system will be able to read barcodes from books and members' library cards.

Requirements

1. Any library member should be able to **search books** by their title, author, subject category as well by the publication date.
2. Each book will have a unique identification number and other details including a rack number which will help to physically locate the book.
3. There could be more than one copy of a book, and library members should be able to **check-out** and **reserve any copy**. We will call each copy of a book, a book item.
4. The system should be able to **retrieve information** like who took a particular book or what are the books checked-out by a specific library member.
5. There should be a maximum limit (5) on how many books a member can check-out.
6. There should be a maximum limit (10) on how many days a member can keep a book.
7. The system should be able to **collect fines** for books returned after the due date.
8. Members should be able to **reserve books** that are not currently available.
9. The system should be able to **send notifications** whenever the reserved books become available, as well as when the book is not returned within the due date.
10. Each book and member card will have a unique barcode.
11. The system will be able to **read barcodes** from books and members' library cards.

Actors

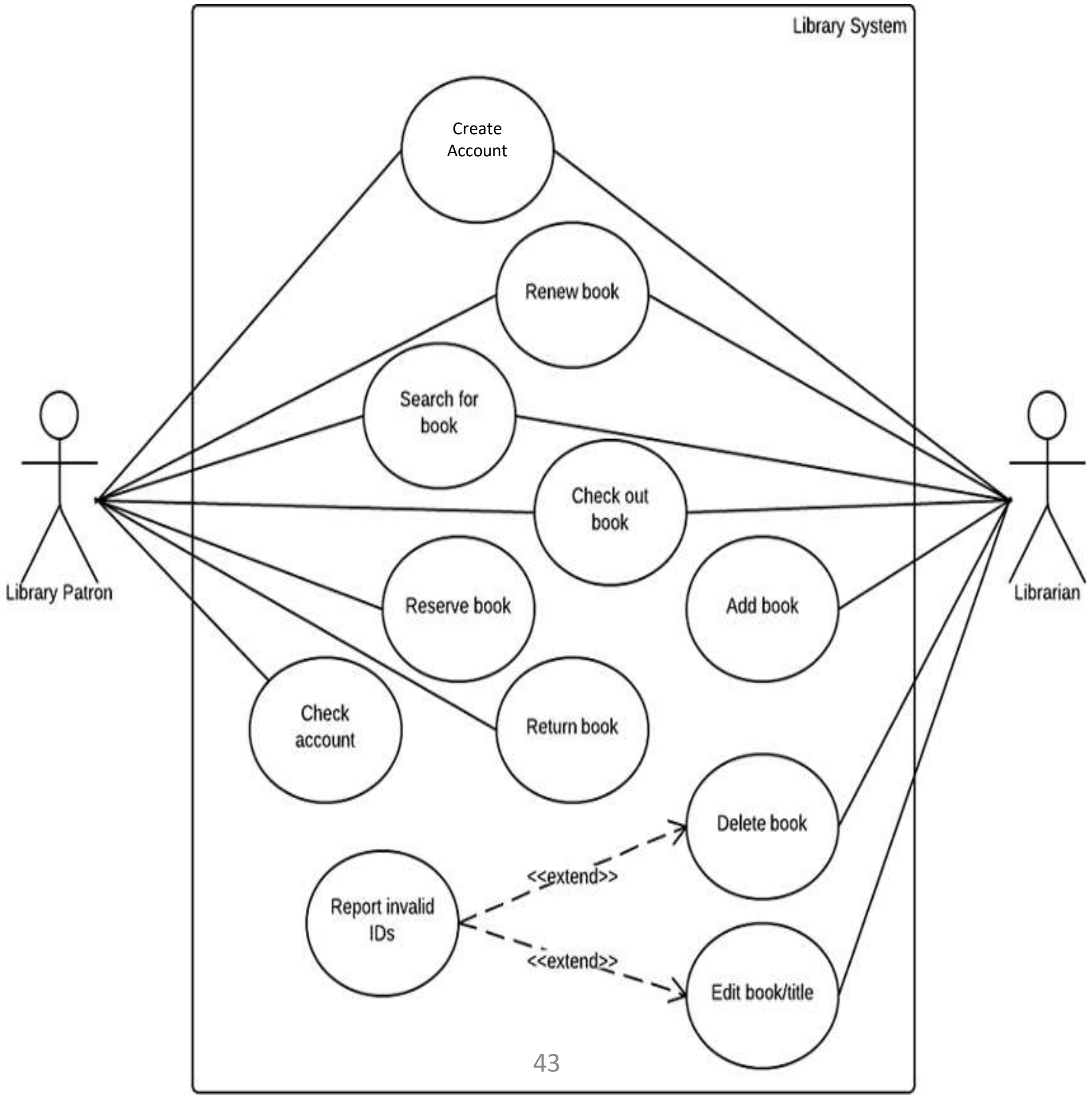
- **Librarian:** Mainly responsible for adding and modifying books, book items, and users. The Librarian can also issue, reserve, and return book items.

- **Patron:** All members can search the catalog, as well as check-out, reserve, renew, and return a book.

User Goals

- **Add/Remove/Edit book:** To add, remove or modify a book or book item.
- **Search catalog:** To search books by title, author, subject or publication date.
- **Register new account/cancel membership:** To add a new member or cancel the membership of an existing member.
- **Check-out book:** To borrow a book from the library.
- **Reserve book:** To reserve a book which is not currently available.
- **Renew a book:** To reborrow an already checked-out book.
- **Return a book:** To return a book to the library which was issued to a member.

Use Case Diagram for Library Management System



Identify Actors

Driver
Hostess
Customer
Company Staff/ Administrator

Case

Driver and hostess cannot be an Actor. Why?

Consider an online reservation system for a bus company. The bus company includes several buses and manage trips to different cities. Each bus is identified by its plate number and a separately assigned bus number. The trips are based on a predefined schedule and stop at predefined bus stations. Each bus can have only one trip per day. Each bus includes a driver and one hostess. For long trips, the bus will have breaks at service and rest areas. There are two types of trips, normal trips and express trips. Express trips do not stop at intermediate stations and get faster at the destination. Seats can be reserved by customers on the web site of the bus company. The customer has the option to directly pay for the seat through the website. In that case, the seat cannot be cancelled (neither by the customer nor by the bus company). If the customer has not paid for the seat, the bus company can cancel the seat if the customer does not show up one hour before the trip. When the reservation is cancelled, the seat will become free and can be sold to another customer. Both the customer and the company staff must authenticate themselves for performing operations with the system.

Identify Verb / Goal/Use Case

- Reserve Seat
- Pay for the Seat
- Cancel Seat
- Register
- View Seat

Case

Consider an online reservation system for a bus company. The bus company includes several buses and realizes trips to different cities. Each bus is identified by its plate number and a separately assigned bus number. The trips are based on a predefined schedule and stop at predefined bus stations. Each bus can have only one trip per day. Each bus includes a driver and one hostess. For long trips, the bus will have breaks at service and rest areas. There are two types of trips, normal trips and express trips. Express trips do not stop at intermediate stations and get faster at the destination. Seats can be reserved by customers on the web site of the bus company. The customer has the option to directly pay for the seat through the website. In that case, the seat cannot be cancelled (neither by the customer nor by the bus company). If the customer has not paid for the seat, the bus company can cancel the seat if the customer does not show up one hour before the trip. When the reservation is cancelled, the seat will become free and can be sold to another customer. Both the customer and the company staff must authenticate themselves for performing operations with the system.

So far what we have

- Actors
  - Customer
  - Bus Company Administrator
- Use Cases
  - Reserve Seat
  - Pay for the Seat
  - Cancel Seat
  - Register
  - View Seat

Start Making a Use Case Diagram with the identified information

Actor Goals?

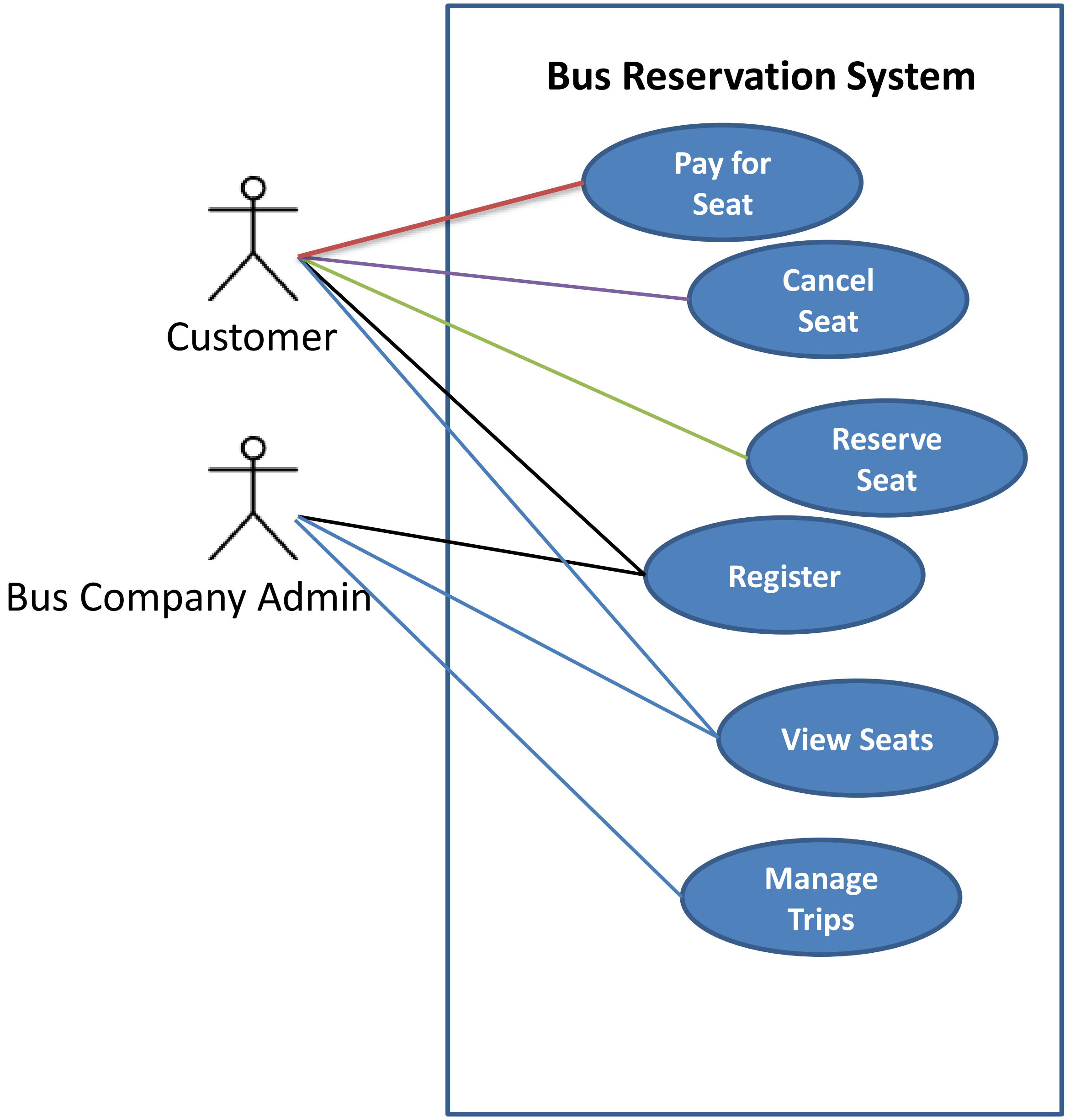
Customer Reserves the Seat
Customer Pays for the Seat
Customer & Admin Cancels the Seat
Customer & Admin must Register
Customer & Admin can View the Seat

Case

Complete the Use Case Diagram

Consider an online reservation system for a bus company. The bus company includes several buses and realizes trips to different cities. Each bus is identified by its plate number and a separately assigned bus number. The trips are based on a predefined schedule and stop at predefined bus stations. Each bus can have only one trip per day. Each bus includes a driver and one hostess. For long trips, the bus will have breaks at service and rest areas. There are two types of trips, normal trips and express trips. Express trips do not stop at intermediate stations and get faster at the destination. Seats can be reserved by customers on the web site of the bus company. The customer has the option to directly pay for the seat through the website. In that case, the seat cannot be cancelled (neither by the customer nor by the bus company). If the customer has not paid for the seat, the bus company can cancel the seat if the customer does not show up one hour before the trip. When the reservation is cancelled, the seat will become free and can be sold to another customer. Both the customer and the company staff must authenticate themselves for performing operations with the system.

One Possible Solution for the Case

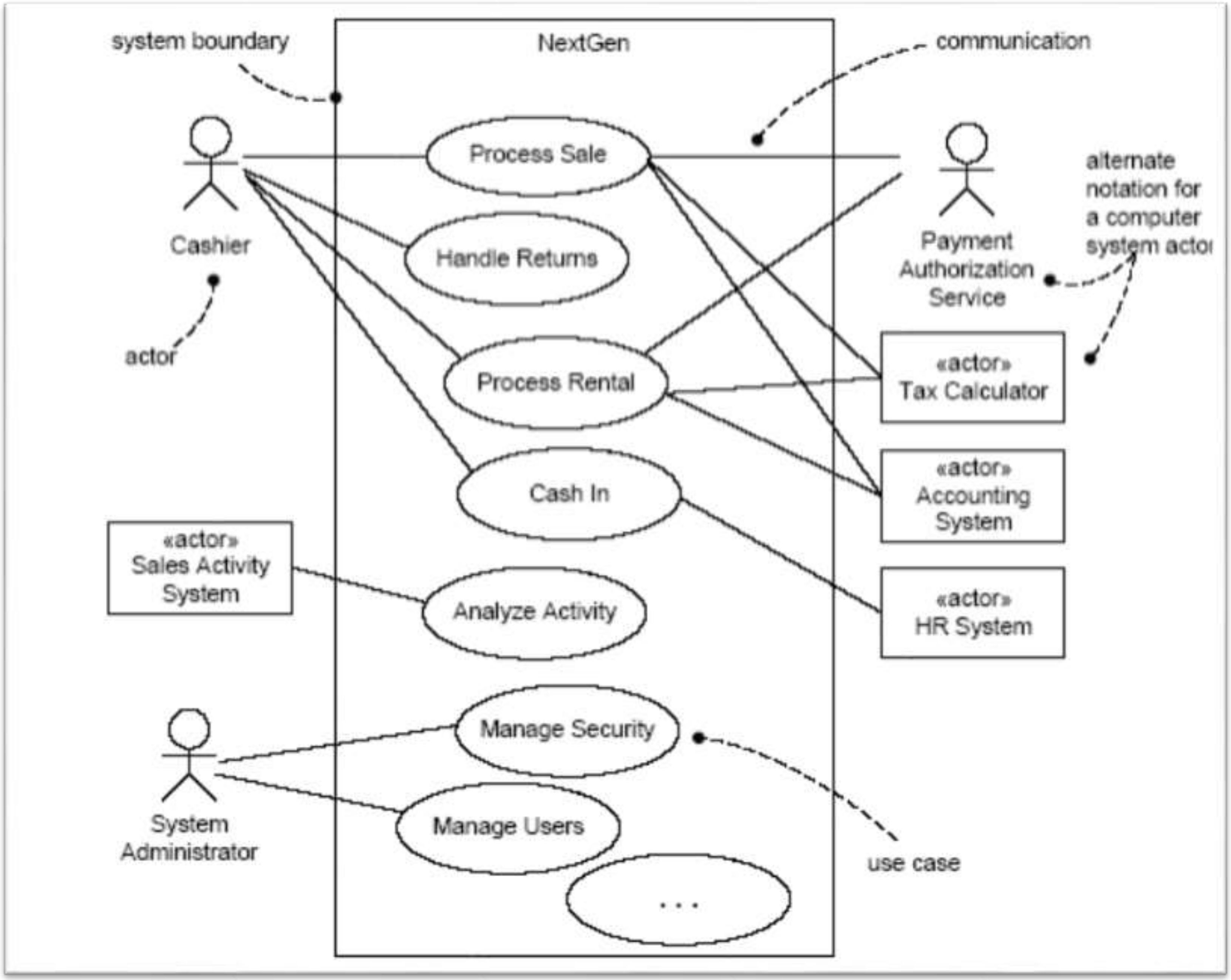


Process Sale:

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.

Actors and User Goals

Actor	Goal	Actor	Goal
Cashier	1. Process sales 2. Process rentals 3. Handle returns 4. Cash in 5. Cash out …	System Administrator	1. Add users 2. Modify users 3. Delete users 4. Manage security 5. Manage system tables
Manager	1. Start up 2. Shut down	Sales Activity System	1. Analyze sales and performance data
…	…	…	…


system boundary
NextGen
communication
Process Sale
Cashier
Payment Authorization Service
alternate notation for a computer system actor
Handle Returns
actor
Process Rental
«actor» Tax Calculator
«actor» Accounting System
Cash In
«actor» Sales Activity System
Analyze Activity
«actor» HR System
Manage Security
System Administrator
Manage Users
use case
. . .