

BEFORE WE START... LET'S TALK ABOUT YOU

IT5004 > Quizzes > Quiz Lecture 1 Survey

[2320] 2023/2024 Semester...

[Home](#)

[Announcements](#)

[Assignments](#)

[Discussions](#)

[Grades](#)

[People](#)

[Pages](#)

[Files](#)

[Syllabus](#)

[Outcomes](#)

[Rubrics](#)

[Quizzes](#)

[Modules](#)

[Collaborations](#)

[New Analytics](#)

Published

Preview

Edit

:

Quiz Lecture 1 Survey

Quiz type Ungraded survey

Points

Shuffle answers No

Time limit No time limit

Multiple attempts No

View responses Always

Show correct answers No

One question at a time No

Anonymous submissions No

Due	For	Available from	Until
-	Everyone	18 Jan at 0:00	25 Jan at 23:59

Preview

LECTURE 1

INTRODUCTION

LEK HSIANG HUI

LEARNING OBJECTIVES

At the end of this lecture, you should understand:

- What an information system is
- The role of (IT) business analyst
- What methodologies, models, tools, and techniques are
- Stages in building information systems



WHAT IS AN INFORMATION SYSTEM?

What is an
information
system?

Role of (IT)
Business
Analyst

Methodologies,
Models , Tools
and Techniques

SDLC

Successful
System
Development



**Systems are built
to make the
business more
competitive**

A group of six diverse business professionals in suits are cheering with their fists raised, set against a white background.

Crucial to the
success of
modern business
organizations

INFORMATION SYSTEMS



TYPES OF INFORMATION SYSTEMS

**DECISION SUPPORT
SYSTEM (DSS)**

**EXECUTIVE
INFORMATION
SYSTEM (EIS)**

**ECONOMIC AND
COMPETITIVE
DATA**

DOCUMENTS

**MANAGEMENT
INFORMATION
SYSTEM (MIS)**

**TRANSACTION
DATA**

**COMMUNICATION
AND OFFICE SUPPORT
SYSTEM**

**TRANSACTION
PROCESSING
SYSTEM (TPS)**



Collection of
inter-related
components

Use to complete
a business task



GROUP DISCUSSION

Suppose we are designing the information system that powers the Grab business

1. Discuss with fellow classmates seated around you and come up with a list of the key IT components that powers the Grab Information System
2. Post your list onto Canvas discussion forum

ROLE OF (IT) BUSINESS ANALYST

What is an
information
system?

Role of (IT)
Business
Analyst

Methodologies,
Models , Tools
and Techniques

SDLC

Successful
System
Development



Your Job (System
Analyst/IT Business
Analyst)

=

Solve Problems



problem?

TYPES OF PROBLEMS

Company wants
to sell their
products online

What are the category
of products?

How to take payment
online?

How to account for
inventory stock level?

TYPES OF PROBLEMS

Production
needs to plan
how much to
produce each
week

How to estimate
demand and suggest a
production plan?

What sorts of
forecasting mechanism
will be helpful?

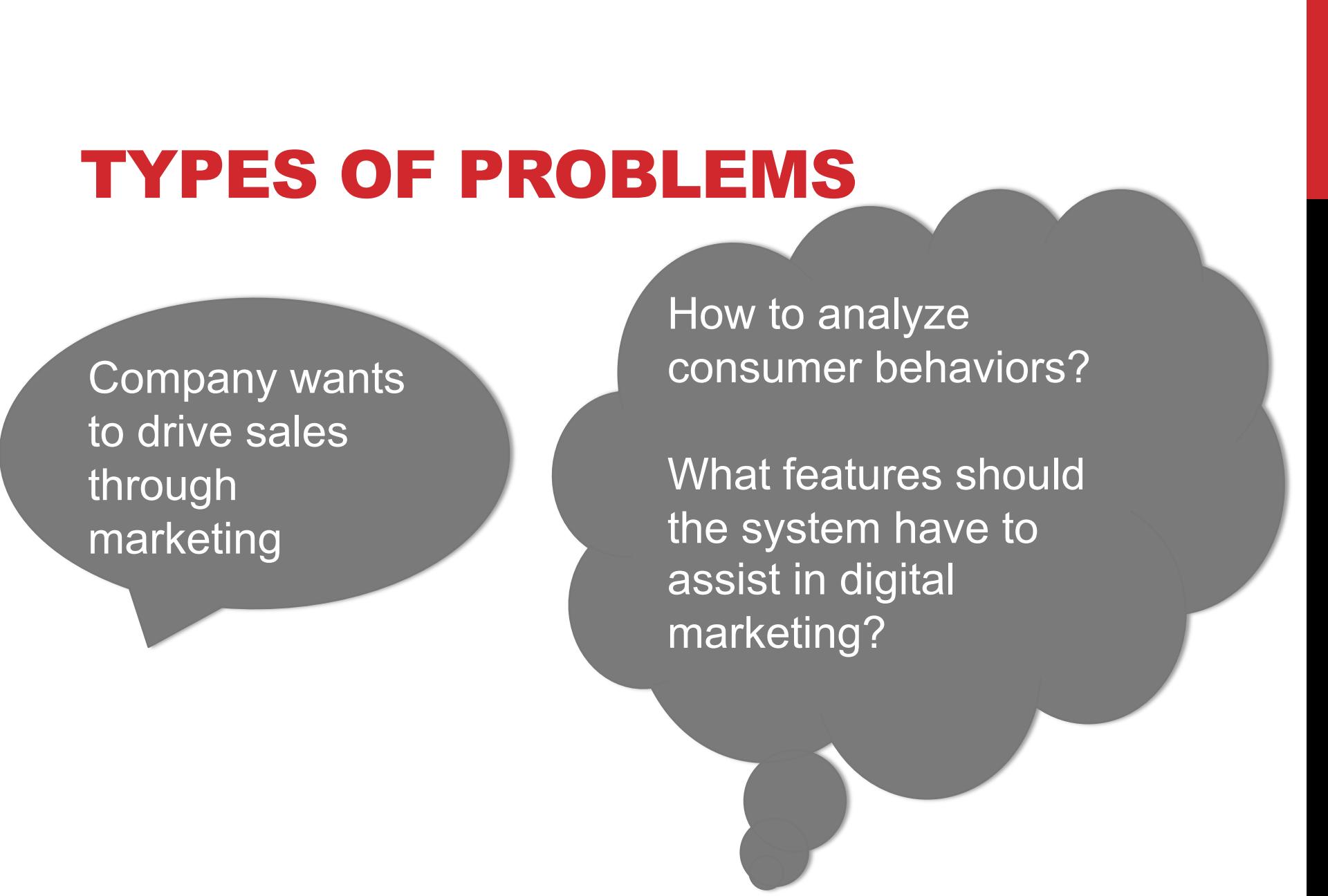
TYPES OF PROBLEMS

Company wants
to reduce
operating cost
and optimize
productivity

What IT solution can
be used to automate
processes?

How can we measure
cost/productivity?

TYPES OF PROBLEMS



Company wants
to drive sales
through
marketing

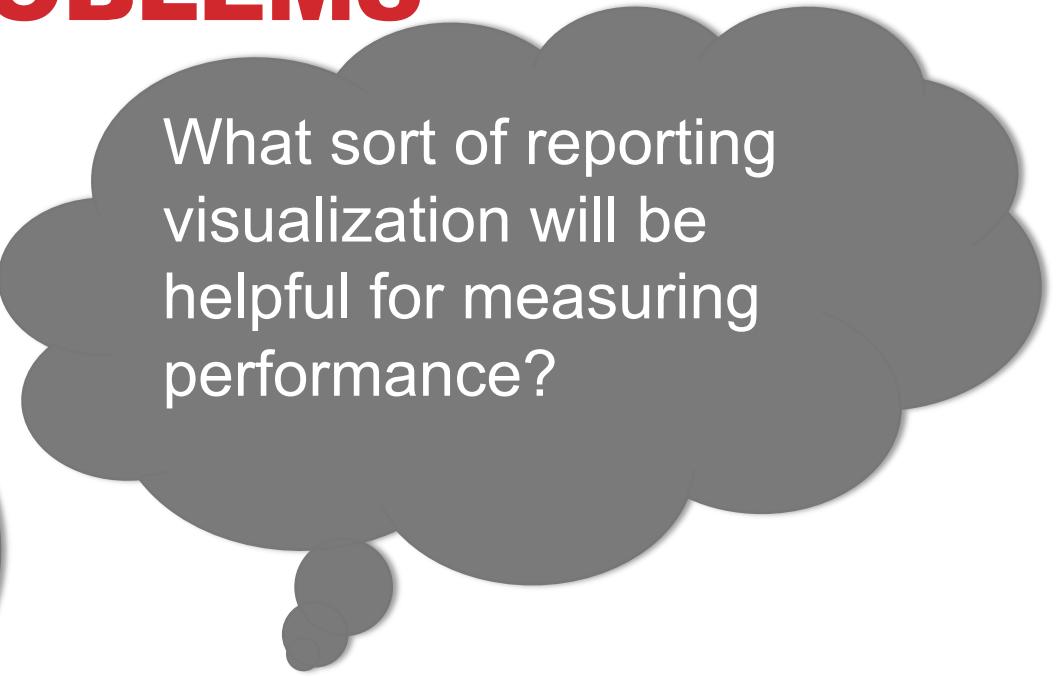
How to analyze
consumer behaviors?

What features should
the system have to
assist in digital
marketing?

TYPES OF PROBLEMS



Management wants to know the financial status of the company

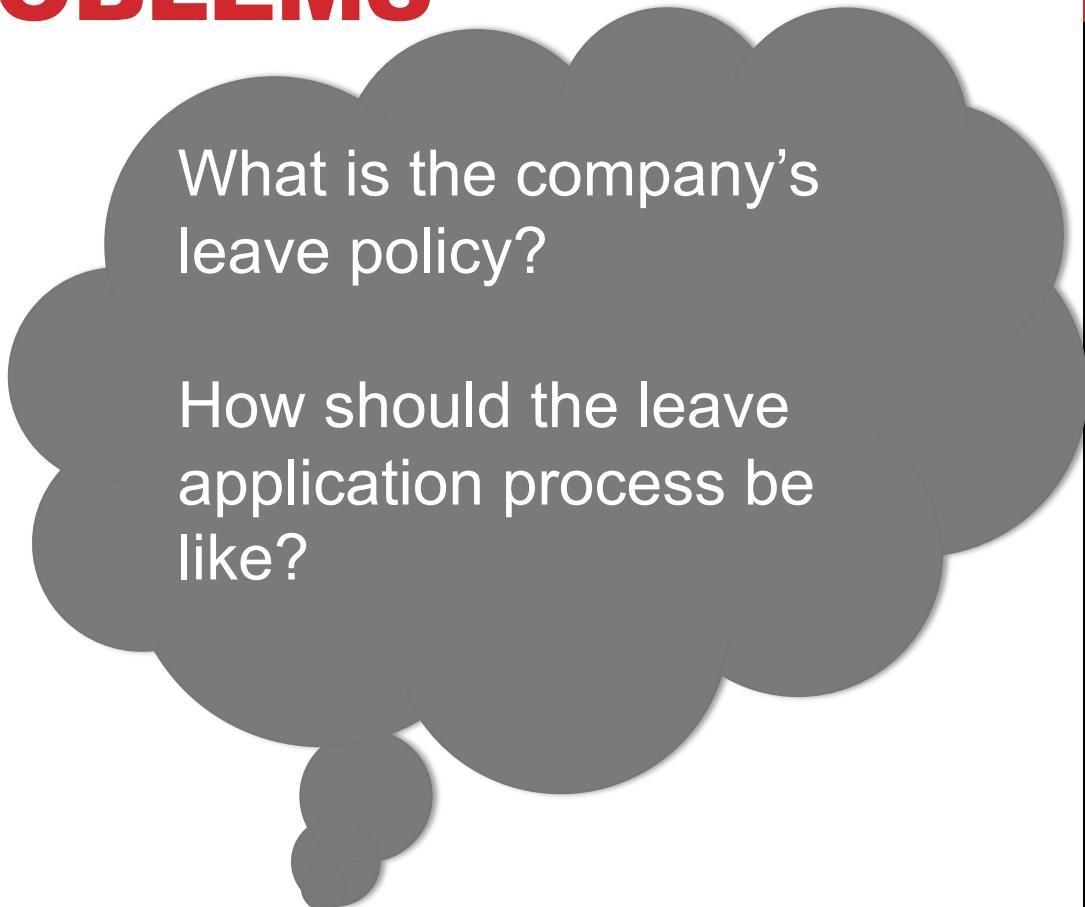


What sort of reporting visualization will be helpful for measuring performance?

TYPES OF PROBLEMS



HR needs to process employee leave/MC/etc requests



What is the company's leave policy?

How should the leave application process be like?

ROLE OF (IT) BUSINESS ANALYST

Perform Business Analysis

Identify the
business
needs

Develop
/Implement
solutions



IT BA

Works in the context of IT projects

Requirement analysis and designing systems

Liaises with business and technical stakeholders



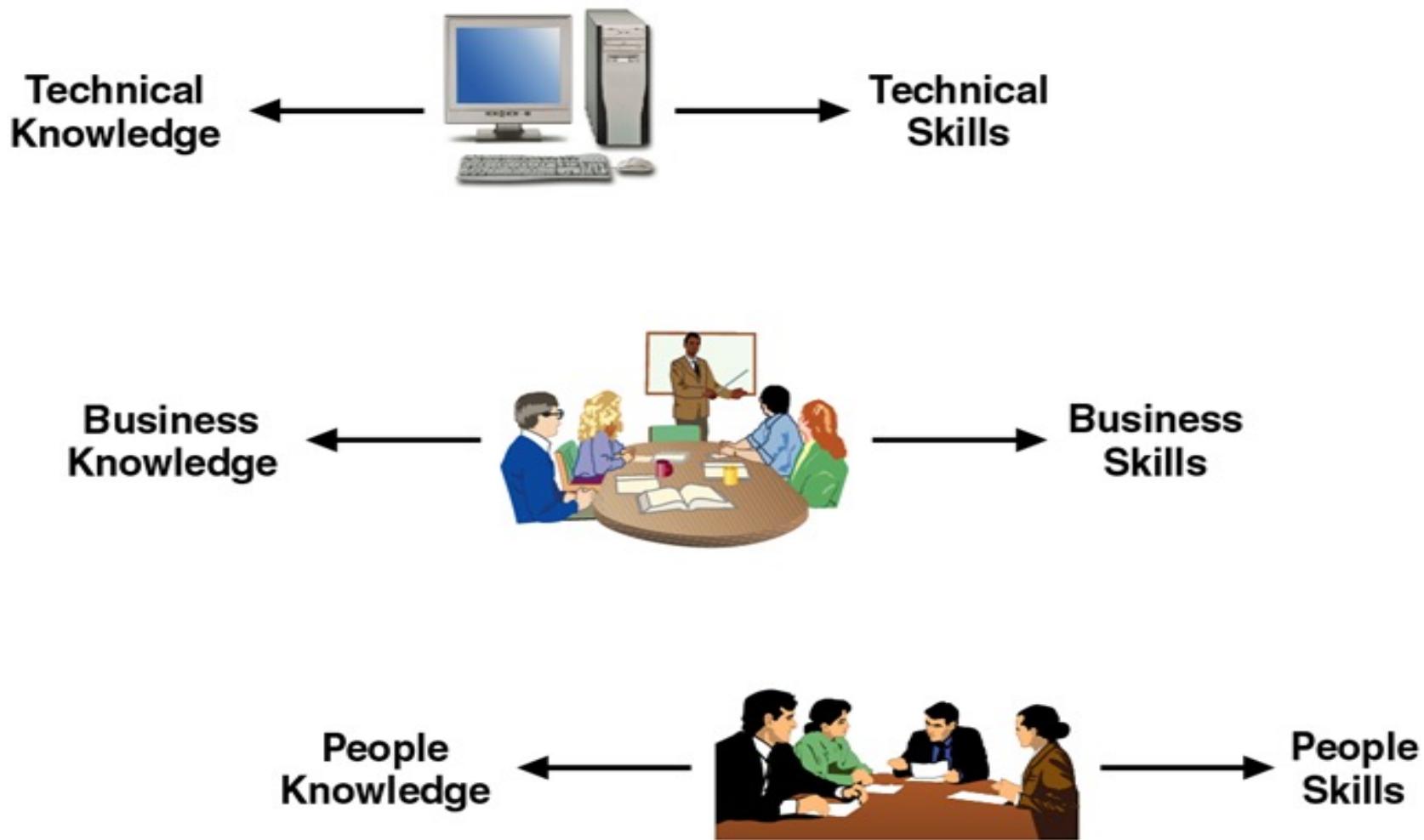
STAKEHOLDERS

Person or organization affected by a project

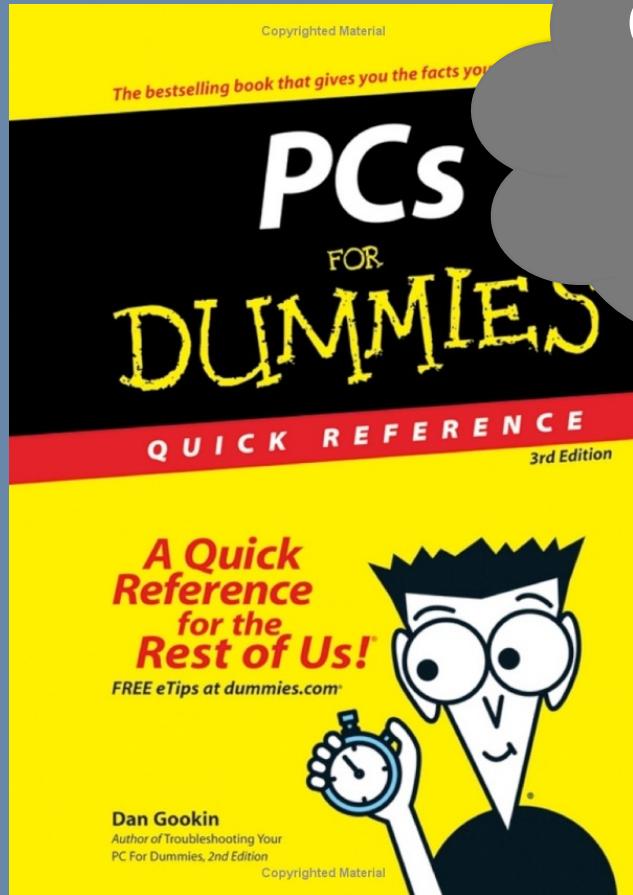
Can be users, customers
(who do not use the system directly)



KNOWLEDGE AND SKILLS REQUIRED OF AN IT BUSINESS ANALYST



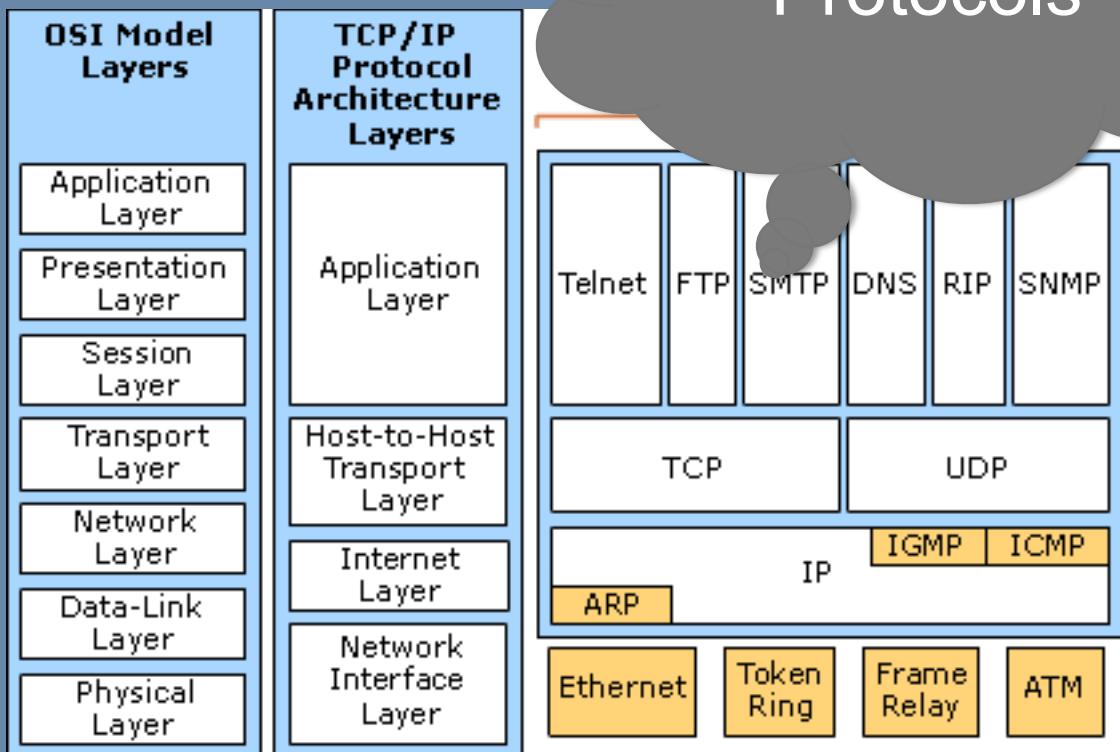
TECHNICAL KNOWLEDGE AND SKILLS



Computers and
How it works
-DUH?!



TECHNICAL KNOWLEDGE AND SKILLS



Networks and
Protocols



TECHNICAL KNOWLEDGE AND SKILLS

Database and
database
management
system

Y U NO TAKE



DATABASE COURSE?



TECHNICAL KNOWLEDGE AND SKILLS

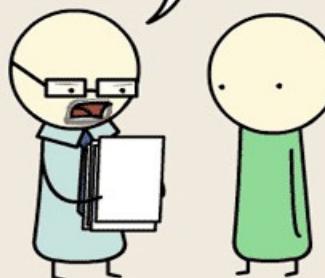
PYTHON

JAVA

C++

UNIX SHELL

THIS IS PLAGIARISM.
YOU CAN'T JUST "IMPORT ESSAY."



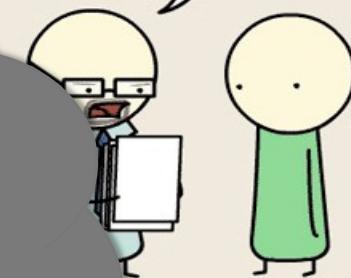
I'M TWO PAGES IN AND I STILL
HAVE NO IDEA WHAT YOU'RE SAYING.



I ASKED FOR ONE COPY,
NOT FOUR HUNDRED.

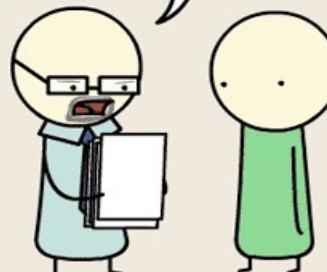


I DON'T HAVE PERMISSION TO
READ THIS.



ASSEMBLY

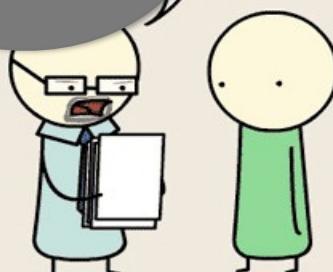
DID YOU REALLY HAVE TO REDEFINE EVERY
WORD IN THE ENGLISH LANGUAGE?



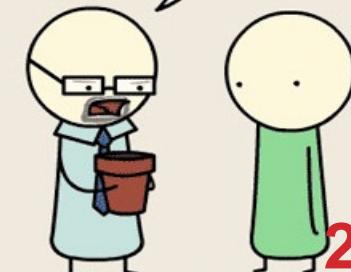
A NULL TERM



END OF THE LINE



THIS IS A FLOWER POT.



TECHNICAL KNOWLEDGE AND SKILLS

Operating
Systems and
Utilities



BUSINESS KNOWLEDGE AND SKILLS

What are the business functions the organization perform?

How is the organization structured?

How is it managed?

BUSINESS KNOWLEDGE AND SKILLS

CORE VALUES

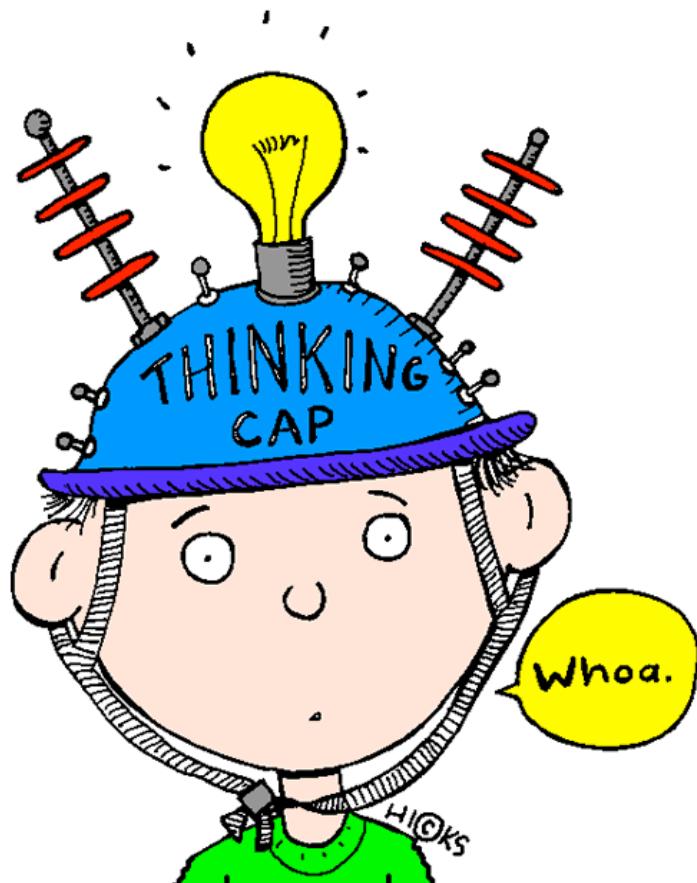
golden rule
vision
customers
corporate
conduct
purpose
culture
marketing
honesty
organization
guiding
excellence
innovation
industry
value
charter
teamwork
ethics
goals
mission
accountability
customer service
firm
principles
employees
integrity
live by
values
code
ideology
business
one company
example
rules

- Need to understand:
 - the organization
 - its culture
 - its mission
 - its objectives

PEOPLE KNOWLEDGE AND SKILLS



PEOPLE KNOWLEDGE AND SKILLS



Need to understand
how people think

PEOPLE KNOWLEDGE AND SKILLS

LEARN

Need to
understand how
people learn

PEOPLE KNOWLEDGE AND SKILLS

Need to understand how
people react to change



PEOPLE KNOWLEDGE AND SKILLS

A photograph of three business professionals in a meeting. A man in a white shirt is on the left, a man in a dark suit is in the center, and a woman in a brown pinstripe suit is on the right. They are seated around a table with a laptop and papers. A large, semi-transparent gray speech bubble is positioned over the center of the group, containing the text "Communicate: Interviewing Techniques".

Communicate:
Interviewing
Techniques

METHODOLOGIES, MODELS, TOOLS AND TECHNIQUES

What is an
information
system?

Role of (IT)
Business
Analyst

Methodologies,
Models , Tools
and Techniques

SDLC

Successful
System
Development

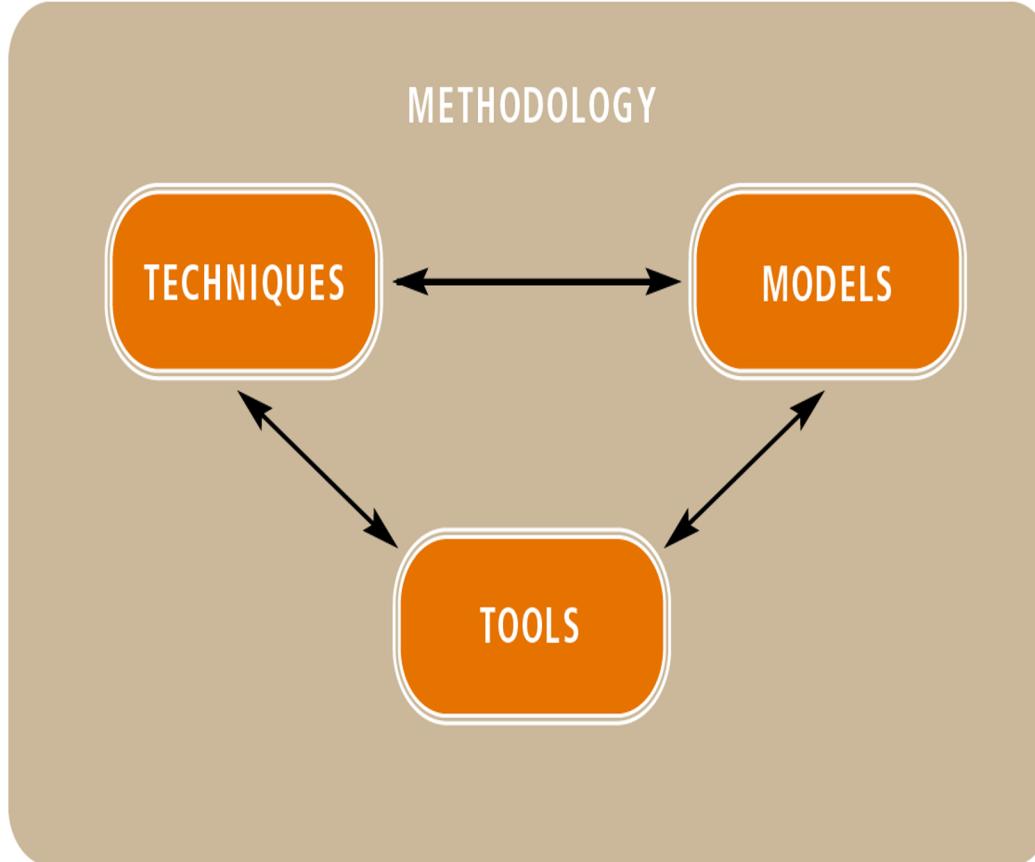
METHODOLOGY



System Development Methodology:

- System development is not just coding only
- Need to gather the requirements, design the system, coding, testing, etc
- Set of guidelines to follow for completing every activity in the development process

METHODOLOGY

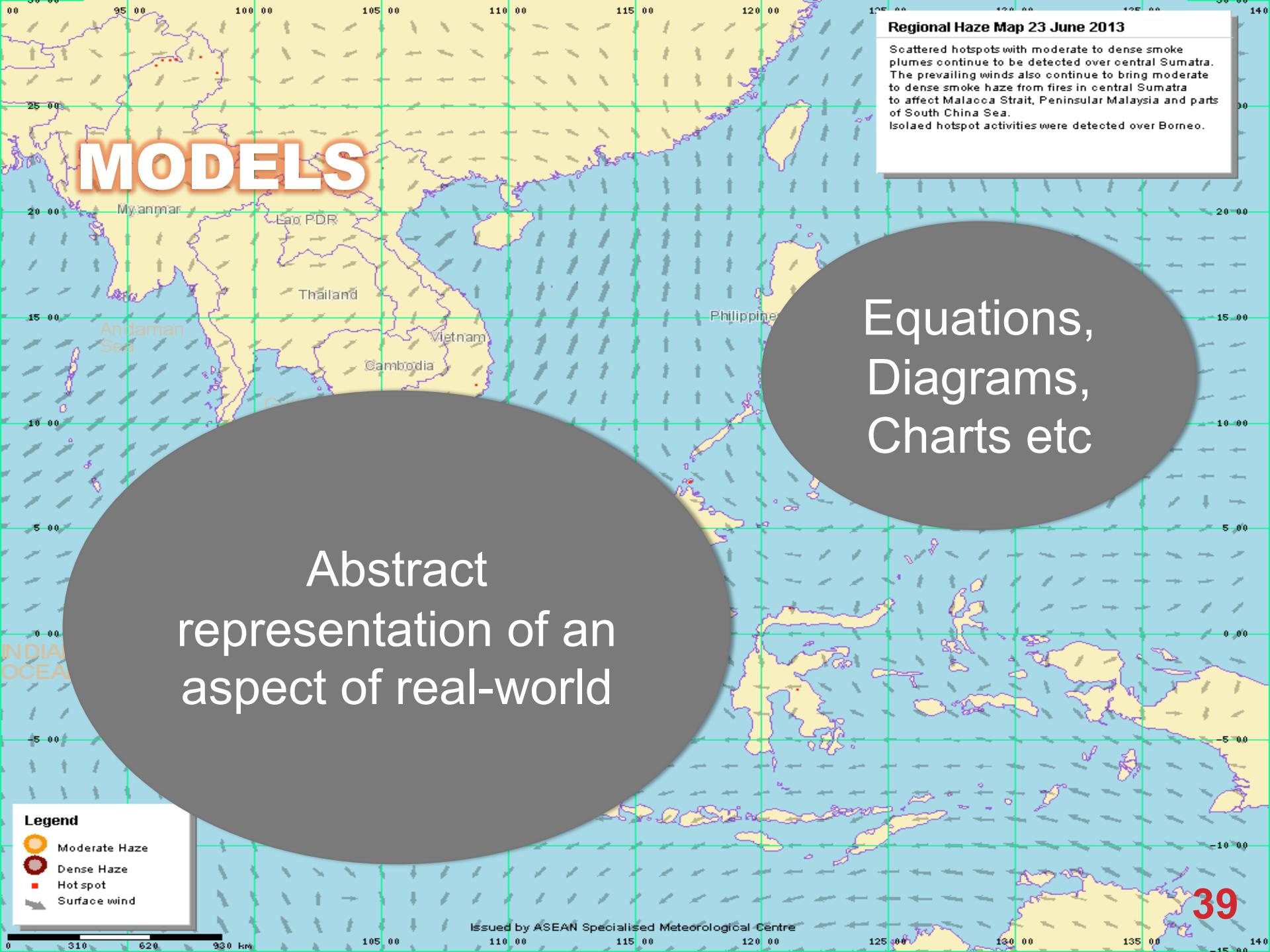


Consists of
Models,
Tools &
Techniques

MODELS

Abstract
representation of an
aspect of real-world

Equations,
Diagrams,
Charts etc



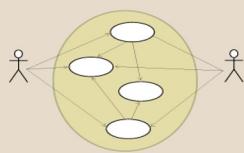
MODELS USE IN SYSTEM DEVELOPMENT



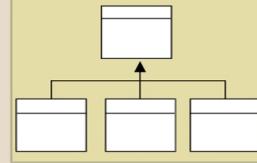
Unified Modeling Language (UML)

- Standard set of model constructs
- Designed for Object-Oriented development

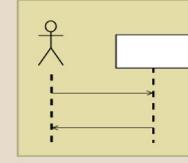
UML DIAGRAMS USED FOR MODELING



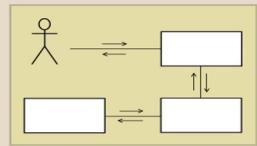
Use case
diagram



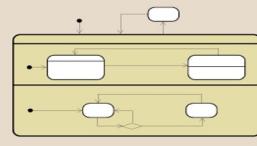
Class
diagram



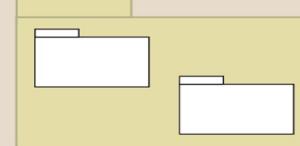
Sequence
diagram



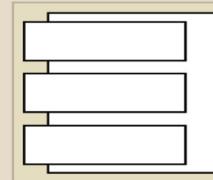
Communication
diagram



Statechart
diagram



Package
diagram



Deployment
diagram

TOOLS



Integrated
Development
Environment (IDE)



Microsoft®
Visual Studio®



TOOLS

Computer Aided System
Engineering (CASE)

Rational® software

- Store information about system specification
- Aid in various aspects of system development
- Modeling, code generator, documentation tools

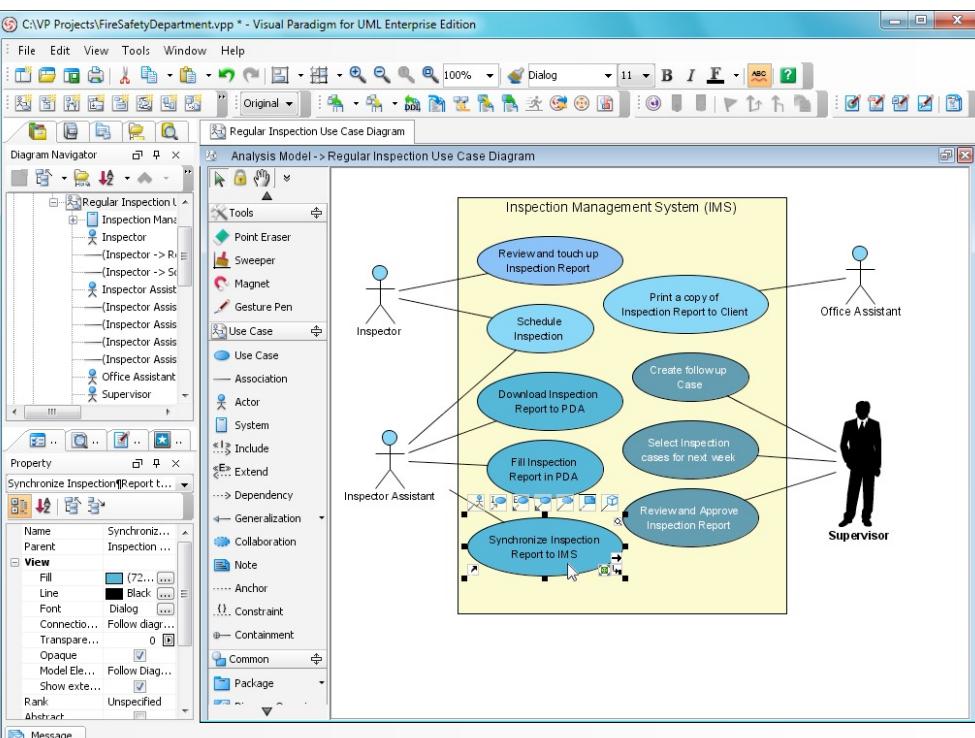


UML Drawing Tool

TOOLS

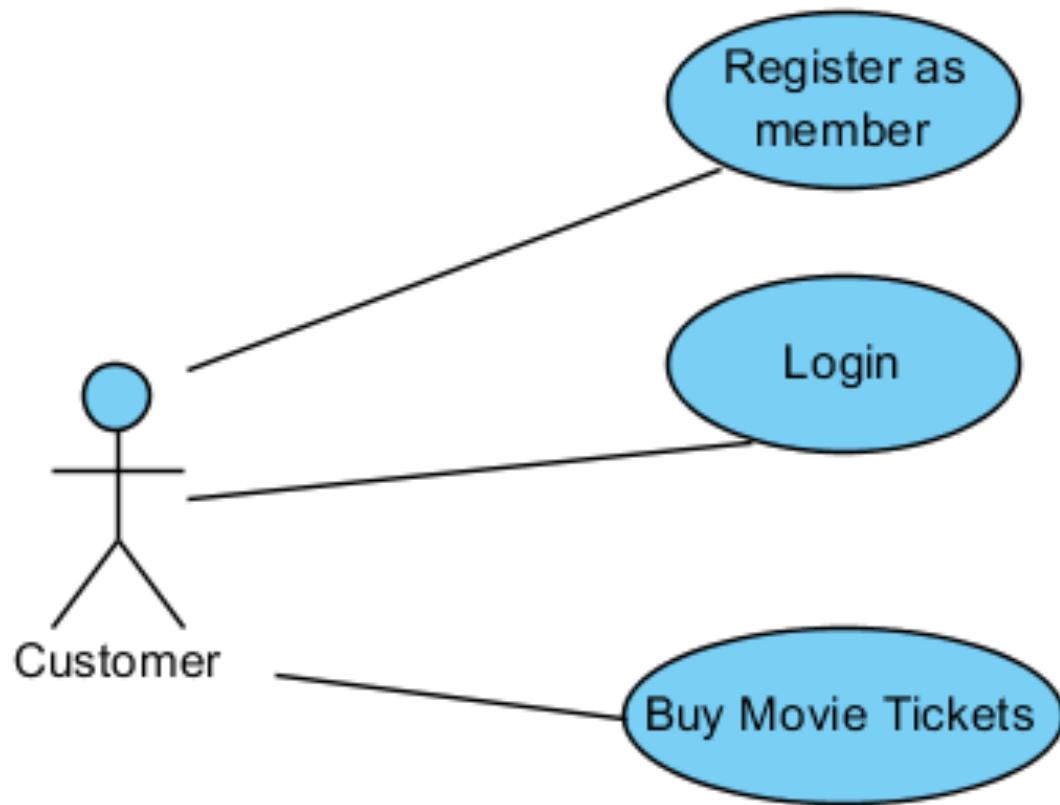


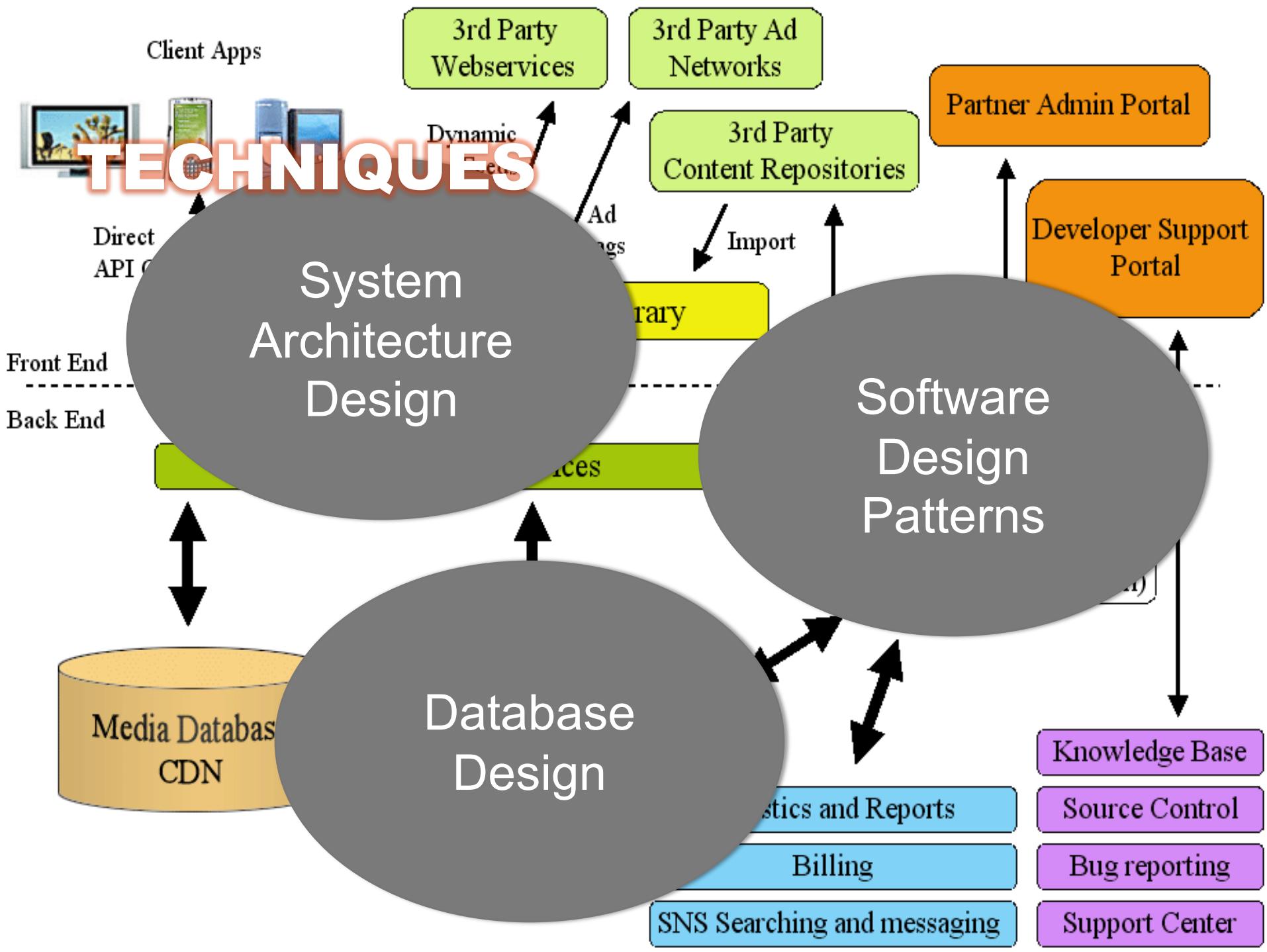
Visual Paradigm
for UML



TECHNIQUES

Requirements
Modeling
Techniques*





TECHNIQUES

Project Planning Techniques

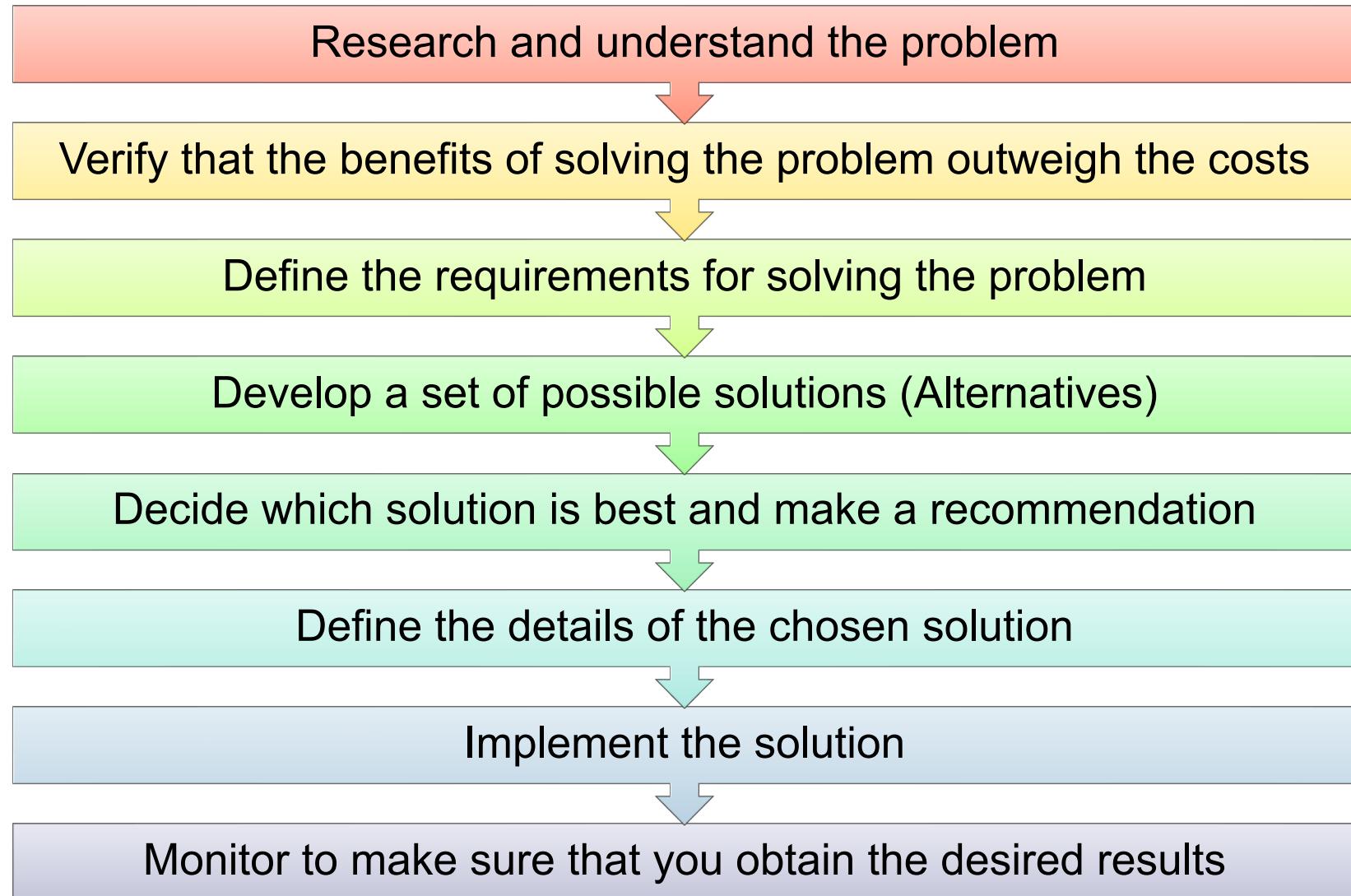
Cost/Benefits Analysis

TECHNIQUES

- Set of guidelines for completing an activity
- Examples:
 - Domain-modeling, use case modeling, software-testing, etc



A GENERAL PROBLEM-SOLVING APPROACH



SYSTEMS DEVELOPMENT LIFE CYCLE

What is an
information
system?

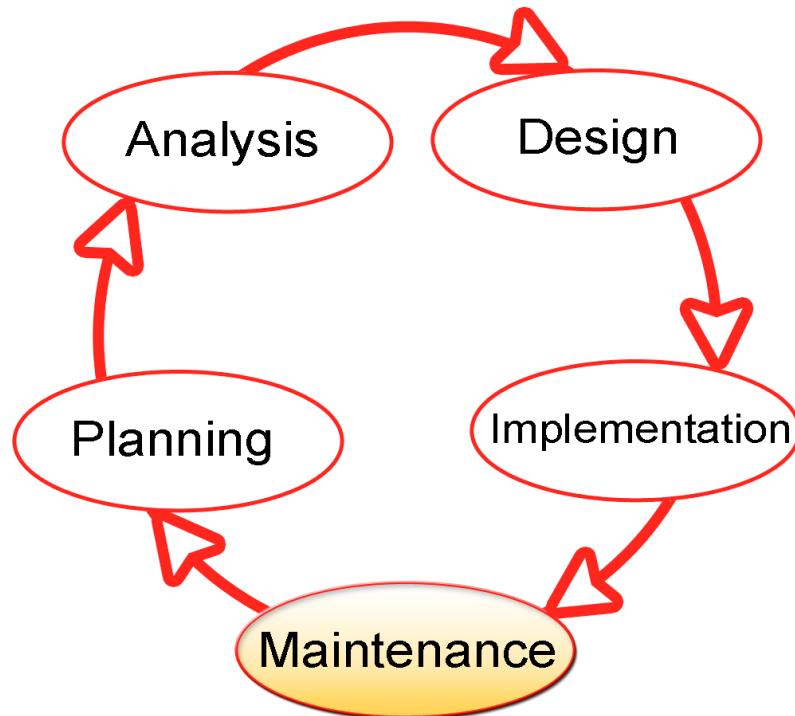
Role of (IT)
Business
Analyst

Methodologies,
Models , Tools
and Techniques

SDLC

Successful
System
Development

SYSTEMS DEVELOPMENT LIFE CYCLE



Entire process of developing a system

Not just about programming!

PROJECT PLANNING PHASE



- Ensure that the project is feasible
- Identify scope of new system
- Develop schedule, resource plan, budget etc

ANALYSIS PHASE



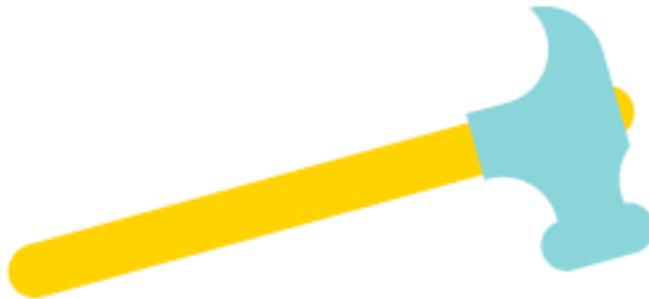
- Understand the document in detail
- Process requirements of the new system

DESIGN PHASE



- Design system based on the requirements

IMPLEMENTATION PHASE



Implementation

- Build system
(includes acquire components)
- Testing

MAINTENANCE PHASE



- Ensure the system is running properly during the lifetime of the system
- Fix any issue that arises

SUCCESSFUL SYSTEM DEVELOPMENT

What is an
information
system?

Role of (IT)
Business
Analyst

Methodologies,
Models , Tools
and Techniques

SDLC

Successful
System
Development

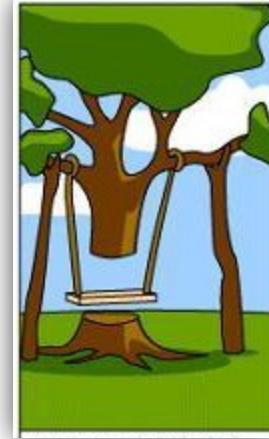
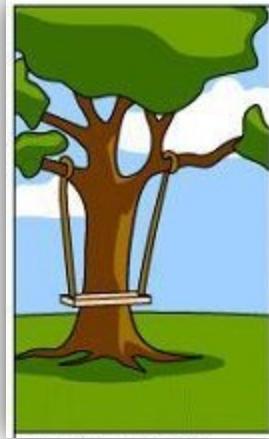
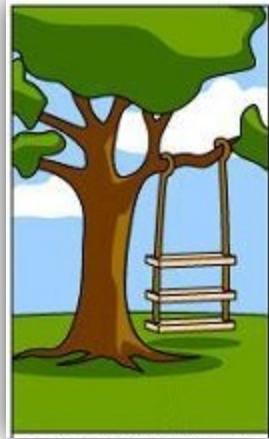
SUCCESSFUL SYSTEM DEVELOPMENT



S U C C E S S

Because you too can own this face of pure accomplishment

THE PROBLEM WITH COMMUNICATION



How the Customer explained it.

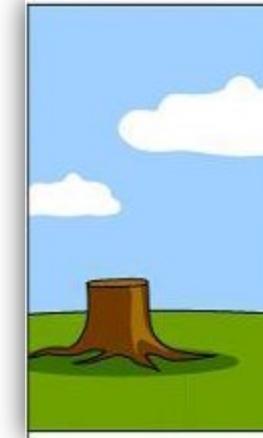
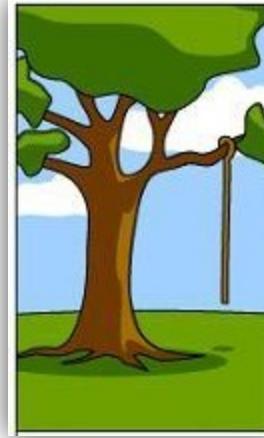
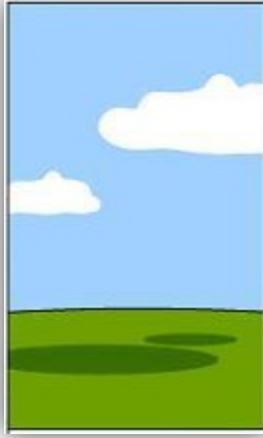
How the Project Leader understood it.

How the Designer designed it.

How the Programmer wrote it.

How the IT Business Analyst described it.

THE FINAL OUTCOME



How the Project was documented.

What Operations were installed.

How the Customer was billed.

How it was supported.

What the Customer really needed!

SUMMARY

Introduction to Information Systems

Role of IT Business Analyst

Some definitions: methodologies, models, tools and techniques

Various activities in system development: business modeling, requirements gathering and analysis, design, implementation, testing, deployment

WHAT'S NEXT?

Systems Development Life Cycle (SDLC)

Planning Phase

LECTURE 2

SYSTEMS

DEVELOPMENT LIFE

CYCLE (SDLC) &

PLANNING PHASE

LEK HSIANG HUI

LEARNING OBJECTIVES

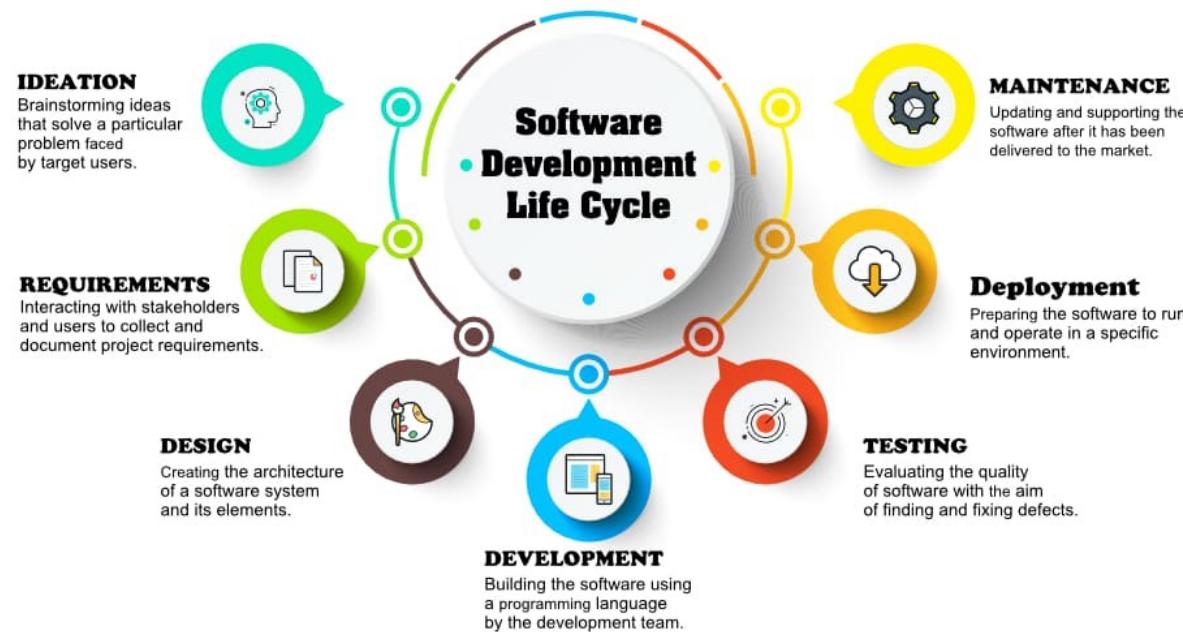
At the end of this lecture, you should understand:

- What predictive SDLC and adaptive SDLC are
- The activities during the planning phase
- What Business Requirement Document (BRD) is
- How to draw activity diagrams

SYSTEMS DEVELOPMENT LIFE CYCLE (SDLC)

Sometimes also known as Software Development Life Cycle (SDLC)

A structured and systematic process for developing system



QUESTION TIME



*“It’s about coming out with the software right? Why bother talking about this? Shouldn’t we just go straight into coding? Shouldn’t we be *agile*?”*

1. What do you think?

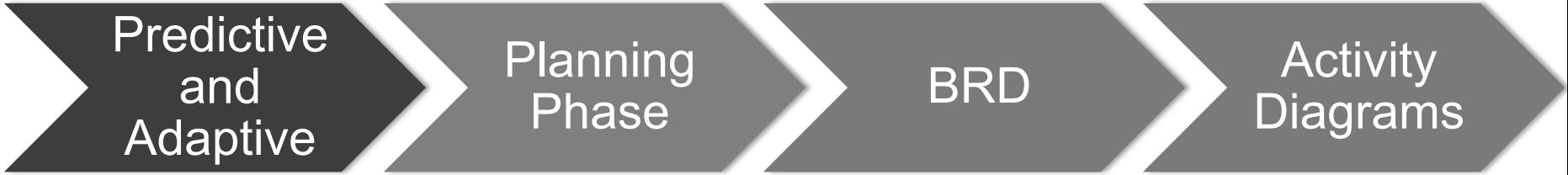
SYSTEMS DEVELOPMENT LIFE CYCLE



Can be generalized
into 2 different
approaches:

1. Predictive SDLC
2. Adaptive SDLC

PREDICTIVE AND ADAPTIVE SDLC



Predictive
and
Adaptive

Planning
Phase

BRD

Activity
Diagrams

PREDICTIVE AND ADAPTIVE SDLC

THE APPROPRIATE SDLC VARIES DEPENDING ON THE PROJECT

PREDICTIVE
SDLC

REQUIREMENTS WELL
UNDERSTOOD AND WELL DEFINED.
LOW TECHNICAL RISK.

ADAPTIVE
SDLC

REQUIREMENTS AND NEEDS
UNCERTAIN.
HIGH TECHNICAL RISK.

SDLC IN PRACTICE

More traditional

New approaches evolved (together with object-oriented approaches)

Predictive
SDLC

Adaptive
SDLC

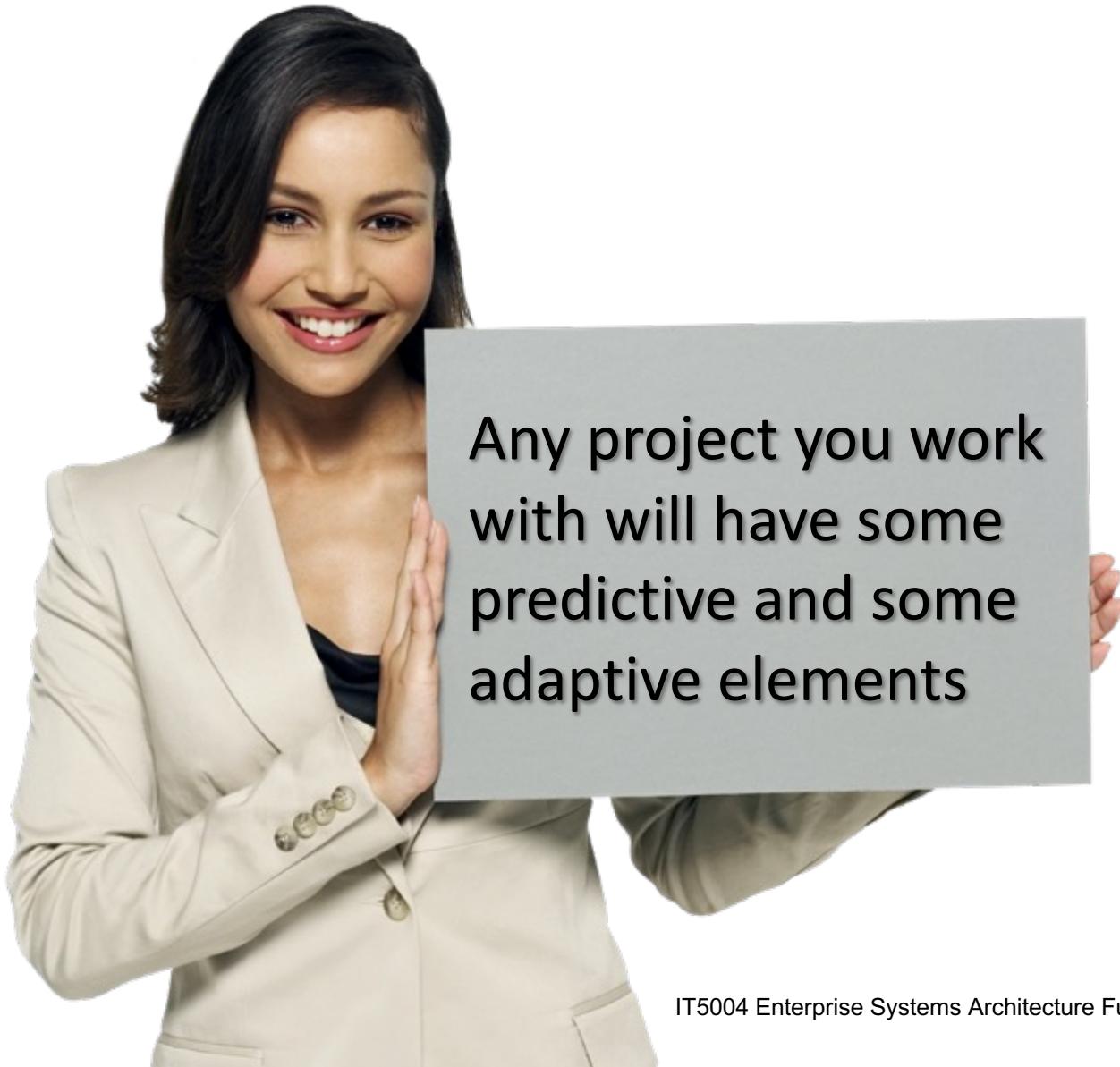
1970s

1980s

1990s

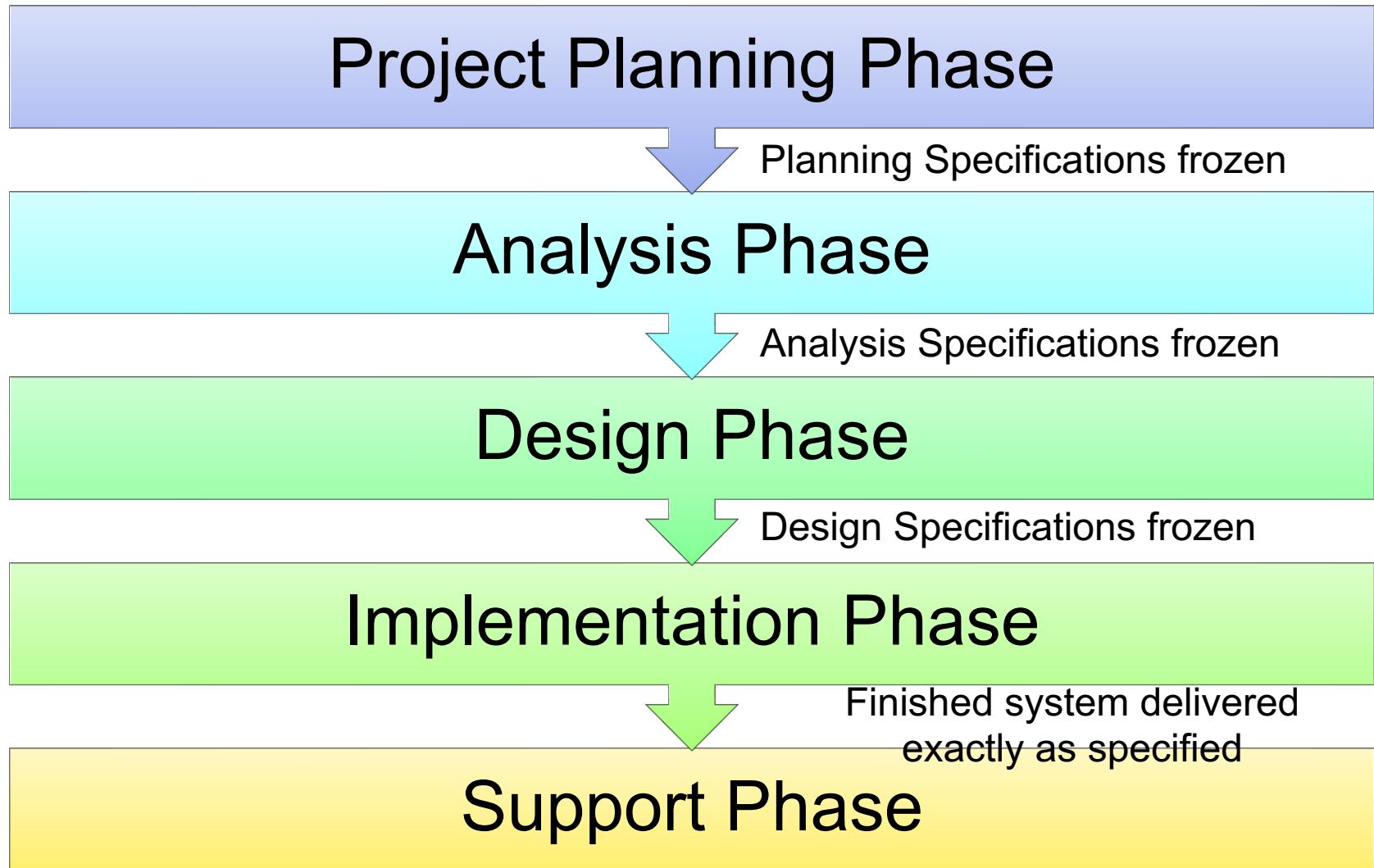
21st Century

SDLC IN PRACTICE



PREDICTIVE (TRADITIONAL) SDLC

PREDICTIVE SDLC APPROACH



QUESTION TIME



Predictive SDLC is also commonly known as the **waterflow methodology**

Why do you think it is called **waterflow methodology?**

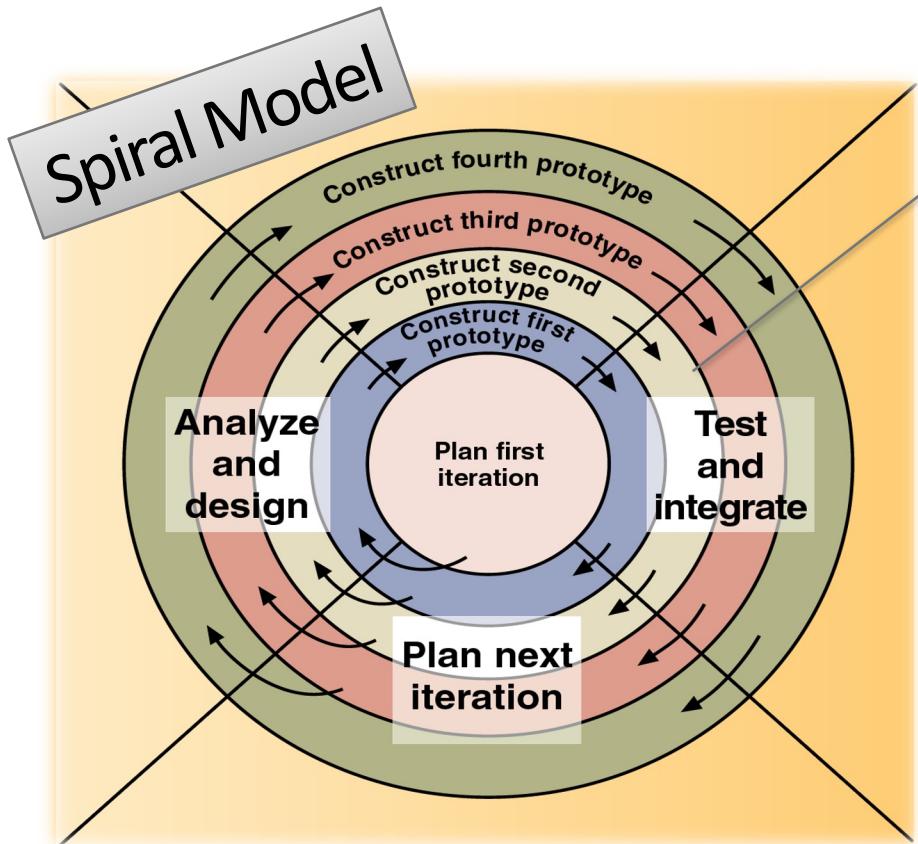


ADAPTIVE SDLC

ADAPTIVE SDLC APPROACH



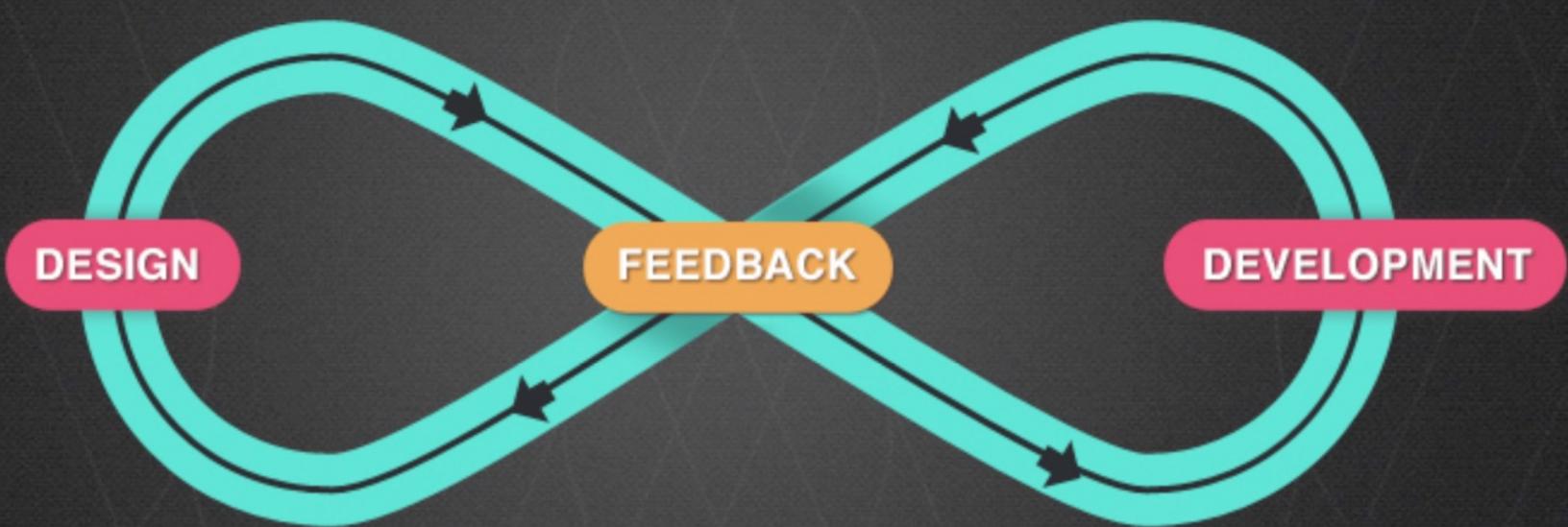
IDEA OF ADAPTIVE SDLC



Contains many activities

- Start from the center and work its way outwards
- Working over and over (until the project is finished)
- Iterative approach

ADAPTIVE SDLC

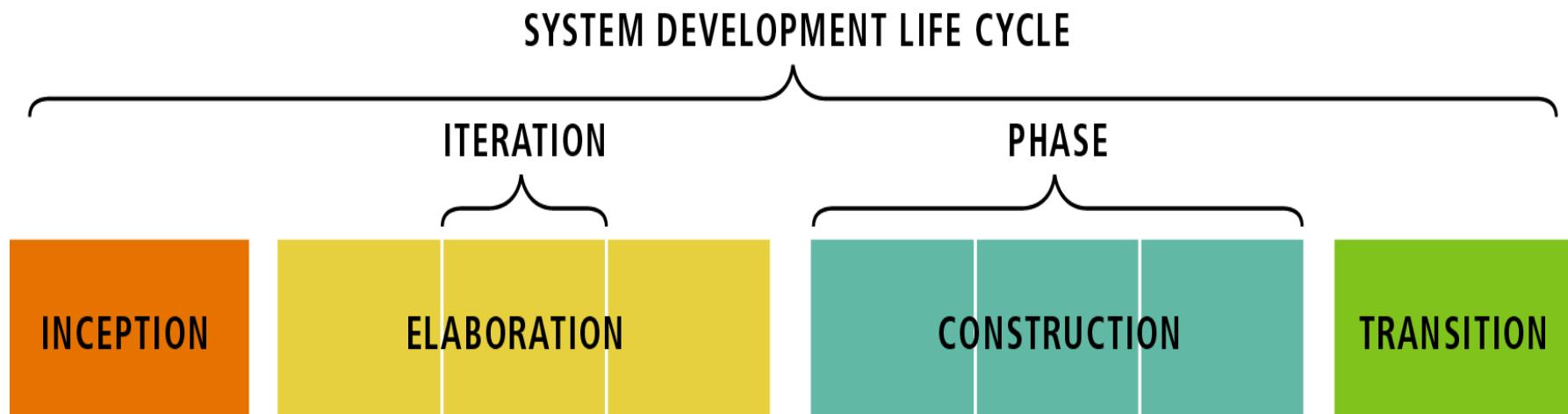


THE UNIFIED PROCESS LIFE CYCLE

AN EXAMPLE OF ADAPTIVE SDLC

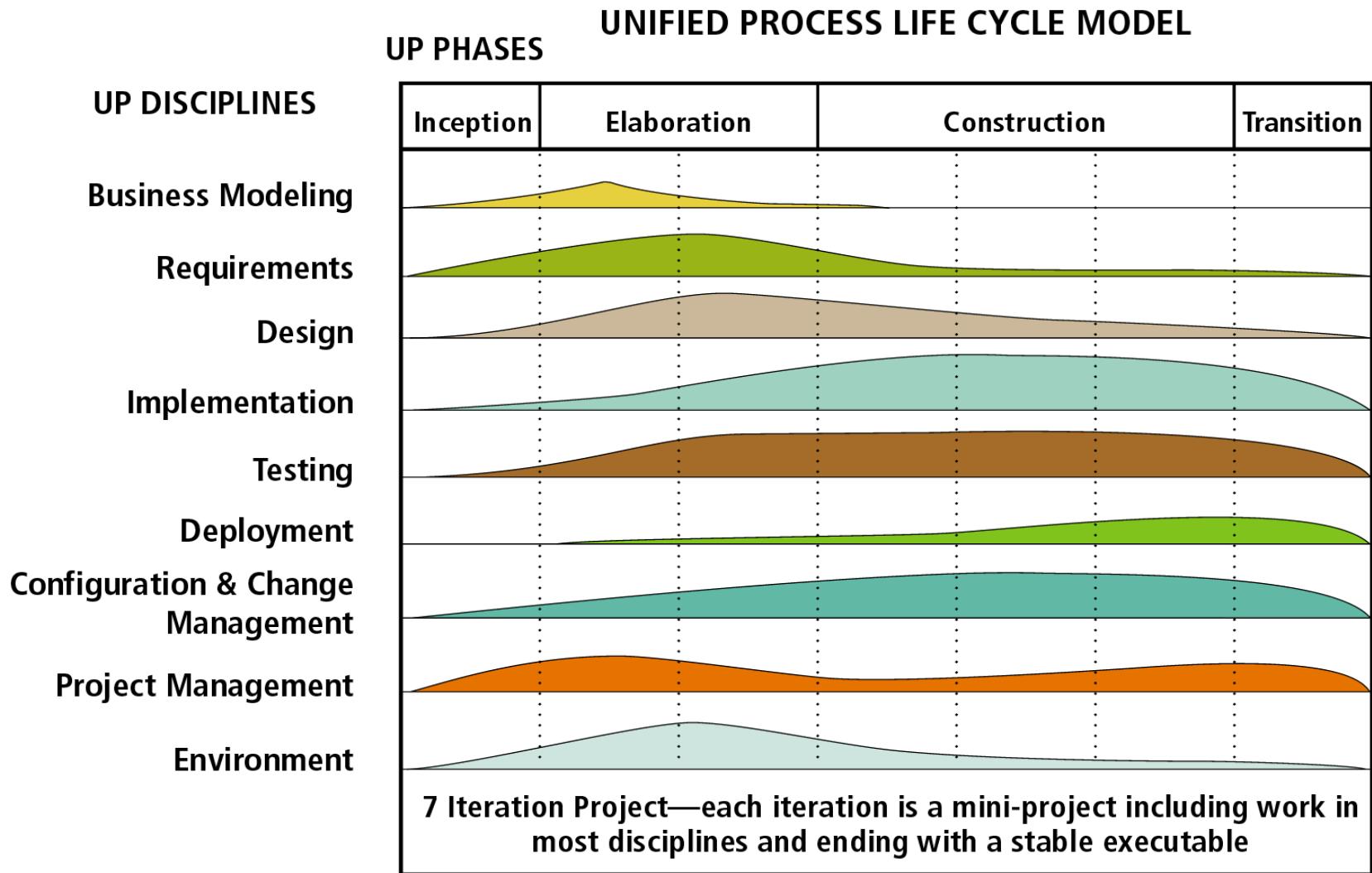
THE UNIFIED PROCESS LIFE CYCLE

Four phases: Inception, Elaboration, Construction, and Transition



PHASES ARE NOT ANALYSIS, DESIGN, AND IMPLEMENT;
INSTEAD, EACH ITERATION INVOLVES A COMPLETE
CYCLE OF REQUIREMENTS, DESIGN, IMPLEMENTATION, AND TEST DISCIPLINES

UP PHASES AND OBJECTIVES



PLANNING PHASE



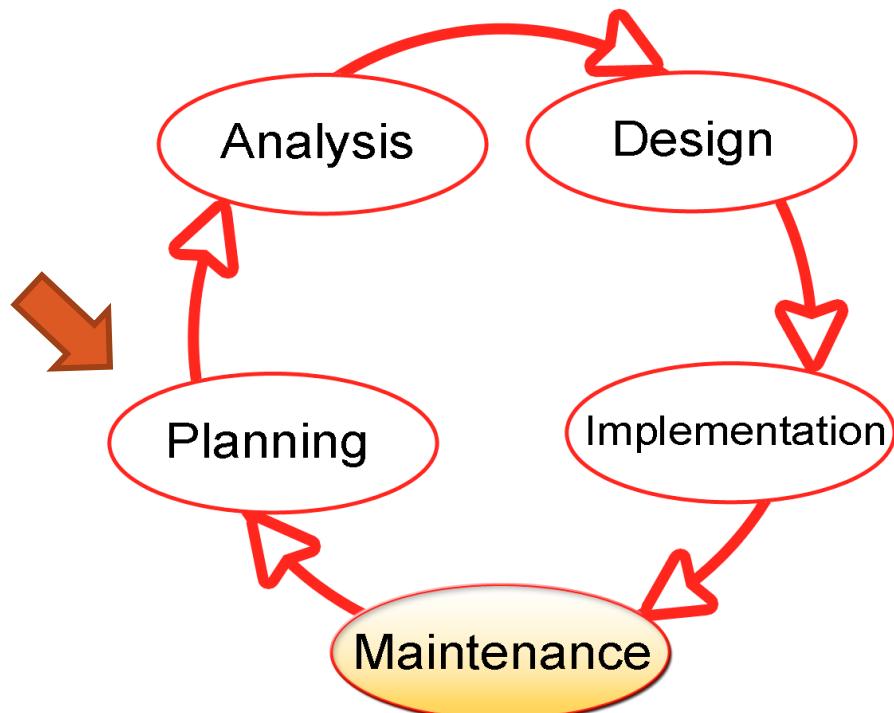
Predictive
and
Adaptive

Planning
Phase

BRD

Activity
Diagrams

PLANNING PHASE



SDLC starts
with
Planning

Involve first
understanding
the business

UNDERSTANDING THE BUSINESS

Need to understand the business environment and potential improvements

By talking to end users, manager, etc

UNDERSTANDING THE BUSINESS

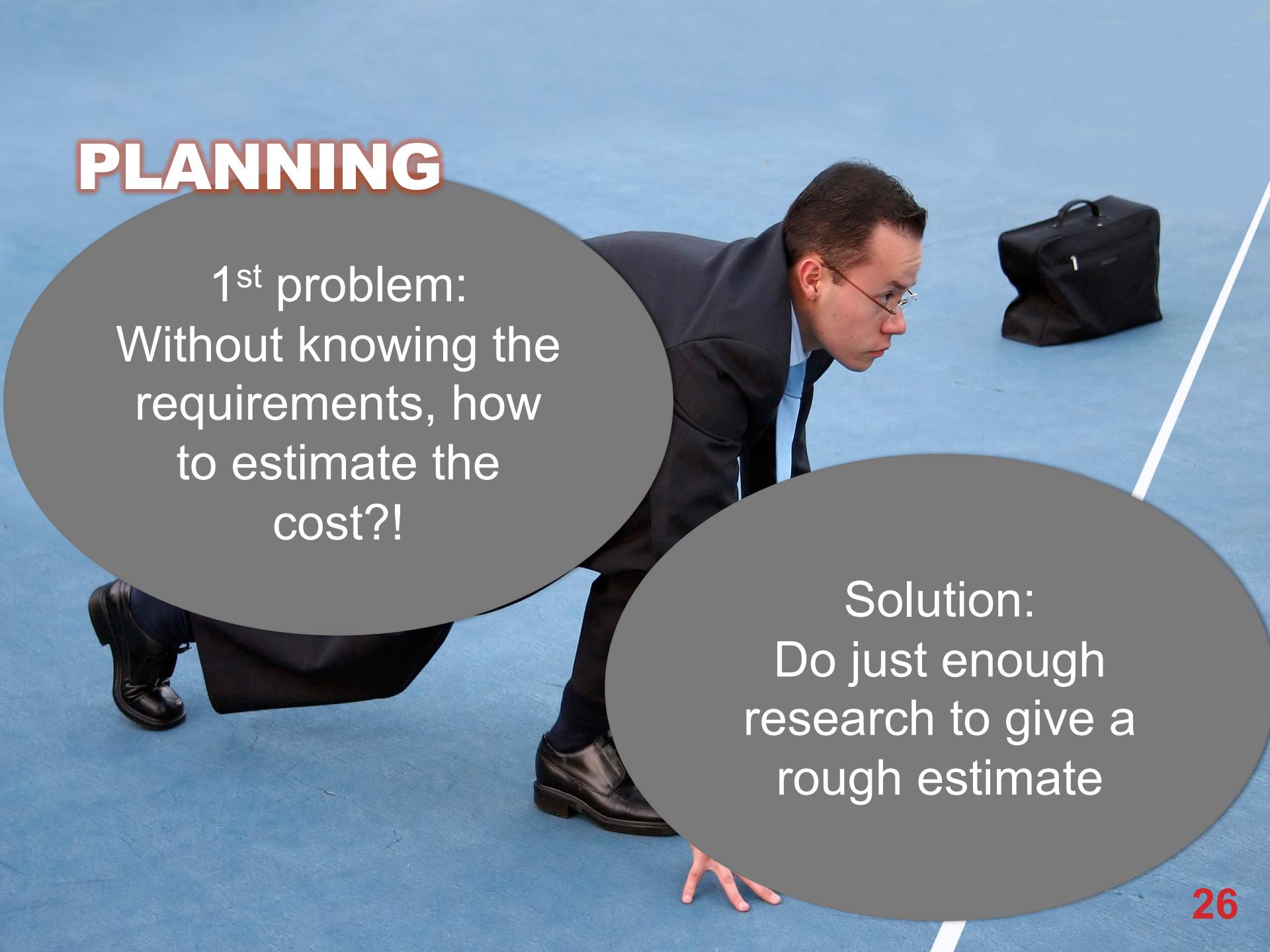
Would need to
document the
business vision, the
business model,
business problems,
etc

PLANNING

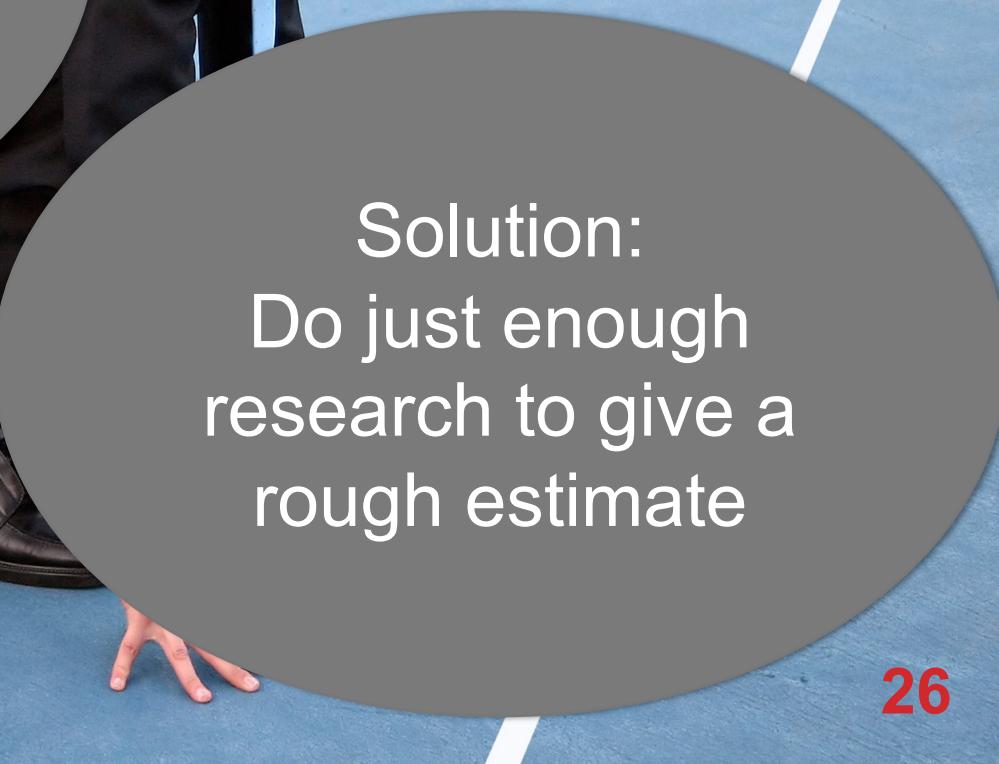
Objectives:

- Develop the business case of the project
- Establish project/product scope
- Explore solutions (Preliminary)

PLANNING

A man in a dark suit, white shirt, and tie is crouching on a blue running track. He is wearing glasses and has his hands on the ground in a starting position. In the background, there is a black bag on the track. A large grey oval shape covers the left side of the image, containing the text.

1st problem:
Without knowing the
requirements, how
to estimate the
cost?!



Solution:
Do just enough
research to give a
rough estimate

PLANNING



Identify and analyze
business
requirements

Everything is
documented into a
document for
reference

BUSINESS REQUIREMENTS DOCUMENT (BRD)

Predictive
and
Adaptive

Planning
Phase

BRD

Activity
Diagrams

DELIVERABLES

A single document: Business Requirements Document (BRD)

Describe business requirements

The BRD will be revised as the project progresses

Key components of the BRD produced during the planning phase include:

- **Business background**
- **Business problems**
- **Organization structure**
- **Project objectives**
- ...

DELIVERABLES

Key components of the BRD produced during the planning phase include:

- ...
- **Feasibility analysis**
- **Resource planning (manpower, timeline, etc)**
- **Team configuration**
- **Business processes**

Business processes can be analyzed using modeling techniques (UML)

BUSINESS REQUIREMENT DOCUMENT (BRD)

Acts as a contract between the business and the developer

So, it's important that all the requirements are documented completely and correctly

If a requirement is not found in the BRD → it's not part of the contract

RECALL: UML



Unified Modeling Language (UML)

- Standard set of model constructs
- Designed for Object-Oriented Development

ACTIVITY DIAGRAMS

Predictive
and
Adaptive

Planning
Phase

BRD

Activity
Diagrams

ACTIVITY DIAGRAMS

Activity diagrams is a modeling technique for documenting the workflow in a graphical manner

- It can be used to model various activities in SDLC
- During the planning phase, it can be used for modeling the business processes

UML defines a set of notations for activity diagrams

ACTIVITY DIAGRAMS ELEMENTS

Initial node: indicates where the workflow begins

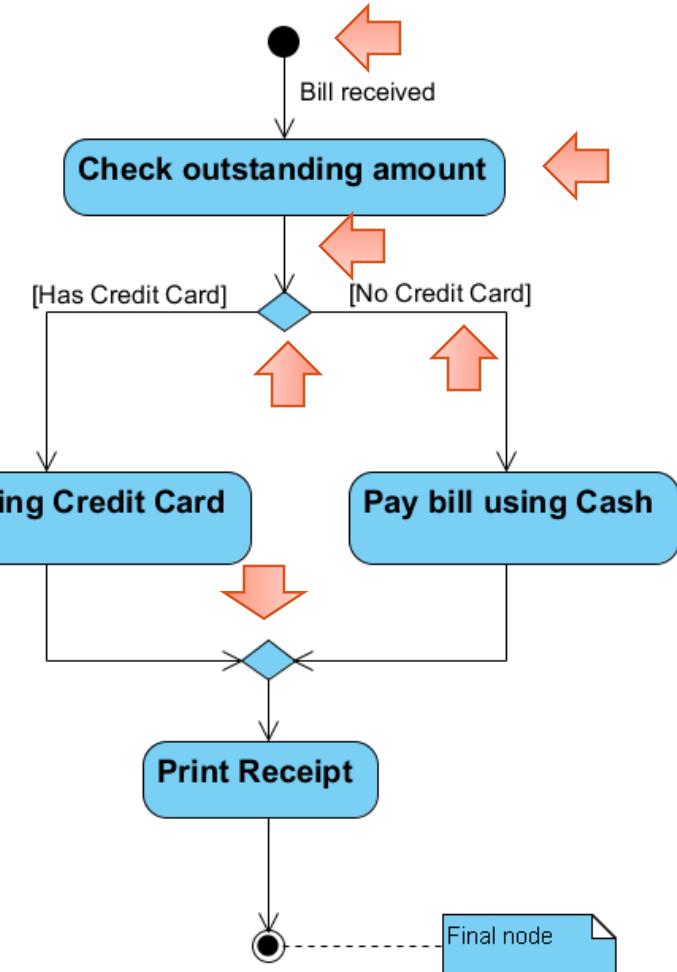
Activity: indicates a step in the process.
Notice anything about the typical naming convention?

Control flow: an arrow showing the direction of the workflow

Decision: a diamond symbol, indicating a possibility of different paths

Guard condition: a condition attached to a control flow. A guard is shown within square brackets

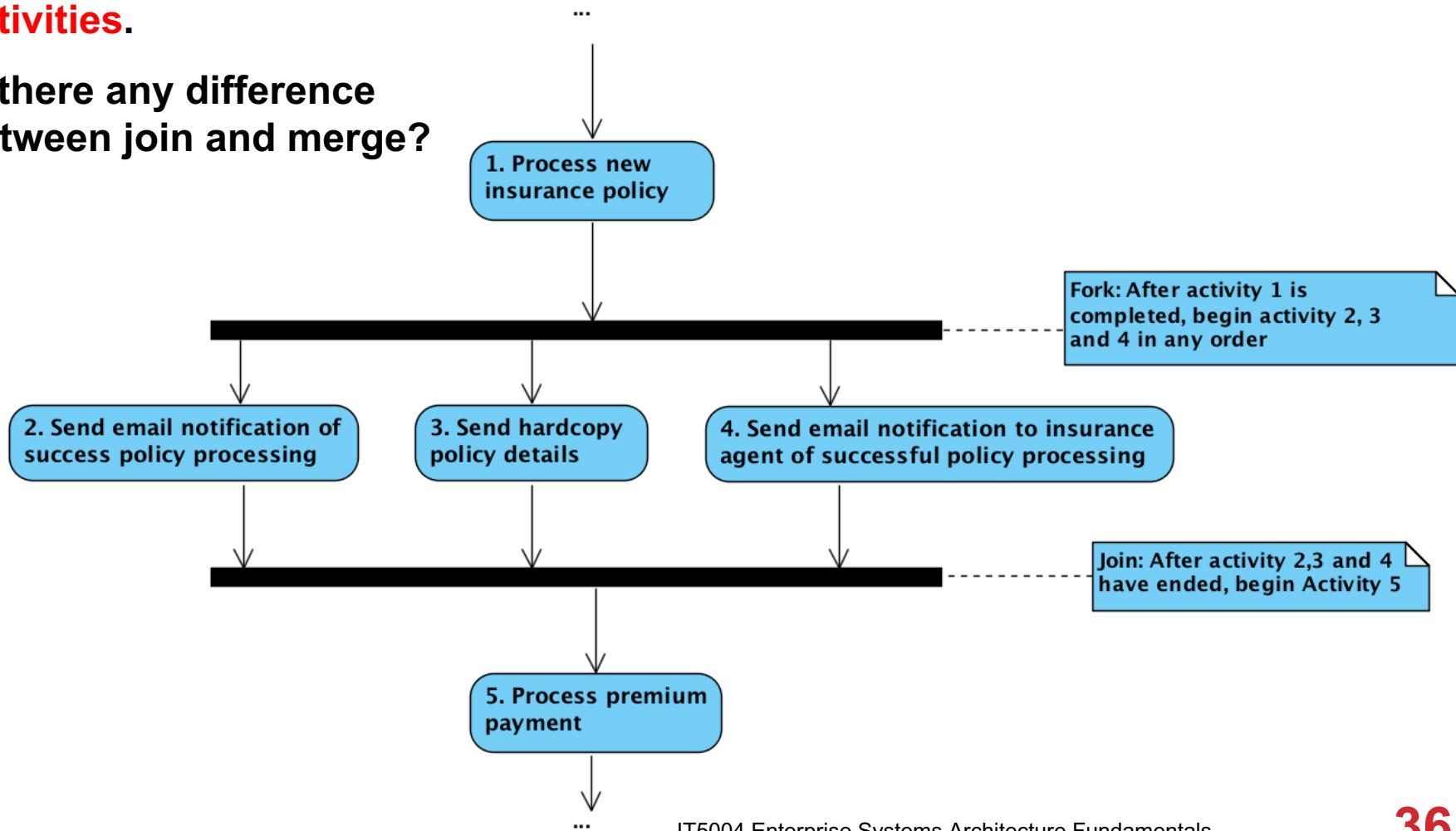
Merge: model a number of alternative flows that lead to the same activity



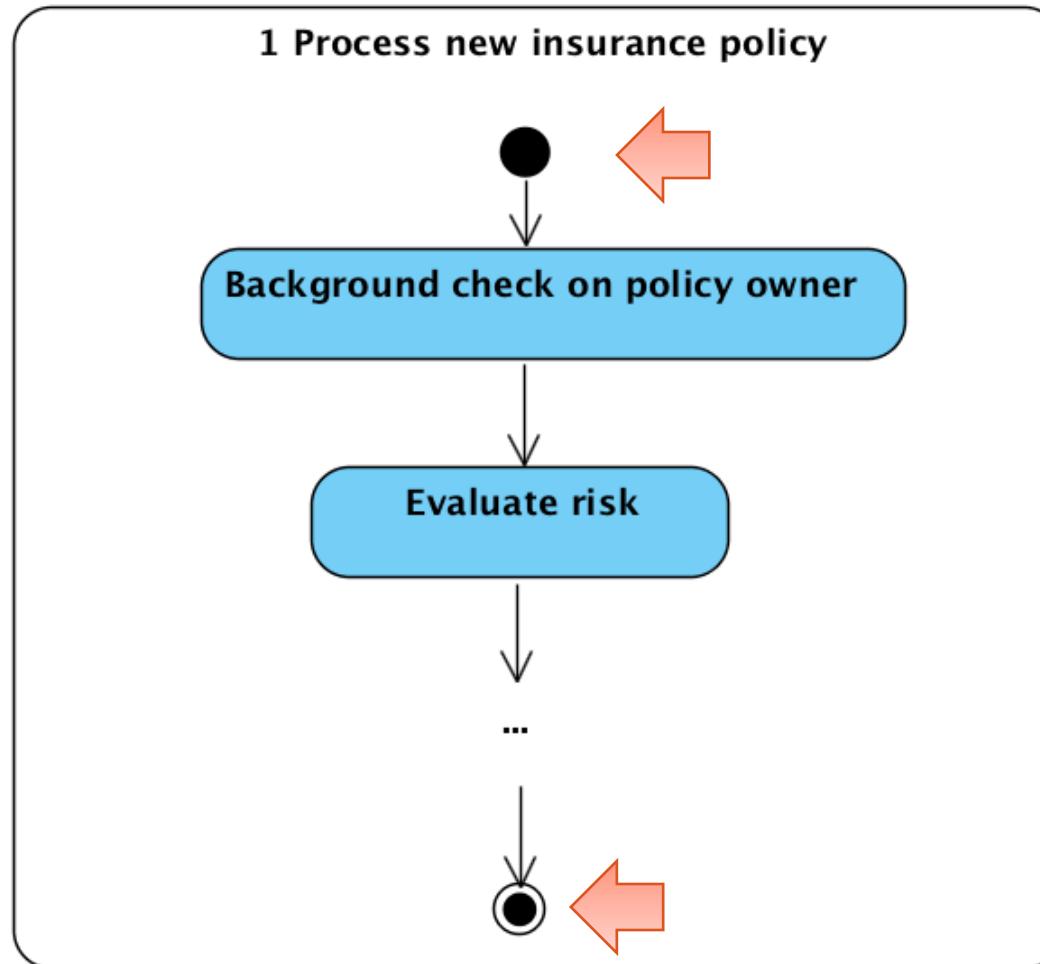
ACTIVITY DIAGRAMS ELEMENTS

Fork and Join: bars used to document parallel activities.

Is there any difference between join and merge?

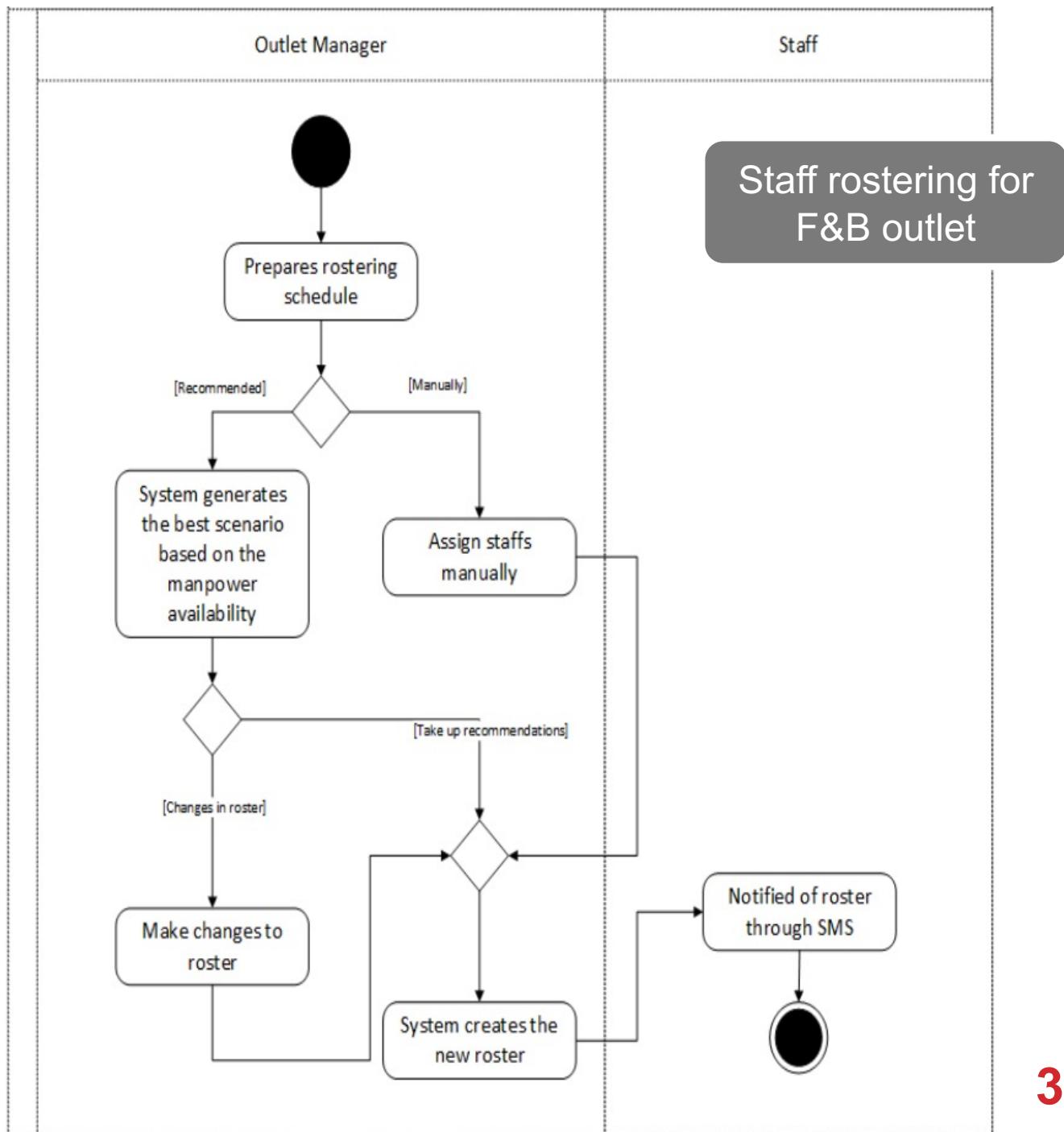


NESTED ACTIVITIES

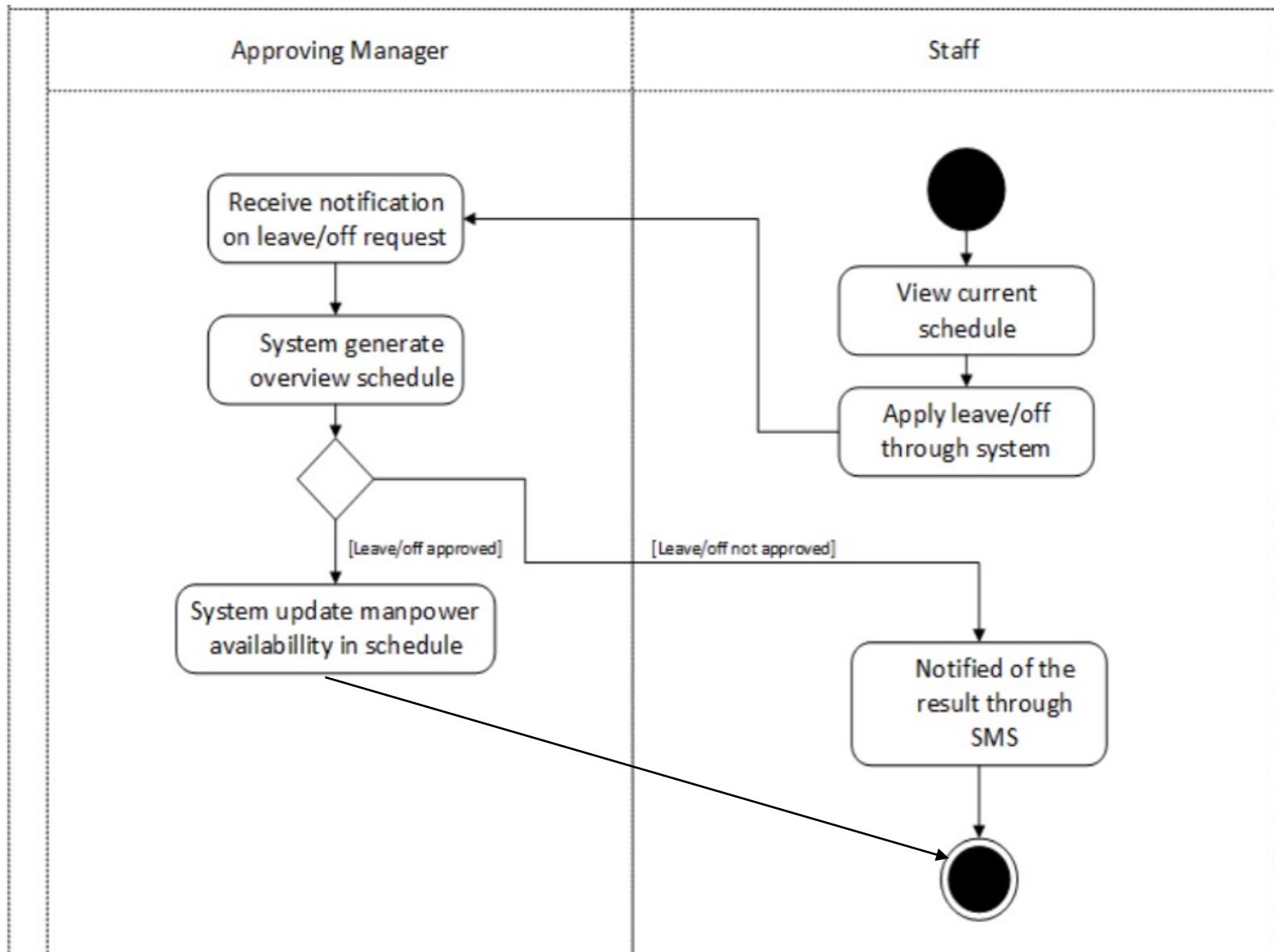


Recommended approach

Use “swimlane” to partition the activities to different parties



Process Leave application



UML DRAWING SOFTWARE

Free UML drawing software available

- Visual Paradigm Community Edition
(<https://www.visual-paradigm.com/download/community.jsp>)
- Draw.io (<https://www.draw.io/>)

A close-up photograph of a man with dark hair and glasses, wearing a grey shirt. He is holding a plain white rectangular card in front of his chest with both hands. His fingers are visible at the edges of the card. The background is a soft-focus green.

It's your turn...

TASK

Draw an activity diagram (swimlane approach**) that models how a customer would buy movie tickets off the counter**

Think about (non-exhaustive):

- What are the parties involved (swimlanes)?
- What information does the customer need to give?
- What are the step-by-step interactions
- Make sure that the notations are correct

TASK

What are the equipment available?

- Is there a tablet for users to choose the seats in front of the counter or does the staff need to make suggestions by looking at his/her screen?

Have you considered the payment options?

- Cash, NETS, Credit Card
- For each case what needs to happen?

SUMMARY

Predictive SDLC vs Adaptive SDLC

Planning Phase Activities

Business Requirements Document

Modeling Business Processes using Activity Diagrams

WHAT'S NEXT?

Requirements Gathering

LECTURE 3

REQUIREMENTS

GATHERING

LEK HSIANG HUI

READINGS

Beginning Software Engineering chapter 4

<https://onlinelibrary-wiley-com.libproxy1.nus.edu.sg/doi/pdf/10.1002/9781119209515.ch4>

LEARNING OBJECTIVES

At the end of this lecture, you should understand:

- What Requirements are and the types of Requirements
- Some of the Requirements Gathering approaches
- How to draw Use Case Diagrams

INTRODUCTION TO REQUIREMENTS

Introduction to Requirements

Types of Requirements

Requirements Gathering Approaches

Use Case Diagrams

REQUIREMENTS

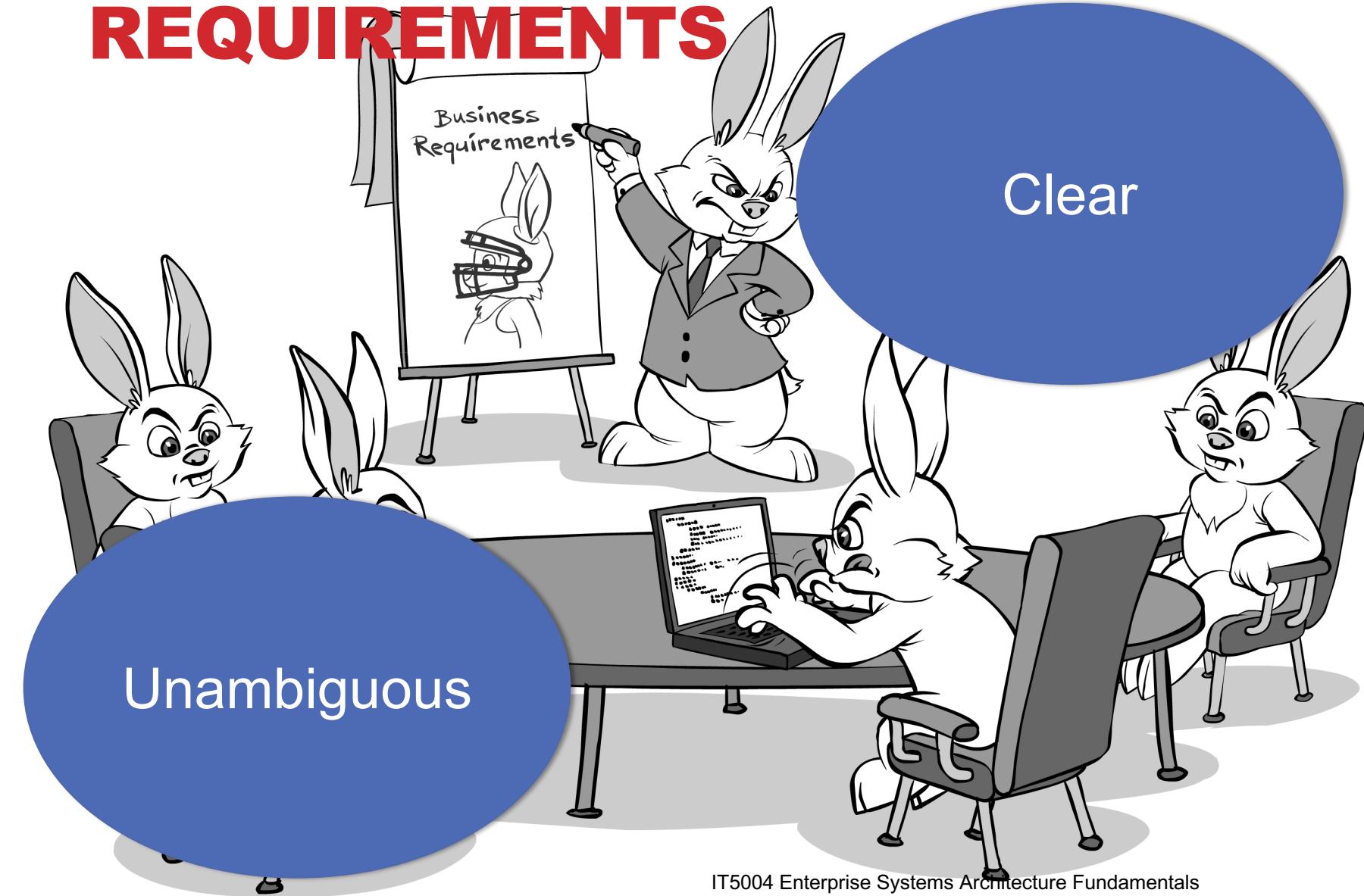
Requirements are features, functionalities, constraints, and qualities that a software must possess

One of the reasons why we use Activity Diagrams is to allow us to discover the requirements

Requirements guides the developments:

- It helps to define the project scope
- Can be used to verify that the finished system addresses the business needs

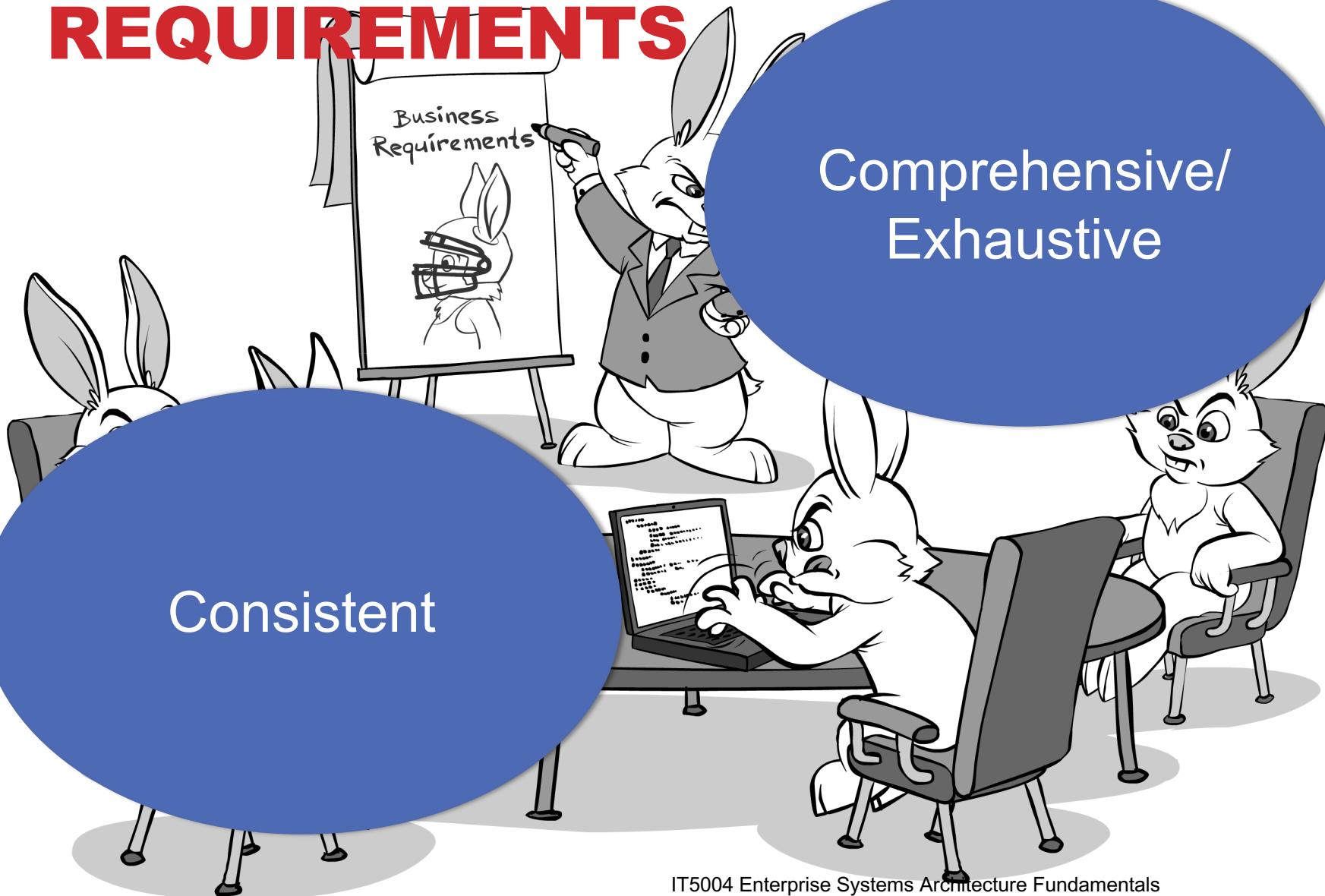
PROPERTIES OF REQUIREMENTS



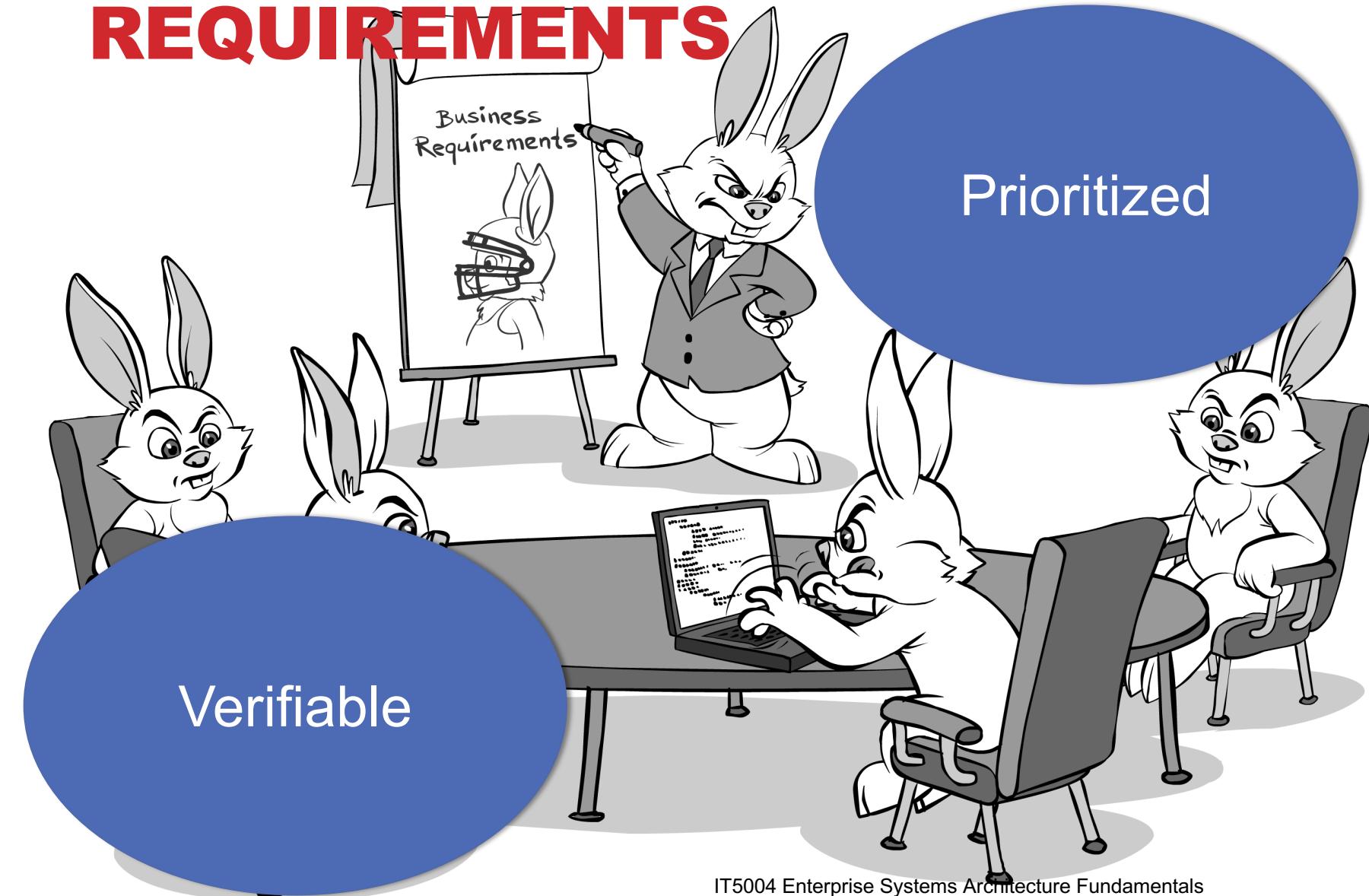
PROPERTIES OF REQUIREMENTS

Consistent

Comprehensive/
Exhaustive



PROPERTIES OF REQUIREMENTS



TYPES OF REQUIREMENTS

Introduction to Requirements

Types of Requirements

Requirements Gathering Approaches

Use Case Diagrams

TYPES OF REQUIREMENTS

Different types of requirements

- Functional Requirements
- Nonfunctional Requirements
- Implementation Requirements

FUNCTIONAL REQUIREMENTS

Restaurant ordering application



Describes what the system should do

Features your customers get in restaurant menu ordering application:

- Signup / login
- Profile settings
- My orders
- Can find restaurant based on category
- Can view the list of restaurant
- Search restaurant by category, food name, etc
- Can view full details of restaurant with reviews & menu list
- Can share reviews of restaurant
- Check out process -
- Can add menu to cart
- User selects food delivery now or later
- Updates on delivery & billing details
- Usage of discount voucher
- Select payment option - COD/ credit or debit card

NONFUNCTIONAL REQUIREMENTS

Describes various aspects/quality of the system:

- Performance (speed)
- Security
- Reliability
- Scalability
- Connectivity
- Etc

They describe properties of the system not captured by functional requirements

IMPLEMENTATION REQUIREMENTS

Temporary requirements that are needed in order to transit from using the old system to new system

Does not necessarily involve programming

- E.g. Ensures old database schema is properly migrated to new database schema naming
- Prepare training videos and materials
- Ensure that server is upgraded to 64GB RAM when deploying new system
- Etc

REQUIREMENTS GATHERING APPROACHES



Introduction to Requirements

Types of Requirements

Requirements
Gathering
Approaches

Use Case
Diagrams

REQUIREMENTS GATHERING APPROACHES

Approaches for gathering requirements include:

- Interviewing end-users
- Understanding business process flow
- Studying existing systems

INTERVIEWING END-USERS

Challenges:

- Need to ask the right questions and learn to listen
- They are not developers and do not think like programmers
- End-users often do not know what they want
- They often do not know what you would need

INTERVIEWING END-USERS

Can use the 5W1H approach

- Who – who is the user?
- What – what should the application/user do?
- When – When is the system used?
- Where – Where is the application used?
- Why – Why is there a need for the system?
- How – How can we solve the problem (user might have some idea)?

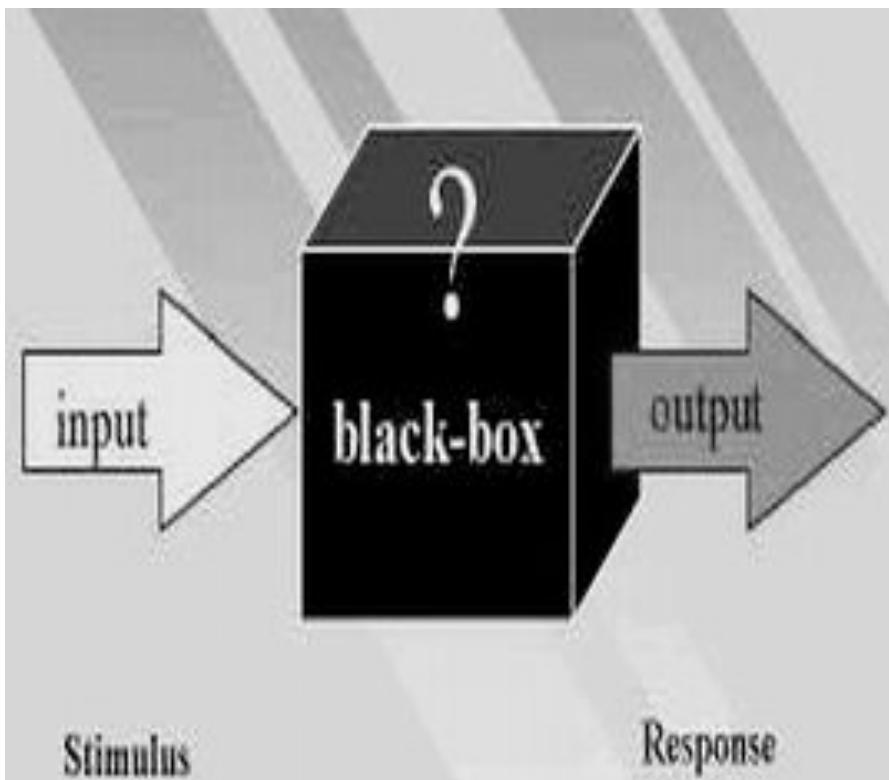
UNDERSTANDING BUSINESS PROCESS FLOW

Often, we can analyze business process flow using events

Event decomposition is a technique that can be used to identify all the events

- Not all events in a business process is relevant
- Only those that should be **remembered by the system** or those that should trigger processing in **the system** are important

EVENT DECOMPOSITION



How to identify Events?

- One approach is
Event Decomposition
- Treat system as a
black box

EVENT DECOMPOSITION



RESULTS

List of use cases
triggered by
business events

3 TYPES OF EVENTS



External Events

- Occur outside the system
- **Usually caused by external agent**
e.g. customer

Temporal Events

- Occurs when system reaches a point (deadline) in **time**

State Events

- Events that occur when something happen inside the system that **trigger the need for processing**

EXTERNAL EVENTS

Examples:

- Customer wants to login
- Customer want to check account details
- Customer want to show the list of transaction
- Customer want to search for items
- Etc

TEMPORAL EVENTS

Examples:

- Time to produce monthly report
- Time to email bill PDF
- Time to send scheduled advertisements
- Etc

STATE EVENTS

Examples (triggered when certain conditions are satisfied):

- Send instant message notification
- Send low stock email notification
- Update delivery status
- Etc

STUDY EXISTING SYSTEMS

Once we have the goals of the system, we can refine the requirements by studying similar systems

- This reduces the chance of missing requirements
- Allows us to rethink about our proposed requirements (might be able to simplify requirements or do differently)

USE CASES

We then document these requirements as **use cases** (i.e. system functionality)

Example:

- Requirement: **Customer want to show the list of transactions**
- Use Case: **Get list of transactions of customer**

- Requirement: **Customer want to buy item**
- Use Case: **Create new order**

USE CASE DIAGRAMS

Introduction to Requirements

Types of Requirements

Requirements Gathering Approaches

Use Case Diagrams

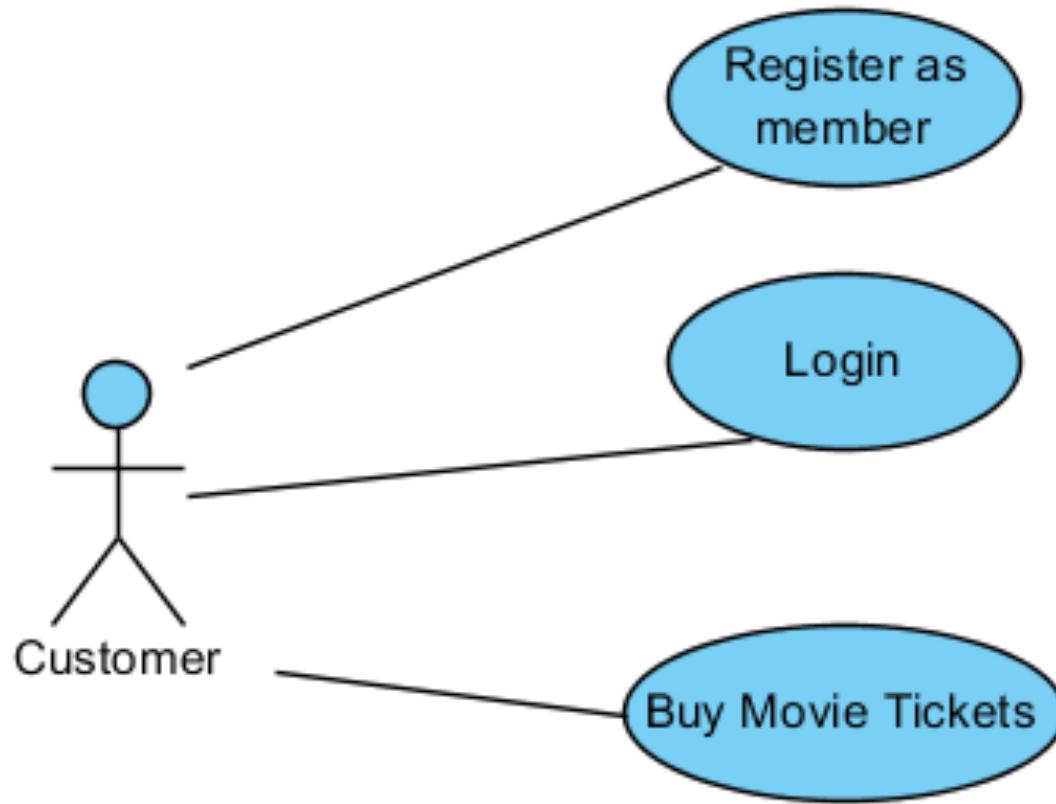
USE CASE DIAGRAMS

Use case diagrams is a **modeling technique** for documenting the **use cases**

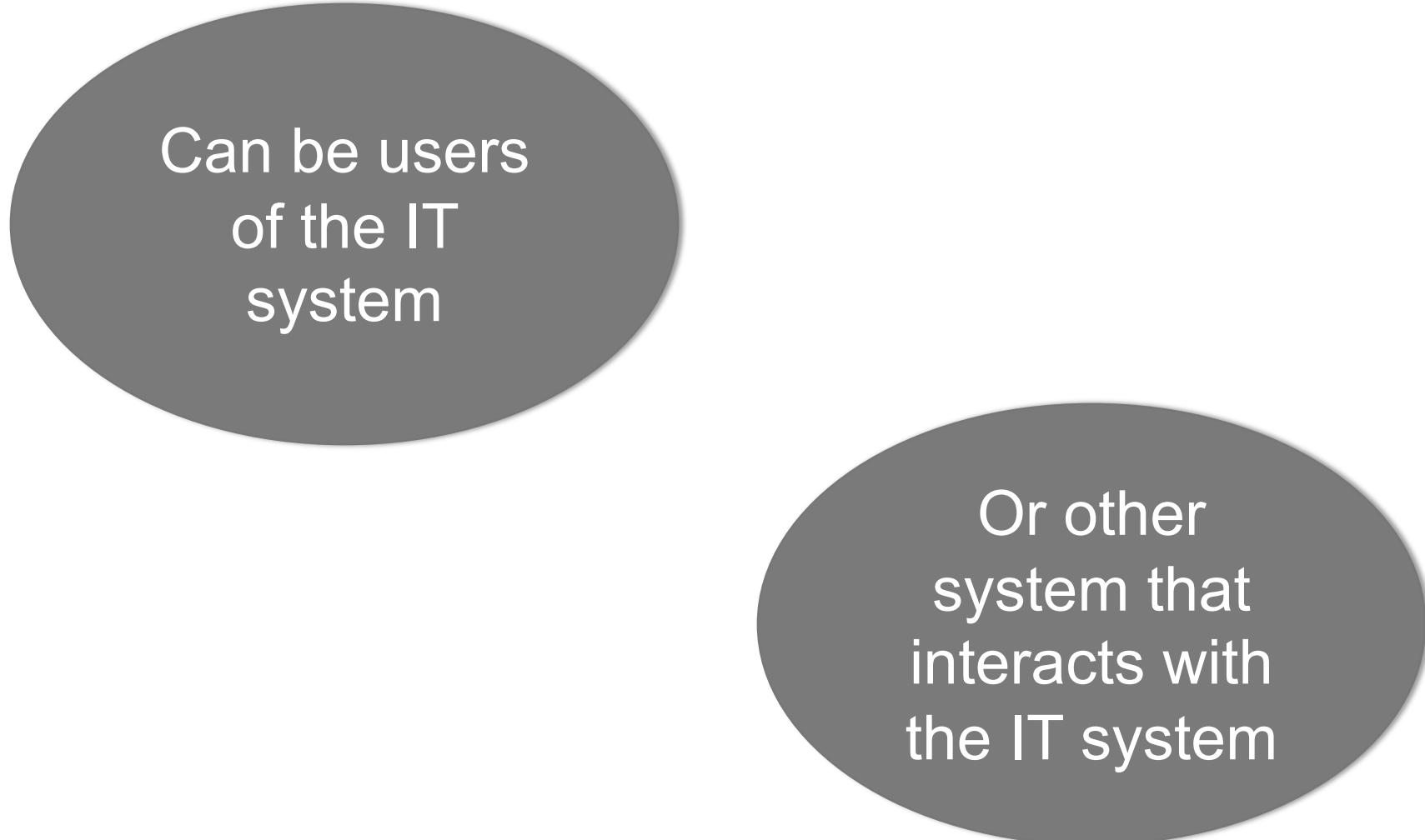
Diagram consists of 3 main elements

- Actors
- Use cases
- Association between actors to use cases

AN EXAMPLE OF A USE CASE DIAGRAM



IDENTIFY ACTORS



Can be users
of the IT
system

Or other
system that
interacts with
the IT system

QUESTIONS TO ASK

Who are the
possible
users?

Is there just
one system
or many
subsystems

How does the
system interact
with other
systems?



QUESTIONS TO ASK

What will the system be like?

Should roughly have an idea

QUESTIONS TO ASK

Who are the
possible
users?

Customers?

QUESTIONS TO ASK

Customers?

How many
types of
customers are
there?

How are the
different types
of customers
using the
system?

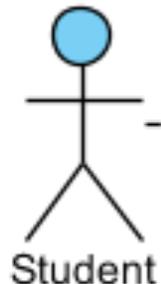
QUESTIONS TO ASK

Employees?

How are the
managers
using the
system?

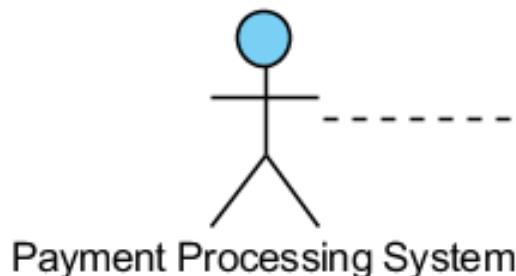
How is the
accountant
using the
system?

ACTOR



Example of a human actor

(This is a comment)



Example of an external computer system actor

(Also draw n using the stick figure)

GENERALIZE ACTOR

After identifying the actors

Generalize them

- i.e. find relations amongst these different types
- e.g. a graduate student is a more specific kind of student

Note that actors can also be systems

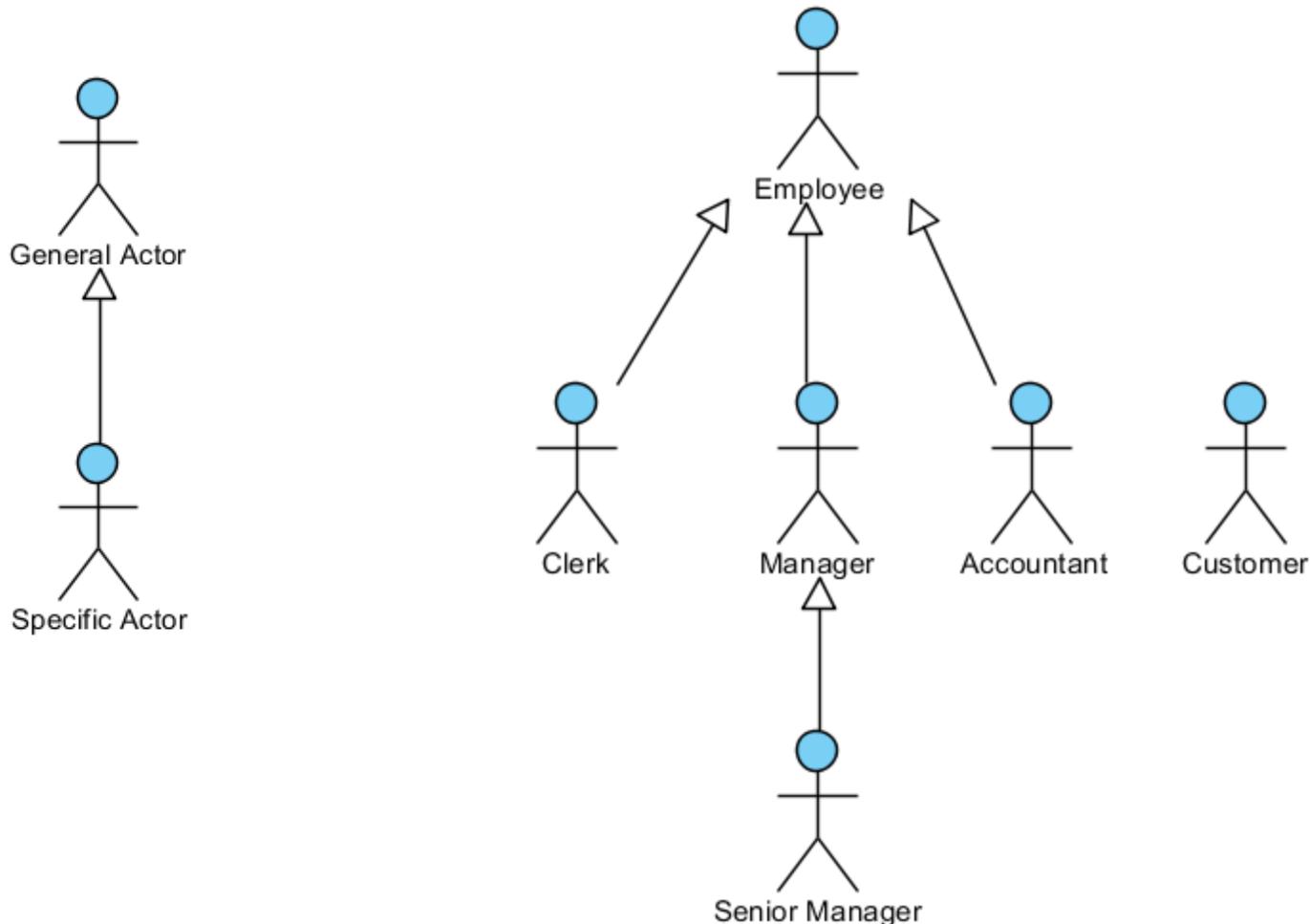
ROLE MAP

Another diagram used to standardize the treatment of users and external systems

Unlike use-case diagram, only shows actors and how actors are related

ROLE MAP

Recall: Superclass and subclass in OO programming



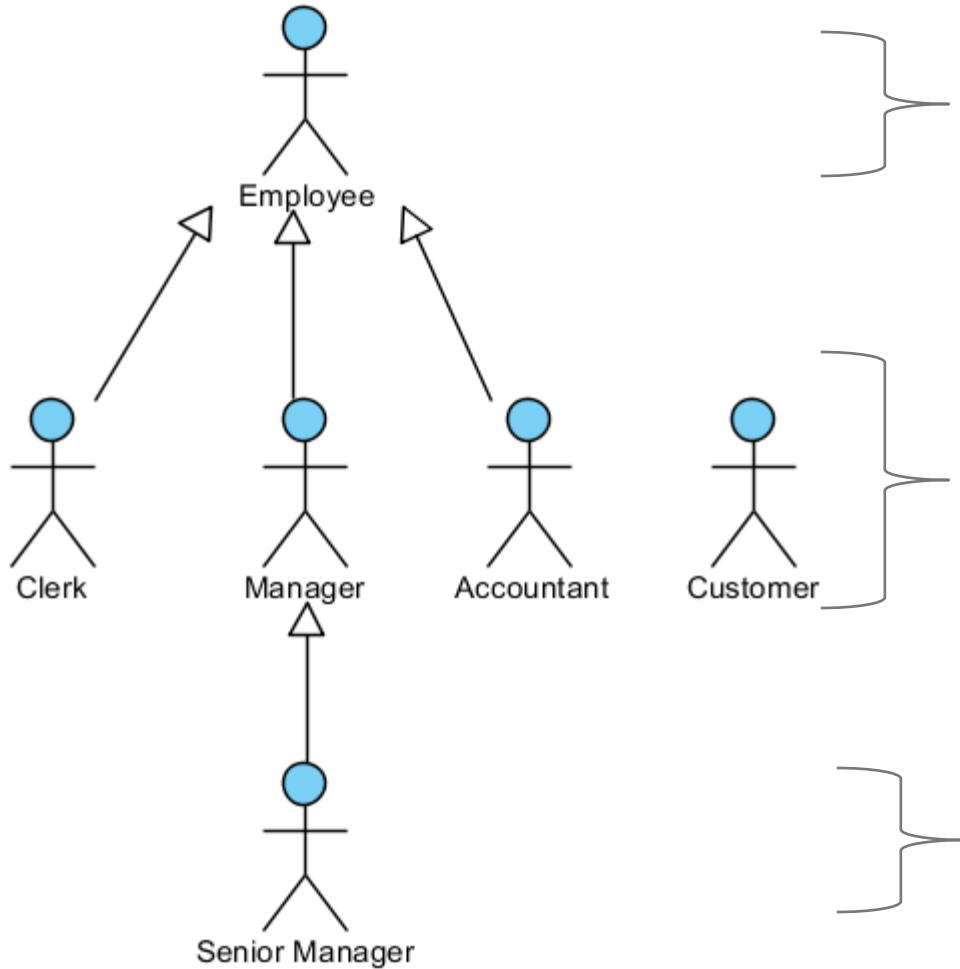
ROLE MAP

The arrow allows us to model actors with overlapping roles

**Think of the generalization relationship arrow as
“a kind of”**

- E.g.
Clerk, Accountant, Manager are different kinds of Employer
- Senior Manager is a kind of Manager

ROLE MAP



General

More Specific

More Specific
=>
they can do more things

Even More Specific

GENERALIZATION AND SPECIALIZATION

Specialized actors inherit the ability to do all the things that a generalized actor can do

Think Object-Oriented Programming (OOP)

Clerk extends (inherits) Employee

- Employee = Generalized actor
- Clerk = Specialized actor

WHY DO WE DEFINE GENERALIZED ACTORS?

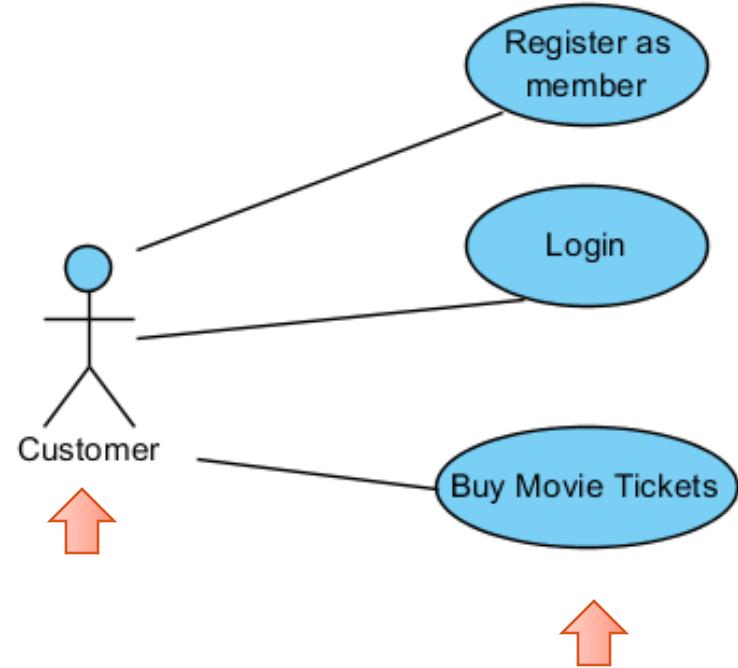
Simplify the
drawing of
use case
diagram

Reduce the
number of
association lines
(just connect to the
generalized actor)

USE CASE MODELING ELEMENTS

Actor: An actor who initiates a use-case interaction.

Use case: A user task (indicated as an oval)



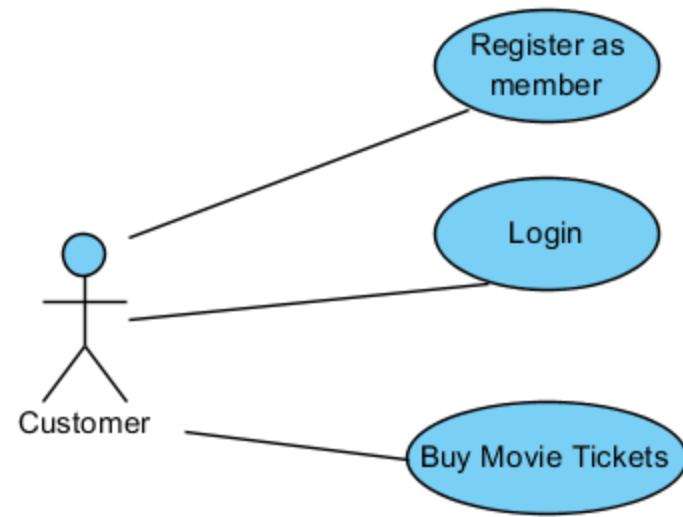
Can draw with either a line or arrow pointing to the use case

IDENTIFY USE CASES

A use case is an **interaction** between an actor and a computer system

Each use case is named using a verb

E.g. “Register as user”,
“Login”,



GUIDE FOR IDENTIFYING USE CASES

1. Need to identify all the actors of this system

- Note that actors need not be just human users, they can be **other systems** that interact with this system

2. A use case must be a system function/interaction

- E.g. clerk calling a customer is NOT a use case - it's not a system function

3. A use case should fulfil a business objective

- E.g.
Actor: Student
Use case: View list of enrolled courses

GUIDE FOR IDENTIFYING USE CASES

4. While a use case is a system function/interaction, need to know how to differentiate between a use case vs a step of a use case

- Note that a use case is a high-level system function
- We will derive the individual steps of a use case in future phases (not now)
- Do NOT try to list out the individual steps of a use case
- E.g
 - Use case: Update user profile
 - Possible steps of the “Update user profile” use case:
enter name, enter date of birth, enter email, etc
- Keep in mind that a use case must fulfil a business objective
- Ultimately whether a system interaction is a use case, or a step depends on the design of the system (will talk more about this later on)

GUIDE FOR IDENTIFYING USE CASES

- 5. Use cases are not just a function in code. It should be seen from a point of view of an actor**
 - E.g. “check password during login”, “send email confirmation”, “add to database”, etc are not use cases. They are what the system/code does
- 6. Use cases are seen from the perspective of a user rather than from the perspective of the system**
 - i.e. “get a list of chat messages” rather than “receive a chat message”
- 7. A use case diagram should not have this system itself as an actor**
 - Usually, we omit time event in the use case diagram
 - But if you really want to draw, you can create an “actor” called TIME

QUESTION TIME



1. Why do we gather requirements using a use case diagram rather than just come up with a list of system functions?

A close-up photograph of a man with dark hair and glasses, wearing a light-colored shirt. He is holding a plain white rectangular card in front of his chest with both hands. His fingers are visible at the edges of the card. The background is a soft-focus green.

It's your turn...

TASK

**Come up with the Use Case Diagram for the
Canvas system**

SUMMARY

Properties of Requirements

Types of Requirements : Functional, Nonfunctional, Implementation Requirements

Requirements Gathering Approaches

Role Map and Use Case Diagrams

WHAT'S NEXT?

Requirements Analysis

LECTURE 4

REQUIREMENTS

ANALYSIS

LEK HSIANG HUI

LEARNING OBJECTIVES

At the end of this lecture, you should understand:

- The activities in the analysis phase
- How to produce use case descriptions
- How to apply advanced use case features to use case diagrams and use case descriptions
- How to organize use cases in use case diagram

INTRODUCTION TO ANALYSIS PHASE

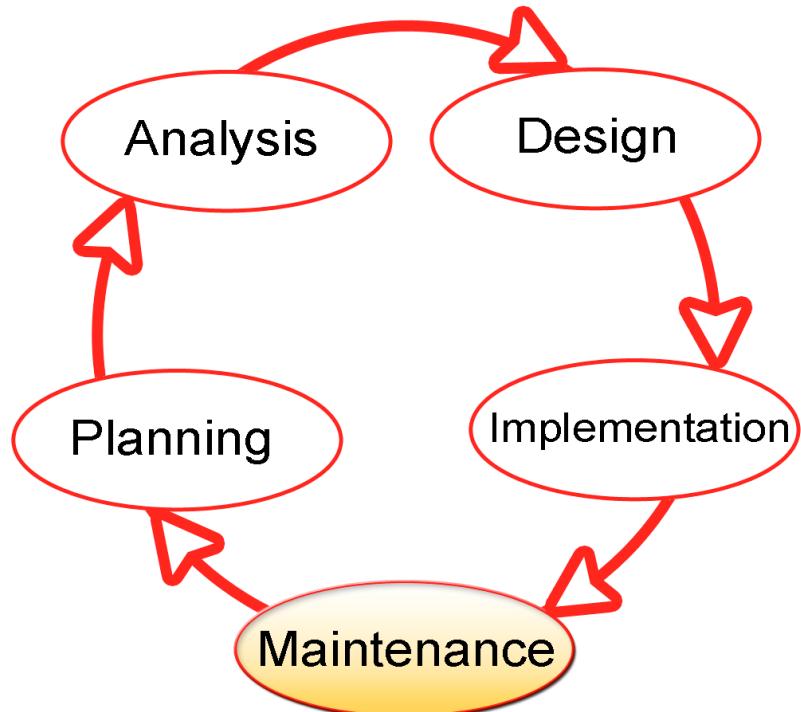
Introduction to Analysis Phase

Behavioral Modeling using Use Case Description

Advanced Use-Case Features

Organizing Use Cases

ANALYSIS PHASE



We can generalize SDLC into 5 different phases:

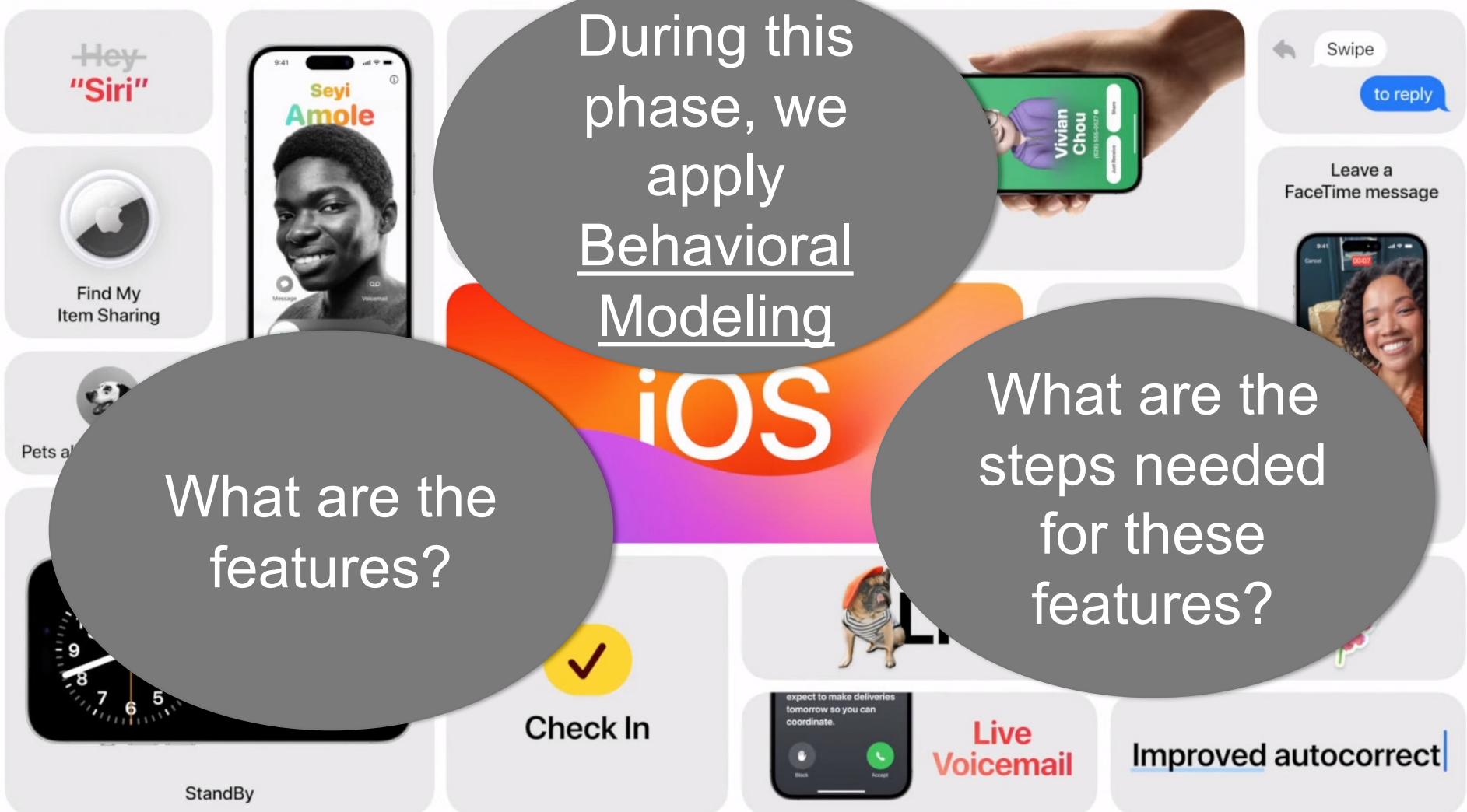
1. Planning
2. **Analysis**
3. Design
4. Implementation
5. Support/Maintenance

BEHAVIORAL MODELING

What are the features?

During this phase, we apply Behavioral Modeling

What are the steps needed for these features?



USE CASES

Use cases and use case diagrams are tools for discovering high-level system functionality

- However, they only provides very limited information

The next step is to convert use cases into **use case descriptions**

BEHAVIORAL MODELING USING USE CASE DESCRIPTION



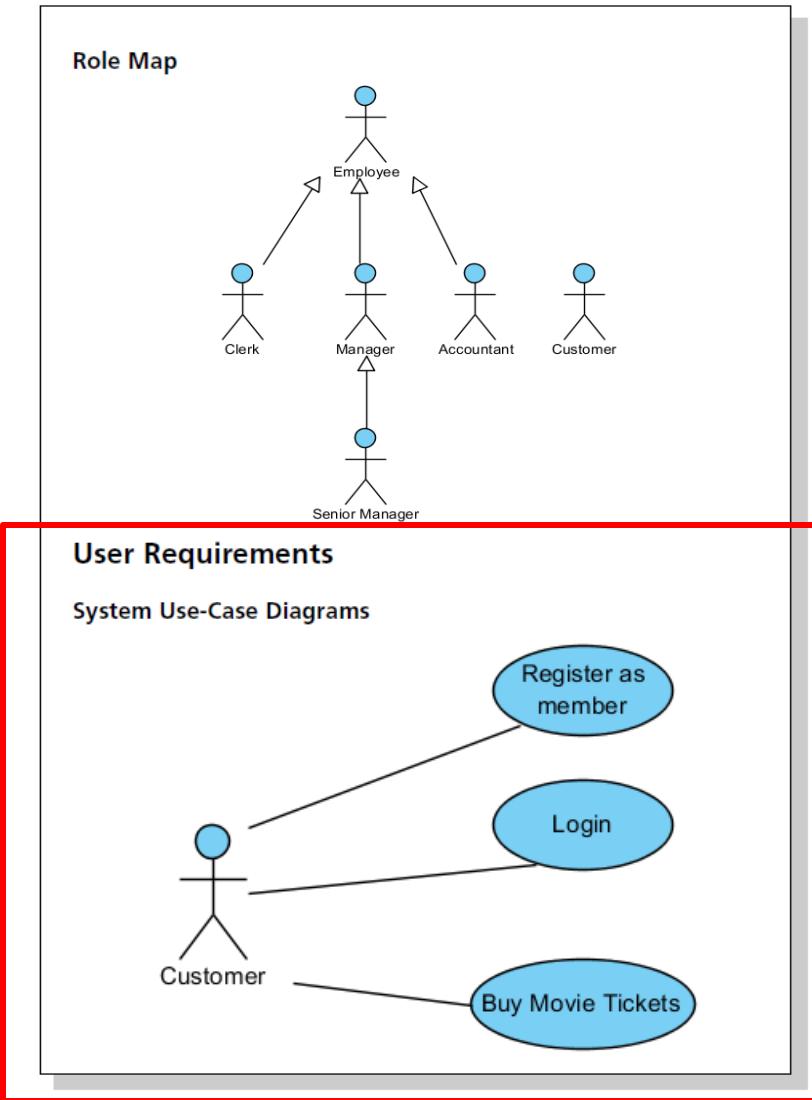
Introduction to Analysis Phase

Behavioral Modeling using Use Case Description

Advanced Use-Case Features

Organizing Use Cases

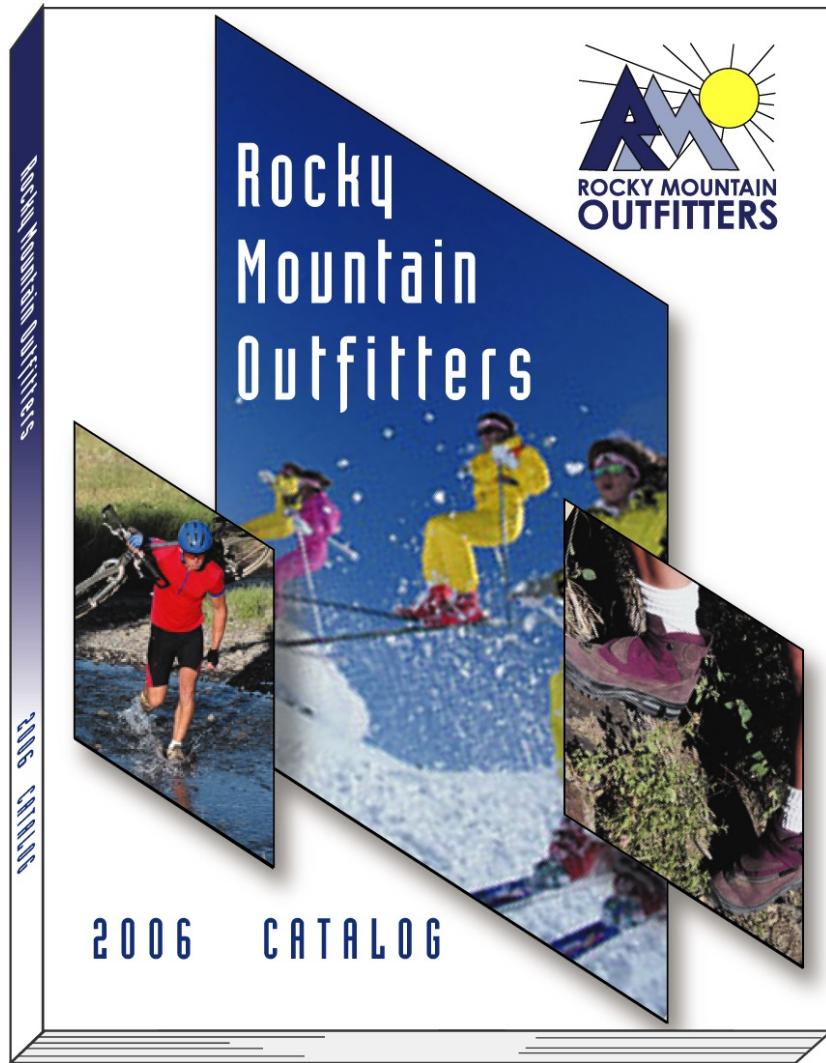
SYSTEM USE CASES



What we have
covered in
Lecture 3

Next thing is to
convert the
System Use
Cases to Use
Case Description

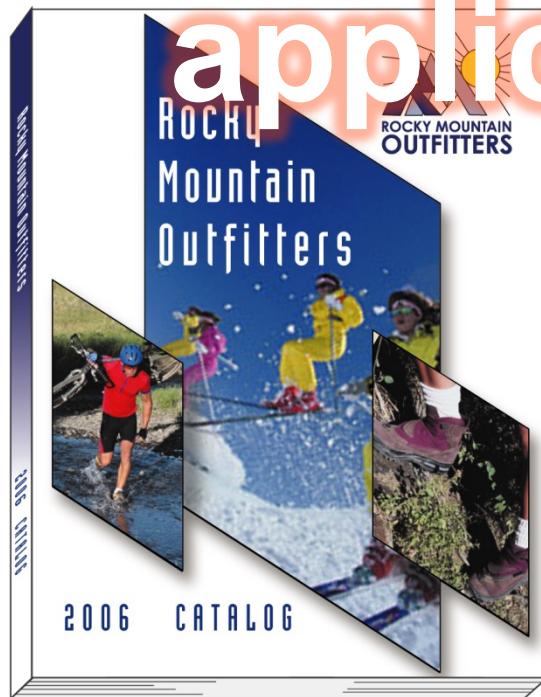
ROCKY MOUNTAIN OUTFITTER CASE STUDY

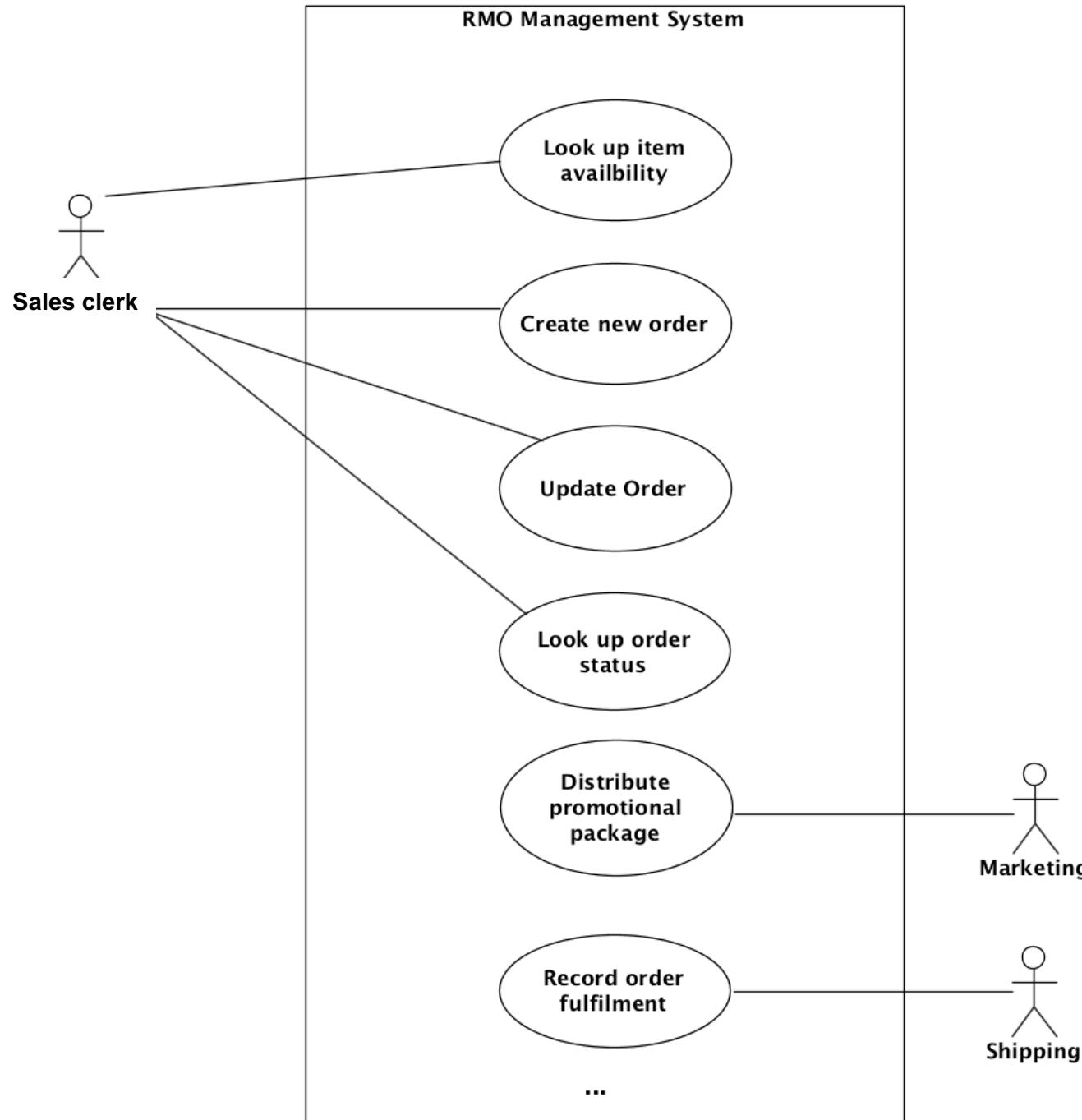


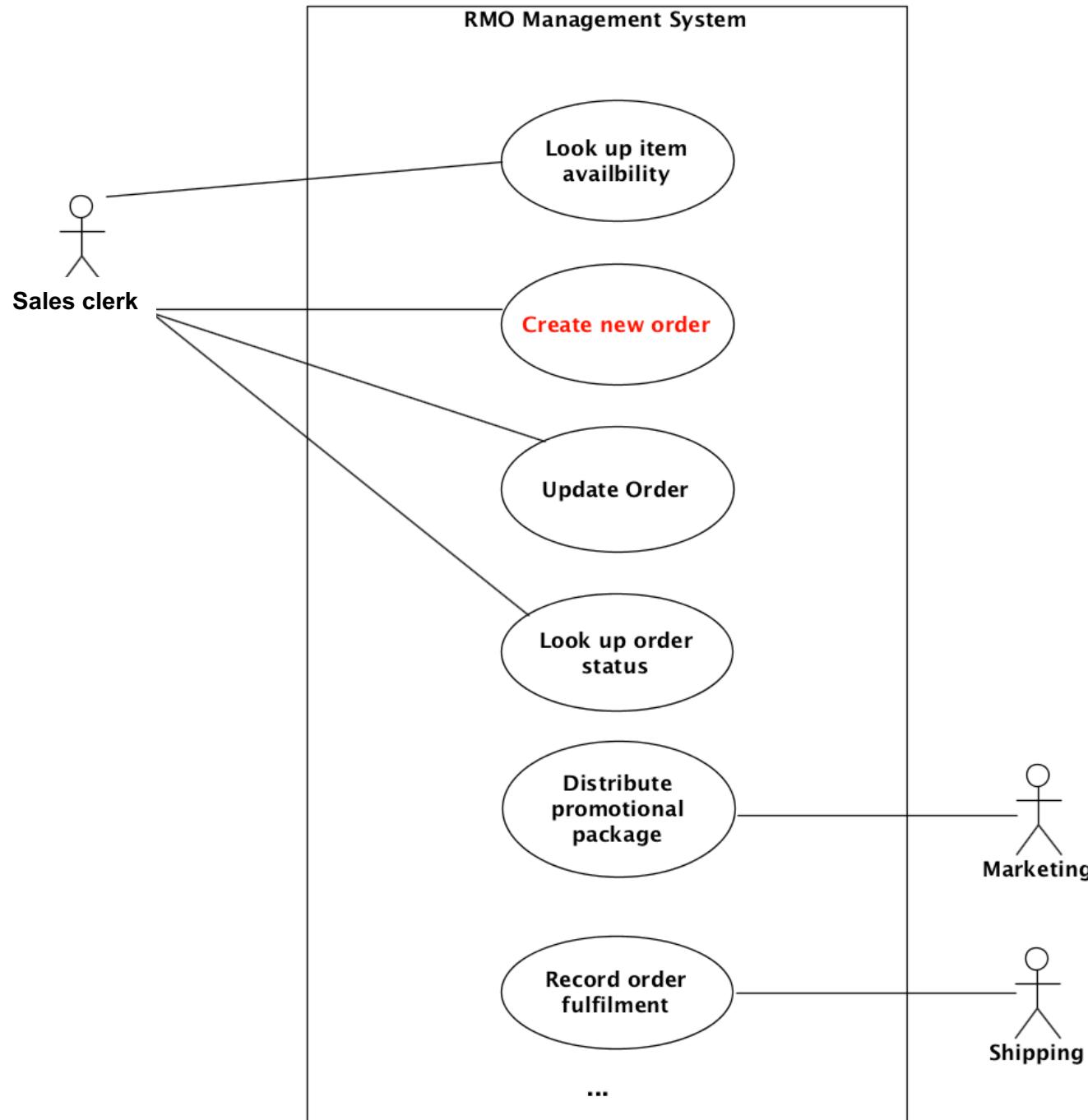
Rocky Mountain Outfitters (RMO) is a sports clothing distributor

- Significant regional distributor
- Employs more than 600 people
- \$100 million annual sales

What do you think are
some of the use cases
applicable to RMO?







CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order	
Triggering event	Clerk wants to create a new telephone order	
Brief description	When customer calls in to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.	
Actors	Telephone sales clerk	
Pre-conditions	-	
Post-conditions	Order and order line items must be created.	
Flow of events	Actor	System

Exception conditions	...	

CREATE NEW ORDER

USE CASE DESCRIPTION

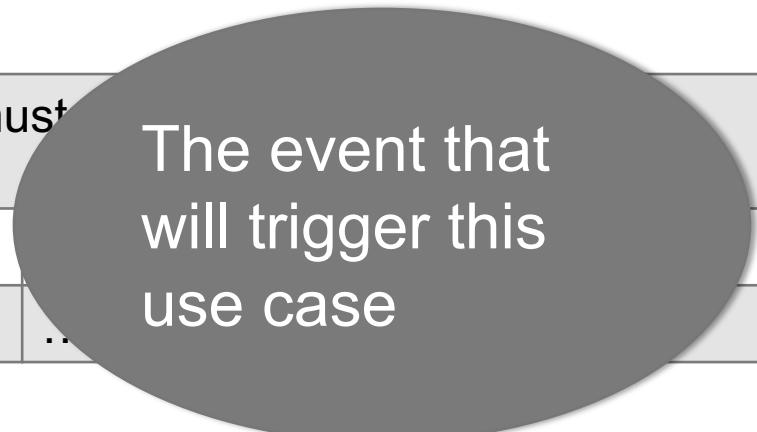
Use case name	Create new order				
Triggering event	Clerk wants to create a new telephone order				
Brief description	When customer calls in to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.				
Actors	Telephone sales clerk				
Pre-conditions	-				
Post-conditions	Order and order line items must be created.				
Flow of events	<table border="1"><thead><tr><th>Actor</th><th></th></tr></thead><tbody><tr><td>...</td><td>...</td></tr></tbody></table>	Actor	
Actor					
...	...				
Exception conditions	...				

The use case
in the use
case diagram

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order
Triggering event	Clerk wants to create a new telephone order
Brief description	When customer calls in to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.
Actors	Telephone sales clerk
Pre-conditions	-
Post-conditions	Order and order line items must be created.
Flow of events	Actor
	...
Exception conditions	...



The event that will trigger this use case

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order
Triggering event	Clerk wants to create a new telephone order
Brief description	When customer calls in to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.
Actors	Telephone sales clerk
Pre-conditions	-
Post-conditions	Order and order line item
Flow of events	Actor
	...
Exception conditions	...



Give a more complete description to supplement the Triggering Event

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order
Triggering event	Clerk wants to create a new telephone order
Brief description	When customer calls in to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.
Actors	Telephone sales clerk
Pre-conditions	-
Post-conditions	Order and order line items
Flow of events	Actor
Exception conditions	...

The actor that will execute the use case (i.e. the person/system that will be interacting directly with the system)

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order	
Triggering event	Clerk wants to create a new order.	
Brief description	When customer calls in to order, verify customer information, create a new order record, add items to the order, verify payment, create the order transaction, and finalize the order.	
Actors	Telephone sales clerk	
Pre-conditions	-	
Post-conditions	Order and order line items must be created.	
Flow of events	Actor	System

Exception conditions	...	

This defines the state of the system before this use case can be executed

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order
Triggering event	Clerk wants to create a new order.
Brief description	When customer calls in to order, verify customer information, create a new order record, add items to the order, verify payment, create the order transaction, and finalize the order.
Actors	Telephone sales clerk
Pre-conditions	-
Post-conditions	Order and order line items must be created.
Flow of events	Actor
	...
System	
Exception conditions	...

This defines the state of the system after this use case is executed

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order	
Triggering event	Clerk wants to create a new order.	
Brief description	When customer calls in to order, the clerk will ask for order information, verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.	
Actors	Telephone sales clerk	
Pre-conditions	-	
Post-conditions	Order and order line items must be created.	
Flow of events	Actor	System

Exception conditions	...	

Series of interaction between the actor and the system

Use case name	Create new order	
...	...	
Flow of events	<p>Actor</p> <ol style="list-style-type: none"> 1. Enters customer details (name, phone number) supplied by customer 2. Initiates the creation of a new order 3. Searches for item that the customer wants 4. Adds item to order 5. Repeat step 3 and 4 until all items are added to the order 6. Clicks on “Complete Order” 7. Enters payment details (cc number, expiry date, CVV) and payment mode 	<p>System</p> <ol style="list-style-type: none"> 1.1 Verifies customer information 2.1 Creates a new order 3.1 Displays item information 4.1 Adds order item 6.1 Displays order summary 7.1 Verifies payment details 7.2 Create order transaction 7.3 Sends order confirmation email

FLOW OF EVENTS

We focus mainly on the interaction between the **actor** and the **system**

- Note that in this case, the actor (clerk) is just a proxy on the customer's behalf
- Even though the customer is the one telling the clerk the necessary information, the clerk is the one interacting with the system
- So the clerk (not the customer) is the actor

The flow of events focuses on documenting the steps

FLOW OF EVENTS

Other things to take note:

- The steps must be exhaustive
- The steps must also connected with each other in a logical manner
- For e.g., before the clerk adds the item, the system should show the item details (so that clerk is sure that it is the right item)
- Note that these steps will be used by the system design process later so it is important to be as detailed as possible

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	

Another example

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	
4. Initiates check out	

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	
4. Initiates check out	4.1 Displays item summary page

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	
4. Initiates check out	4.1 Displays item summary page
5. Enters credit card details (cc number, expiry date, CVV) and complete order	

SYSTEM USE CASE OF A TYPICAL E-COMMERCE SITE: BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	
4. Initiates check out	4.1 Displays item summary page
5. Enters credit card details (cc number, expiry date, CVV) and complete order	5.1 Displays order confirmation 5.2 Creates order transaction 5.3 Sends order confirmation email

ALTERNATE FLOW

ALTERNATE FLOWS



For documenting conditional scenarios

- More than one possible option
- User not logged in
- Depending on the customer type, system may want to react differently (sms if number is available, email if email is available, etc)
- etc

BUY ITEMS

Actor	Want to handle differently depending whether the user is logged in
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
2. Selects an item, input quantity and add to shopping cart	2.1 Updates the current shopping cart by adding the item
3. Repeat step 1 and 2 until all the items have been added	
4. Initiates check out	4.1 Displays item summary page
5. Enters credit card details (cc number, expiry date, CVV) and complete order	5.1 Displays order confirmation 5.2 Creates order transaction 5.3 Sends order confirmation email

BUY ITEMS

Actor	Want to handle differently depending whether the user is logged in
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
...	...
4. Initiates check out	4.1 Displays item summary page
5. Enters credit card details (cc number, expiry date, CVV) and complete order	5.1 Displays order confirmation 5.2 Creates order transaction 5.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords	1.1 Displays the listing associated with keywords
...	...
4. Initiates check out	4.1 Displays item summary page
...	...
6. Enters credit card details (cc number, expiry date, CVV) and complete order	6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 4. Initiates check out ... 6. Enters credit card details (cc number, expiry date, CVV) and complete order	1.1 Displays the listing associated with keywords ... 4.1 Displays item summary page ... 6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 4. Initiates check out 5. If the buyer is logged in, go to step 6 ... 6. Enters credit card details (cc number, expiry date, CVV) and complete order	1.1 Displays the listing associated with keywords ... 4.1 Displays item summary page ... 6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 ... 6. Enters credit card details (cc number, expiry date, CVV) and complete order	1.1 Displays the listing associated with keywords 6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member ... 6. Enters credit card details (cc number, expiry date, CVV) and complete order	1.1 Displays the listing associated with keywords 6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member	1.1 Displays the listing associated with keywords
5b. If the buyer is not logged in, but is a member, he/she enters login details	...
6. Enters credit card details (cc number, expiry date, CVV) and complete order	6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member	1.1 Displays the listing associated with keywords ... 5a.1 Creates a new member record and save the record in database
5b. If the buyer is not logged in, but is a member, he/she enters login details	...
6. Enters credit card details (cc number, expiry date, CVV) and complete order	6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member 5b. If the buyer is not logged in, but is a member, he/she enters login details 6. Enters credit card details (cc number, expiry date, CVV) and complete order	1.1 Displays the listing associated with keywords ... 5a.1 Creates a new member record and save the record in database 5a.2 Logs the user into the system and display the form for user to enter credit card details ... 6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6 5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member	1.1 Displays the listing associated with keywords ... 5a.1 Creates a new member record and save the record in database 5a.2 Logs the user into the system and display the form for user to enter credit card details
5b. If the buyer is not logged in, but is a member, he/she enters login details	5b.1 Verifies the login details and log the user into the system ...
6. Enters credit card details (cc number, expiry date, CVV) and complete order	6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

BUY ITEMS

Actor	System
1. Searches for listings using keywords ... 5. If the buyer is logged in, go to step 6	1.1 Displays the listing associated with keywords ...
5a. If the buyer is not logged in, but is not a member, he/she enters his/her personal details to register as a new member	5a.1 Creates a new member record and save the record in database 5a.2 Logs the user into the system and display the form for user to enter credit card details
5b. If the buyer is not logged in, but is a member, he/she enters login details	5b.1 Verifies the login details and log the user into the system 5b.2 Logs the user into the system and display the form for user to enter credit card details
6. Enters credit card details (cc number, expiry date, CVV) and complete order	6.1 Displays order confirmation 6.2 Creates order transaction 6.3 Sends order confirmation email

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order	
Triggering event	Create new telephone order.	
Brief description	When customer calls, get customer information, create order, verify payment, and then process the order.	
Actors	Telephone sales clerk	
Pre-conditions	-	
Post-conditions	Order and order line items must be created.	
Flow of events	Actor	System

Exception conditions	...	

Unexpected conditions which we cannot proceed further (i.e. cannot handle by alternate flow and have to terminate)

Use case name	Create new order	Credit card verification failed
...	...	
Flow of events	<p>Actor</p> <ol style="list-style-type: none"> 1. Enter customer details supplied by customer 2. Initiates the creation of a new order 3. Searches for item that the customer wants 4. Adds item to order 5. Repeat step 3 and 4 until all items are added to the order 6. Clicks on “Complete Order” 7. Enters payment details (cc number, expiry date, CVV) and payment mode 	<p>System</p> <ol style="list-style-type: none"> 1.1 Verifies customer information 2.1 Creates a new order 3.1 Displays item information 4.1 Adds order item 6.1 Displays order summary 7.1 Verifies payment details 7.2 Create order transaction 7.3 Sends order confirmation

CREATE NEW ORDER

USE CASE DESCRIPTION

Use case name	Create new order				
Triggering event	Clerk wants to create				
Brief description	When customer calls, verify customer info, enter the order, verify payment, finalize the order.				
Actors	Telephone sales clerk				
Pre-conditions	-				
Post-conditions	Order and order line items created				
Flow of events	<table border="1"><thead><tr><th>Actor</th><th>System</th></tr></thead><tbody><tr><td>...</td><td>...</td></tr></tbody></table>	Actor	System
Actor	System				
...	...				
Exception conditions	7.1 Credit card verification failed				

Why we have to terminate in this case?

Because verification is done with the bank later on (by that time the user might no longer be on that screen)

EXCEPTION CONDITIONS

Things that can be resolved by alternate flow should **not** be exception conditions

- When the exception conditions happen, the use case terminates immediately
- Examples of exception conditions: credit card payment failed, email notification failed to send (for a mailing list blast use case), more than 4 invalid login attempts, etc

Note that you must provide the **step number** where this exception will happen (mainly on the system column)

USE CASE DESCRIPTION

The text documentation may be augmented with:

- Related artifacts e.g. User Interface, Prompts and Messages, Business Rules, Class Diagrams, Information Items, Decision tables/trees

RELATED ARTIFACT

RELATED ARTIFACTS

The BRD also contains a “Related Artifacts” section where you can use to provide more information about some points in the use case

- E.g. One of the steps says “System validates eligibility”
- You want to inform what qualifies as eligible → add this in the related artifacts

RELATED ARTIFACTS

Benefits:

- Separate the details from the main use case
- Makes the use case description more complete

3 possible techniques

- Decision Tables
- Decision Tree
- Condition/Response Table

EXAMPLE OF A USE CASE WITH A DECISION TABLE

System use case: **Process Life Insurance Application**

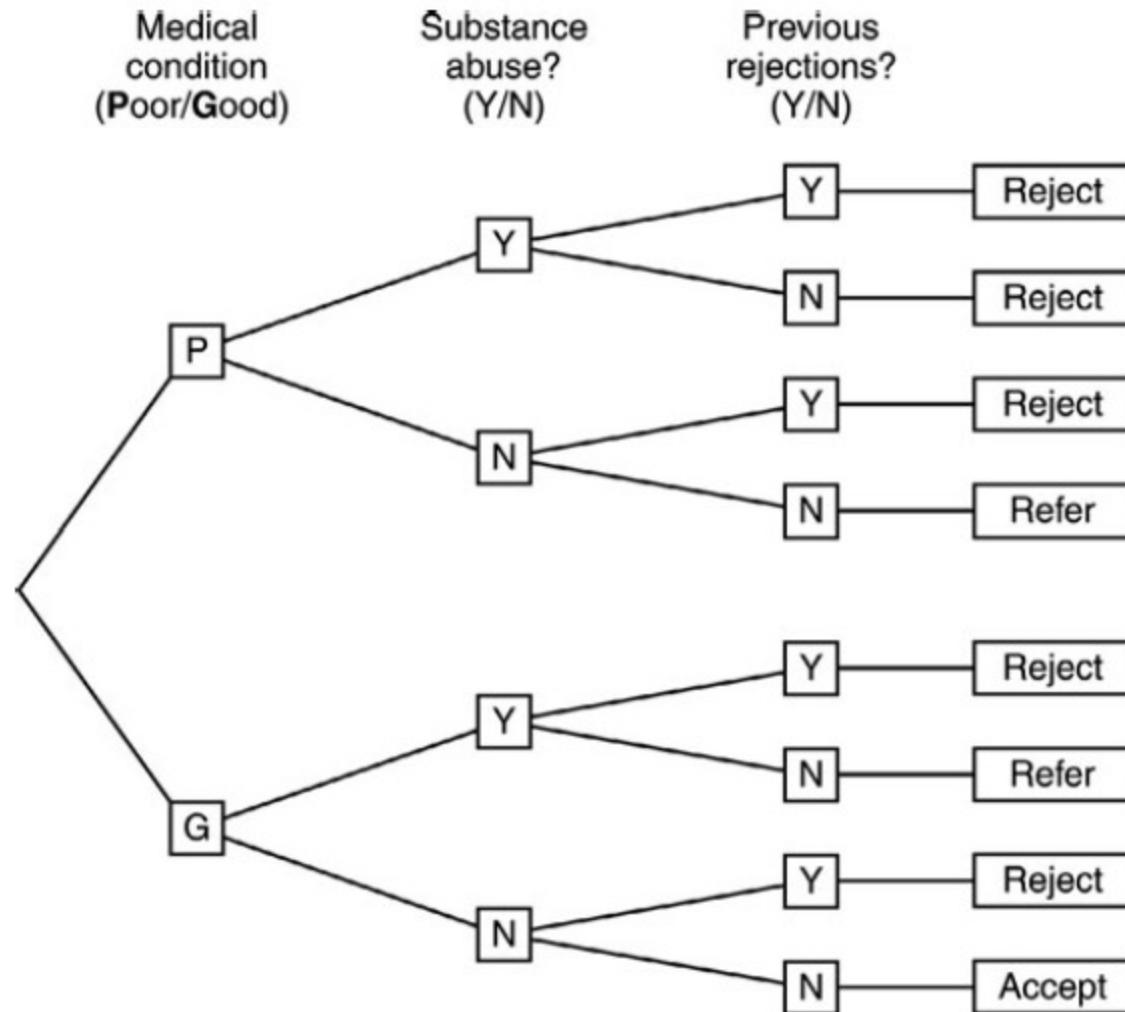
Actor	System
1. Enters application information ...	1.1 Validates eligibility (refer to section 12.1) ...

EXAMPLE OF A USE CASE WITH A DECISION TABLE

12.1 Validate Eligibility Decision Table

		1	2	3	4	5	6	7	8
CONDITION	Medical Condition (Poor/ Good/)	P	P	P	P	G	G	G	G
	Substance Abuse? (Y/N)	Y	Y	N	N	Y	Y	N	N
	Previous Rejections? (Y/N)	Y	N	Y	N	Y	N	Y	N
ACTION	Accept								X
	Reject	X	X	X		X		X	
	Refer				X		X		

DECISION TREES



CONDITION/RESPONSE TABLE

Actor	System
... 10. Enters income details 10.1 Determines tax bracket (see Appendix A) ...

Appendix A:

Tax bracket condition/response table:

Condition	Response
Under minimum	No tax payable
Minimum-\$18,000	Tax bracket A
\$18,000.01-\$60,000	Tax bracket B
\$60,000.01-\$500,000	Tax bracket C
Over \$5000,000	Tax bracket D

ADVANCED USE-CASE FEATURES

Introduction to Analysis Phase

Behavioral Modeling using Use Case Description

Advanced Use-Case Features

Organizing Use Cases

ADVANCED USE-CASE FEATURES

UNIFIED
MODELING
LANGUAGE



- UML also provides some advanced use-case features
- Increases reusability

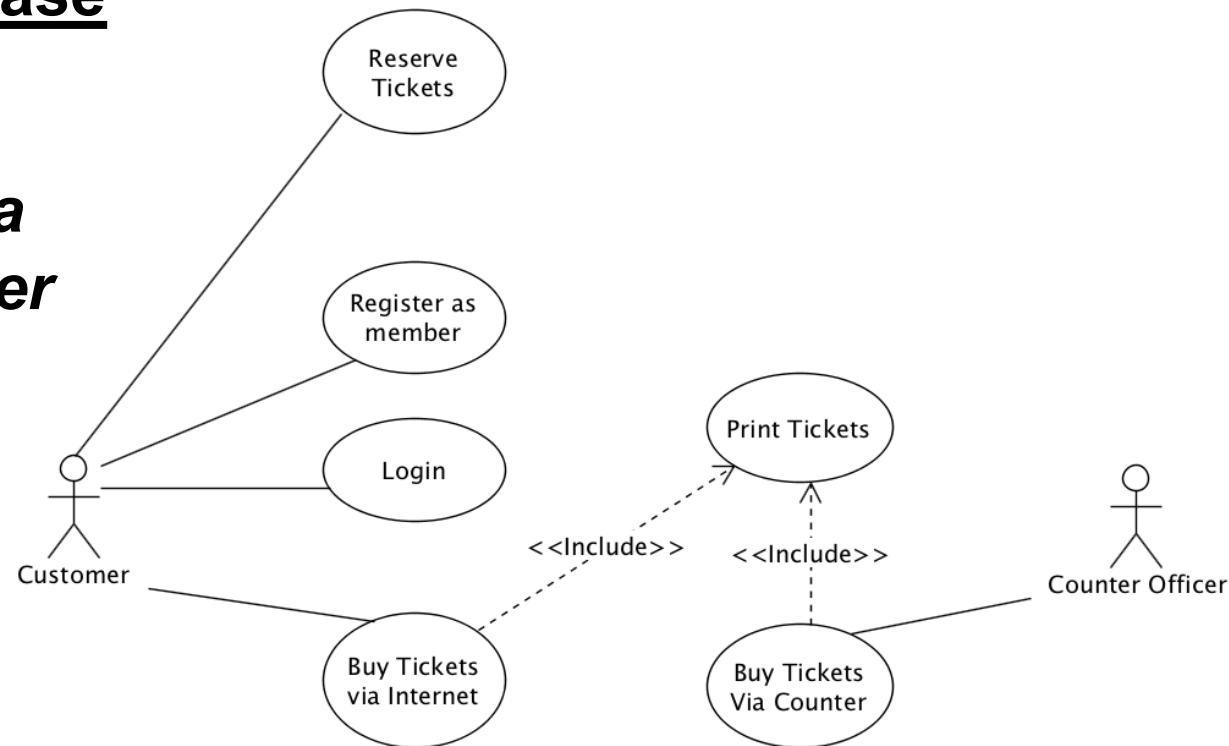
INCLUDE

An including use case invokes the included use case

Buy Tickets Via Internet/Counter

includes

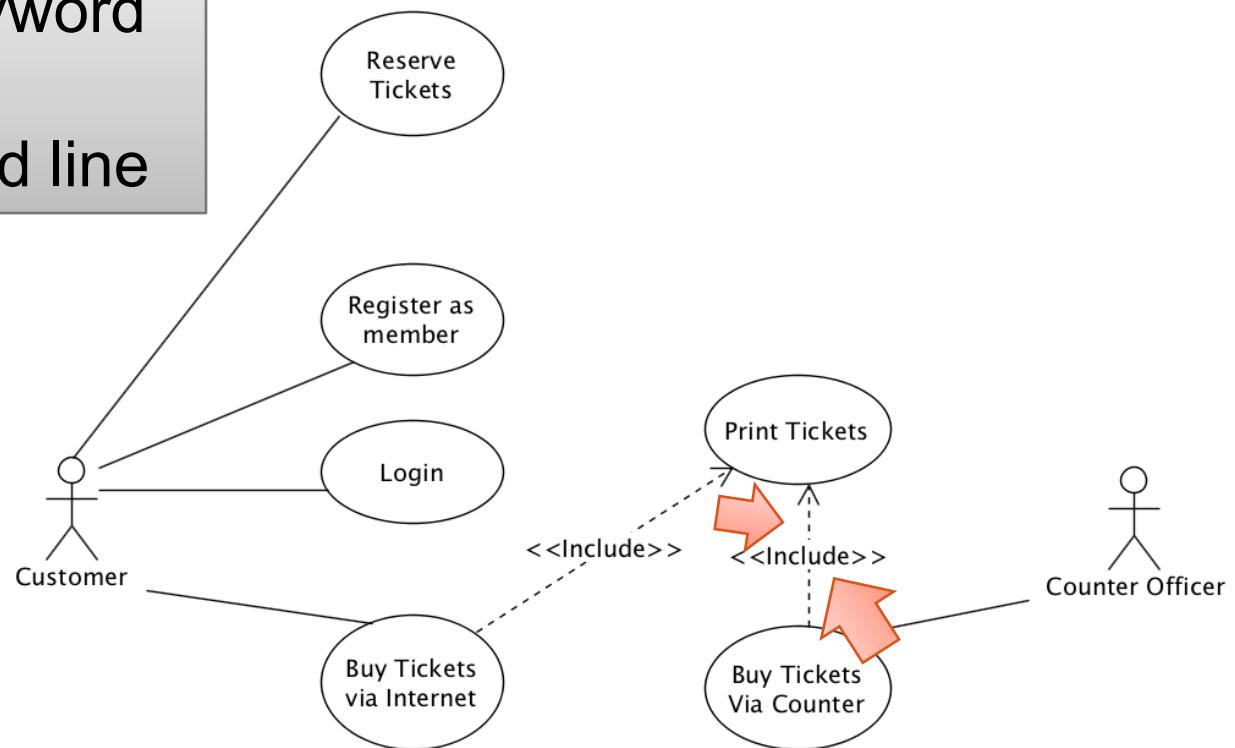
Print Tickets



INCLUDE

<<include>> keyword

Arrow with dotted line

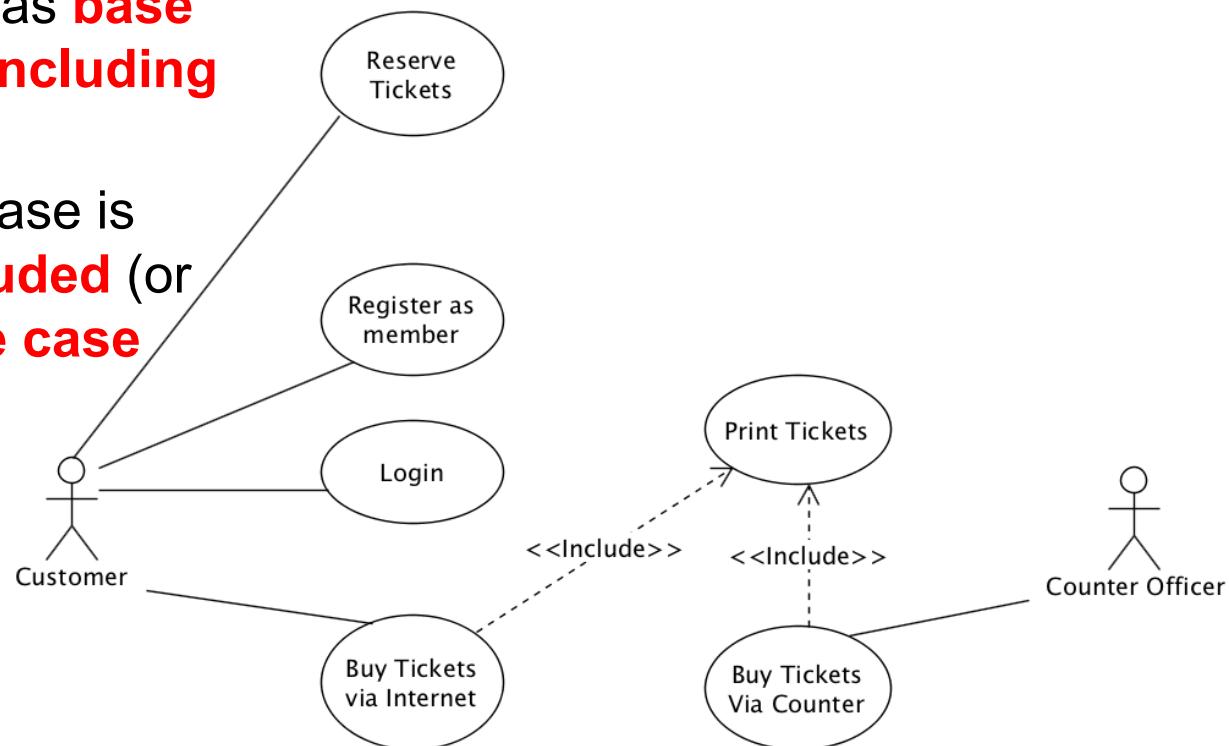


How does this help in reusability?

INCLUDE

Terminology

- The original use cases are referred to as **base use cases** or **including use cases**
- The new use case is called the **included** (or **inclusion**) **use case**

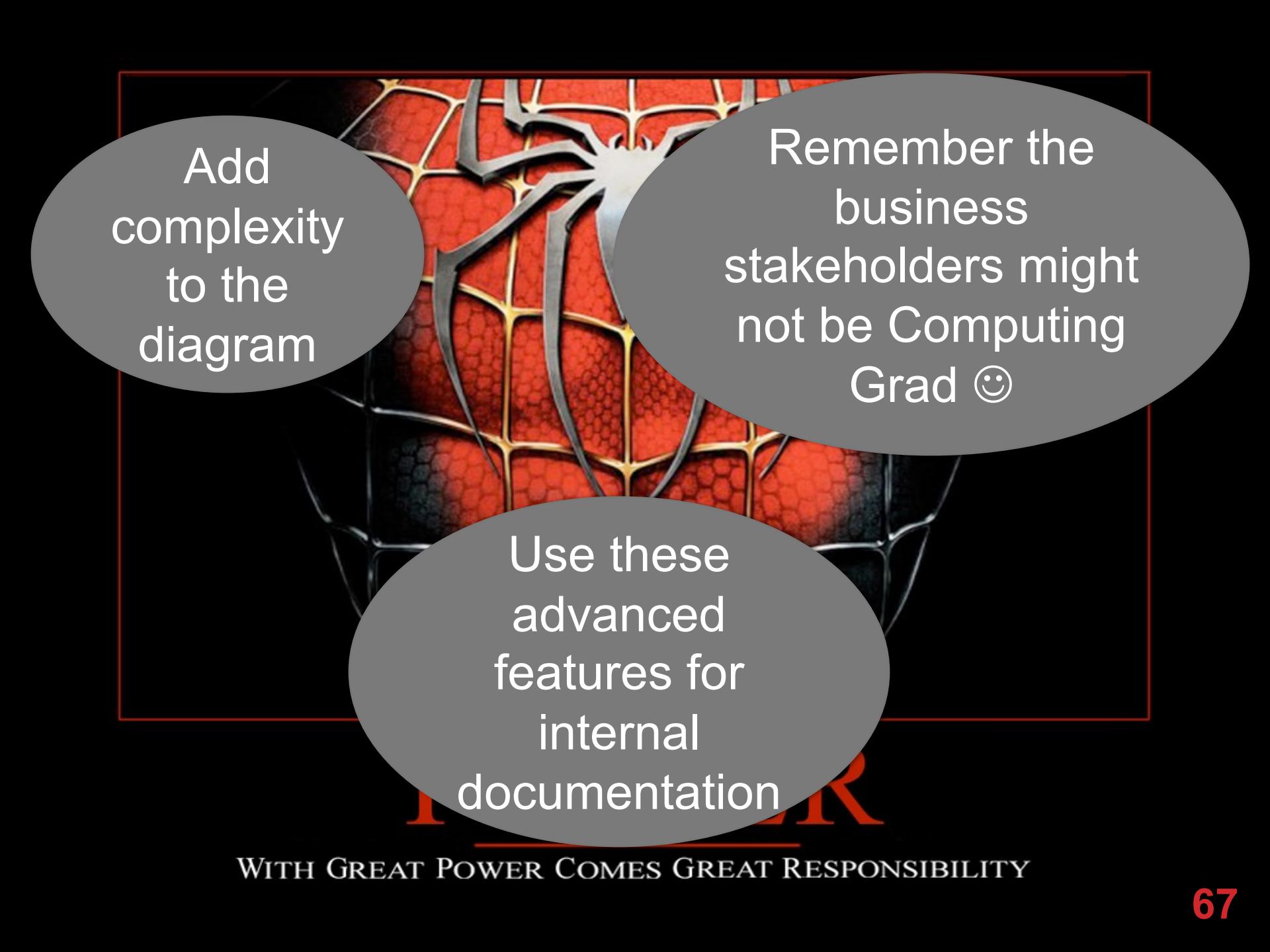


INCLUDE IN USE-CASE DOCUMENTATION

System Use Case: Make Reservation	
Actor	System
1. Selects a trip ...	1.1 Include (Check Available Seats Use Case) ...



System Use Case: Check Available Seats	
Actor	System
	<ol style="list-style-type: none">1. Verifies that seats are available2. Determines and displays the maximum number of seats for the trip3. Determines and displays the number of seats currently reserved ...

A close-up photograph of Spider-Man's suit, showing the red and blue spider-suit pattern and the metallic web-shooters on his wrists. Three grey speech bubbles are overlaid on the image, containing text related to the slide's message.

Add complexity to the diagram

Remember the business stakeholders might not be Computing Grad ☺

Use these advanced features for internal documentation

WITH GREAT POWER COMES GREAT RESPONSIBILITY

ORGANIZING USE CASES

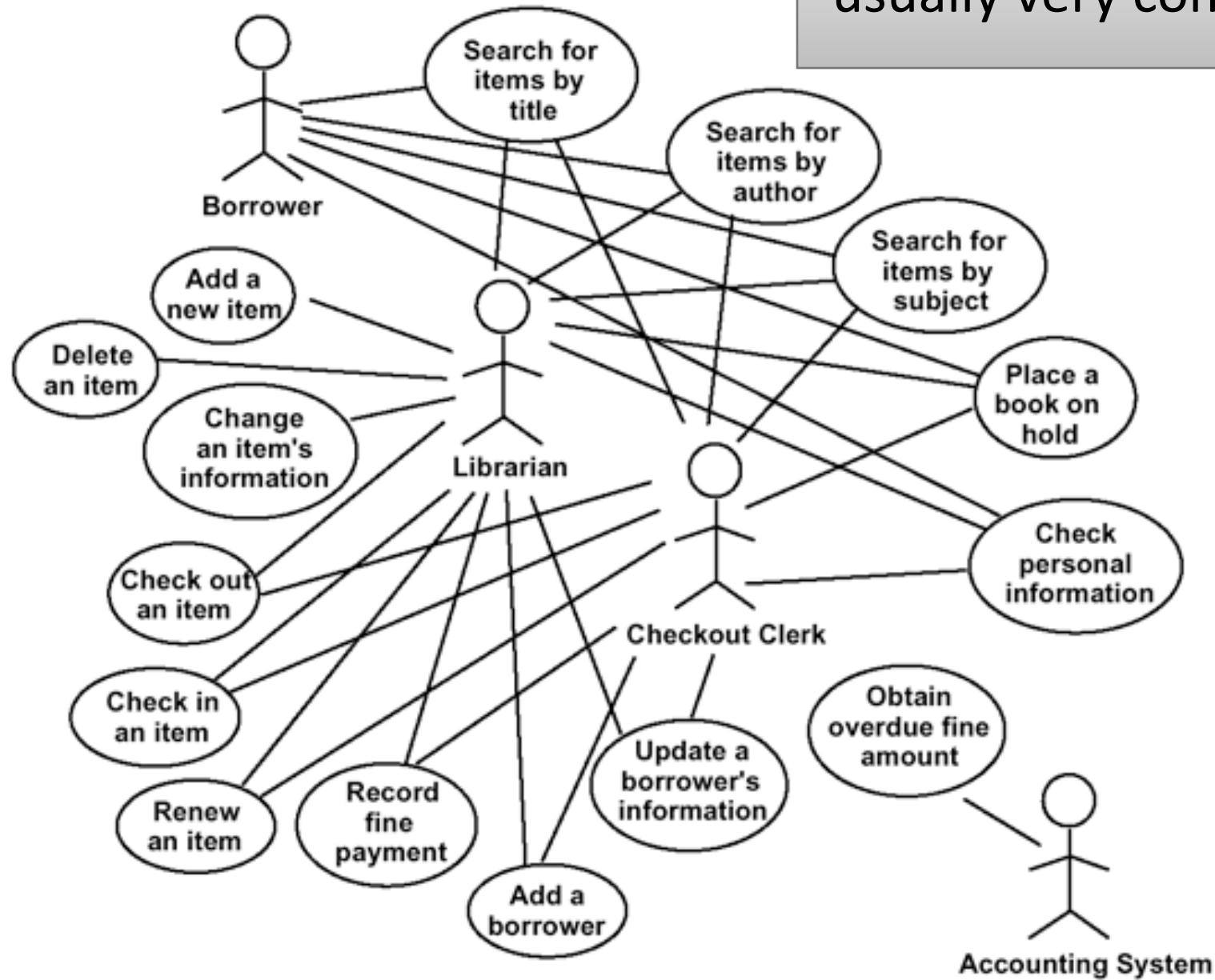
Introduction to
Analysis Phase

Behavioral
Modeling using
Use Case
Description

Advanced Use-
Case Features

Organizing Use
Cases

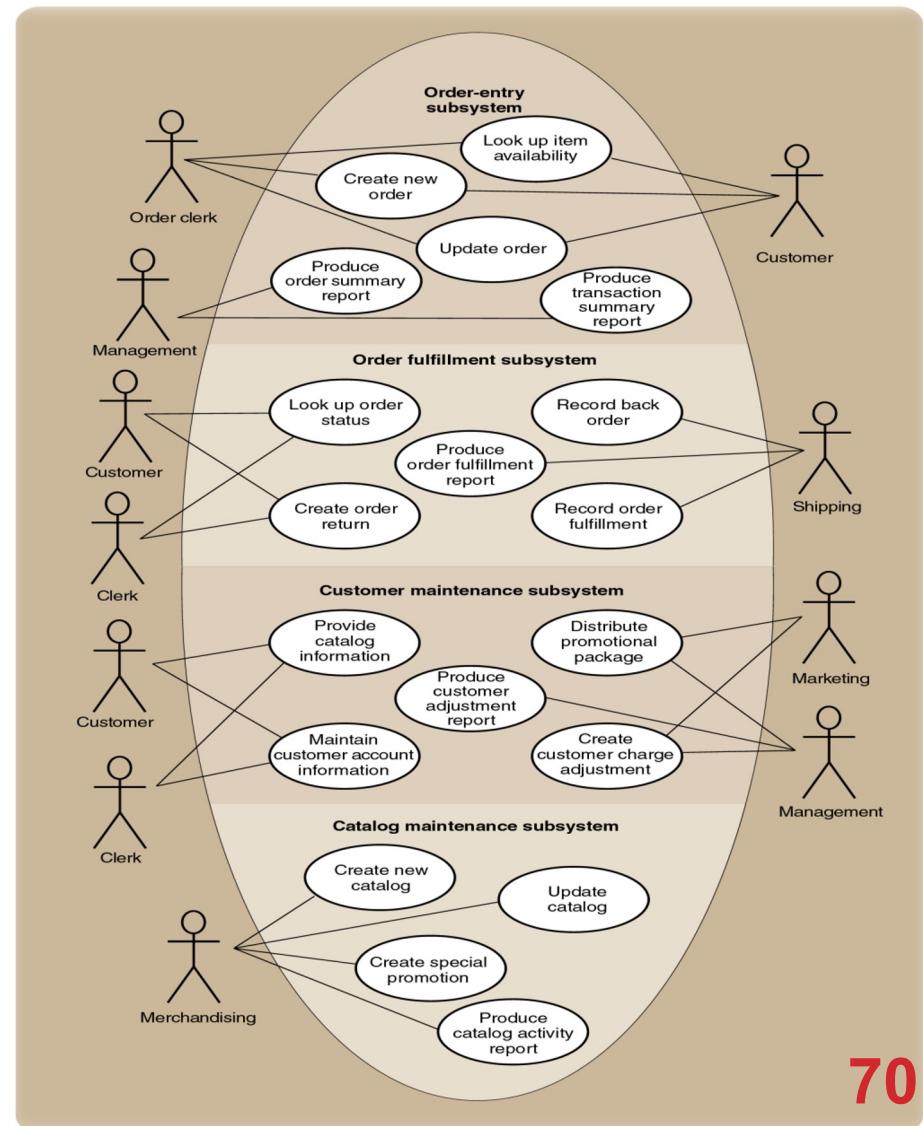
Real-world systems are usually very complicated



ORGANIZING USE CASES

Organize use cases by specific actor

Organize use cases by subsystem



A close-up photograph of a man with dark hair and glasses, wearing a light-colored shirt. He is holding a plain white rectangular card in front of his chest with both hands. His fingers are visible at the edges of the card. The background is a soft-focus green.

It's your turn...

TASK

Use Case Description Exercise

SUMMARY

Analyze behavior of system through behavioral modeling

**Different sections in Use Case Description
(especially the flow of events – behavior of system)**

**Advanced features of use case diagrams (include,
extends, inheritance)**

Organizing use cases

WHAT'S NEXT?

Enterprise System Architecture

LECTURE 5

INTRODUCTION TO

ENTERPRISE SYSTEM

LEK HSIANG HUI

LEARNING OBJECTIVES

At the end of this lecture, you should understand:

- The basics and design of enterprise systems
- The 3 major layers of enterprise systems: data access layer, business logic layer, and presentation layer
- Overview of the most important UML diagram used in the SDLC

BEFORE WE START...

Please note that the 2 major players for Enterprise System Development Framework in the industry is Java Enterprise Edition (JavaEE) and .NET

- They are a lot more established and have a more elaborate structure to support Enterprise System Development
- For this course (since IT5001 was in Python), the code examples given in this lectures will be in Python as far as possible
- But good to understand how such framework works (e.g. JavaEE)

INTRODUCTION TO ENTERPRISE SYSTEM



Introduction to
Enterprise
System

Data Access
Layer

Business Logic
Layer and
Presentation
Layer

UML diagrams

QUESTION TIME



1. After 4 weeks of classes, what are some characteristics of Enterprise system?

ENTERPRISE SYSTEMS

To design and develop such systems requires much planning and resources

- Important to understand that systems development is not just about coding
- But involves a lot of requirements gathering and analysis
- Team working on a project rather than 1 or 2 people
- Need to have a way to “**divide and conquer**”
(divide problem into small tasks)

ENTERPRISE SYSTEMS

The system design and development process needs to be **scalable** and **well-separated**

- Different people working on different things (e.g. analysts designing architecture of system, developers doing coding, designers designing UI, etc)

The design must also be able to accommodate potential future enhancements

- **Systematic** design
- Solid fundamental architecture

ENTERPRISE SYSTEMS

Enterprise systems requirements:

- Able to support multiple (concurrent) users
- Able to interface with other systems (e.g. payment systems)
- Able to support multiple ways to access the service (e.g. browser, smartphone, hardware, etc)
- Robust and scalable
- Able to persist the data even if the application crashes

GENERAL ENTERPRISE SYSTEMS COMPONENTS

Generally, enterprise systems consists of the following components:



Application Server



Web Server(s)



Database Server(s)



Clients

SERVER

The “server” section is divided into multiple servers each in charge of different aspects of the system

Database Server(s): persistent storage of data

Application Server: provides the **business logic** (connects to DB Server)

Web Server(s): consumes the services from the application server to fulfil client requests coming from the web

- Some application server comes with web server capability, so they are combined

DATABASE SERVER

There are different types of databases:

- Relational Database (RDBMS)
- NoSQL

RDBMS is often/traditionally used for enterprise system

Each DB Server can consist of multiple **databases**

- Each database can have multiple **tables**
 - Each table has a **schema** which describes the structure of the data
 - Each record is represented as a row, and its fields are represented as columns

DATABASE

Person Table

id	name	age	contact_num	...
1	John	20	91234567	...
2	Mary	18	97654321	...
...				



Need to have a primary key field that uniquely identifies each record

Person Table Schema

Person

`id: int (primary_key, auto_increment)`
`name: string`
`age: int`
`contact_num: string`

...

DATABASE

Tables can be linked to each other (foreign keys)

Person Table

id	name	age	contact_num	address_id	...
1	John	20	91234567	1	...
2	Mary	18	97654321	2	...
...					

Address Table

id	address	postal
1	Blk 322 Clementi Avenue 5	120322
2	Blk 538 Pasir Ris Street 51	510538
...		

APPLICATION SERVER

Contains the **business logic**

- Algorithms/codes to address a business task
- E.g. Capture the list of staff
- Prevent access to the system by using login mechanism
- etc

Connects to DB Server:

- To access data
- To insert/update/delete data

WEB SERVER

If clients access the services through the web, there should be a web server

- Takes in request from the user (e.g. web browser)
- Connects to the application server and call the relevant services
- Application server returns the results to the web server
- Web server displays the result in a properly formatted manner back to the user

CLIENT

The client provides the interface to use the services provided by the server

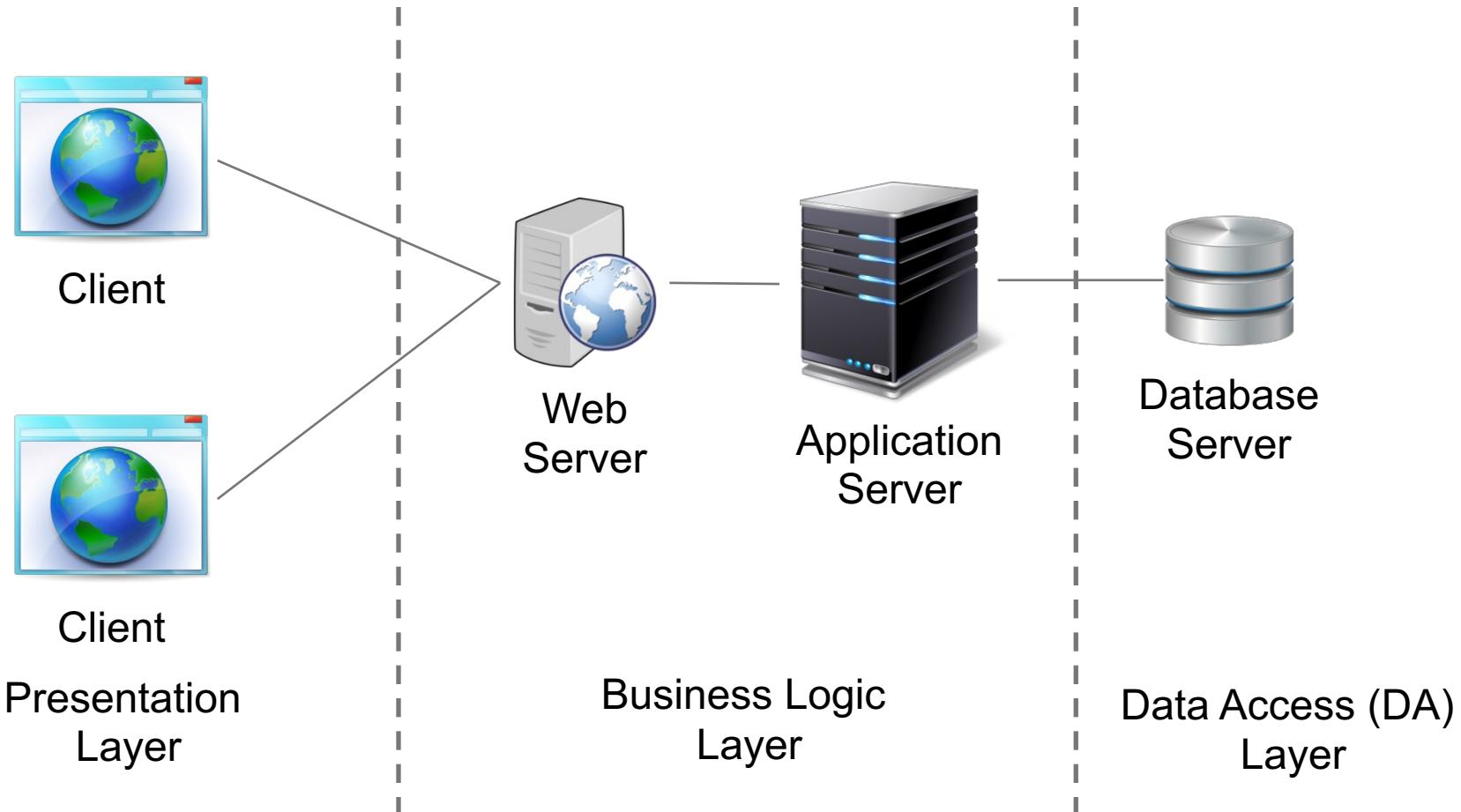
It can talk directly to the application server or the web server

- Web browser talks to the web server
- ATM connects to an application server securely to process transactions

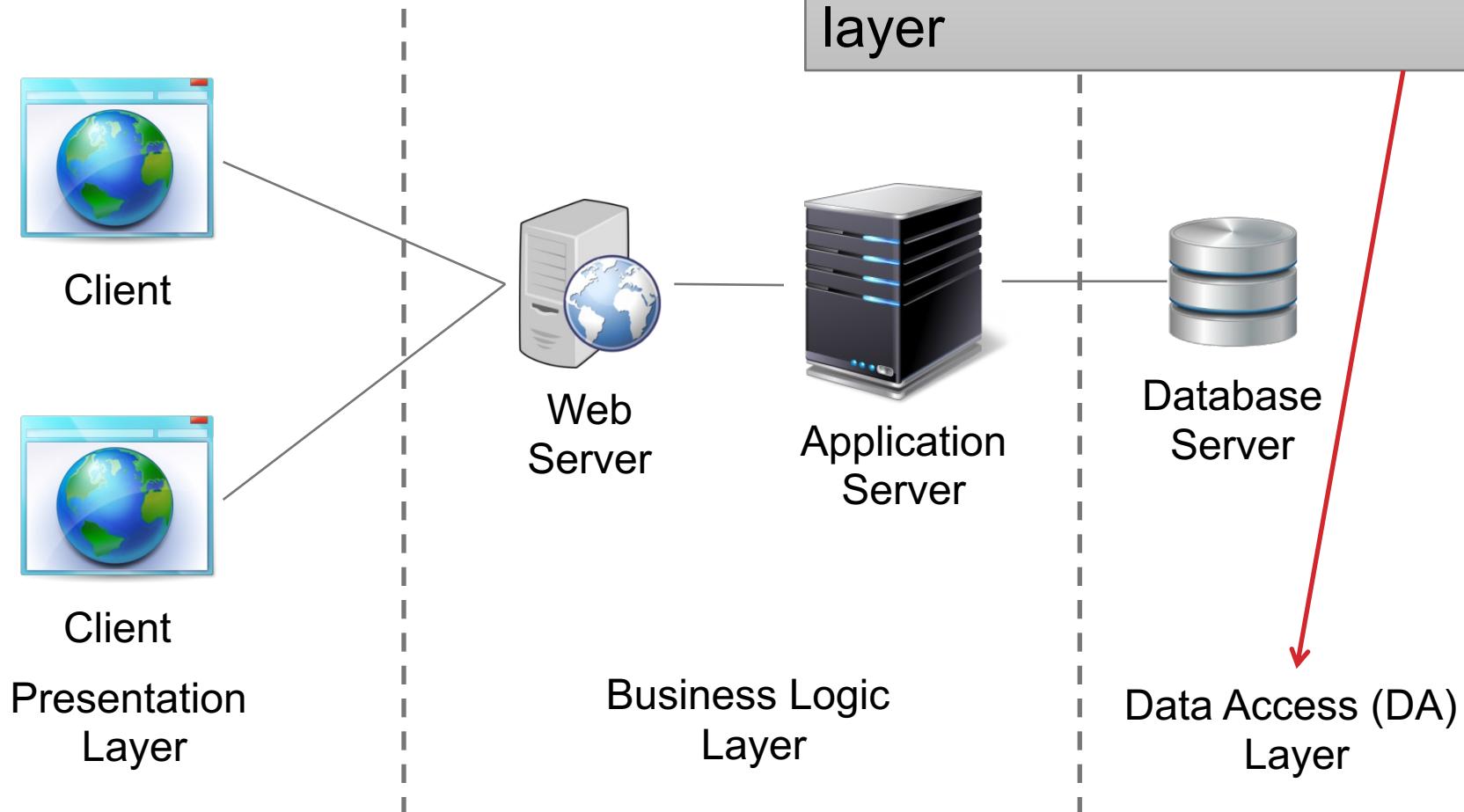
Servers can also become a “client” to another server

- E-commerce system connects to PayPal server to verify credit card details

MULTI-TIER / MULTI-LAYER ARCHITECTURE



MULTI-TIER / MULTI-LAYER ARCHITECTURE



DATA ACCESS LAYER



Introduction to
Enterprise
System

Data Access
Layer

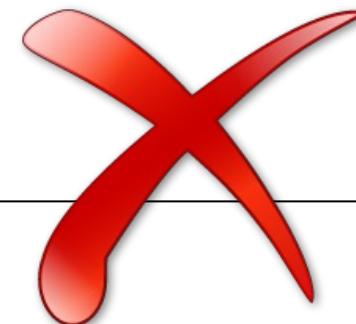
Business Logic
Layer and
Presentation
Layer

UML diagrams

DA LAYER

```
...  
# Connect to db  
conn = sqlite3.connect(...)  
  
cursor = db.cursor()  
  
# Execute SQL select statement  
cursor.execute("SELECT * FROM person")  
  
#list of person  
persons = []  
  
# Get and display one row at a time  
rows = cursor.fetchall()  
for row in rows:  
    person = Person(row[0], row[1], ...)  
    persons.append(person)  
  
# Close the connection  
db.close()
```

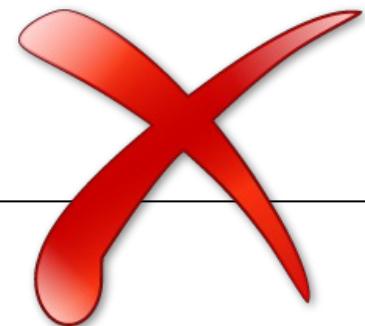
Most of the operations would need DB access
- Imagine the amount of repeated codes!



DA LAYER

```
...  
# Connect to db  
conn = sqlite3.connect(...)  
  
cursor = db.cursor()  
  
# Execute SQL select statement  
cursor.execute("SELECT * FROM person")  
  
#list of person  
persons = []  
  
# Get and display one row at a time  
rows = cursor.fetchall()  
for row in rows:  
    person = Person(row[0], row[1], ...)  
    persons.append(person)  
  
# Close the connection  
db.close()
```

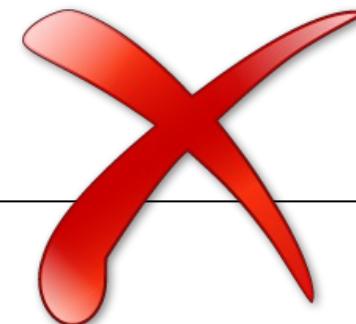
Furthermore, imagine
the amount of
conversion of
database records to
objects!



DA LAYER

```
...  
# Connect to db  
conn = sqlite3.connect(...)  
  
cursor = db.cursor()  
  
# Execute SQL select statement  
cursor.execute("SELECT * FROM person")  
  
#list of person  
persons = []  
  
# Get and display one row at a time  
rows = cursor.fetchall()  
for row in rows:  
    person = Person(row[0], row[1], ...)  
    persons.append(person)  
  
# Close the connection  
db.close()
```

If we had to change
the database
connection
configuration later on,
it will be a disaster!



SIMPLE DATABASE DEMO



**1. If you want to follow along,
download L5_sample_codes.zip**

DA LAYER

Systematic:

- The application (different methods) does not just issue database connect commands and SQL statements each time when there is a need to access the DB
- Instead, 1 or more Data Access (DA) classes are created for interacting with the DB
- Any methods that require DB access goes through these DA classes

DA LAYER

The use of Object-Relational Mapping (ORM) frameworks is preferred

- Convert data from DB to objects and objects to DB records automatically
- Programmers only need to deal with objects
- No more concept of foreign keys, join tables, VARCHAR(...), SQL statements, MySQL, SQLServer, etc

DA LAYER EXAMPLE (DJANGO)

```
from django.db import models

# define the Person model
class Person(models.Model):
    first = models.CharField(max_length=60)
    last = models.CharField(max_length=60)

# create a person object
p = Person(first="John", last="Doe")

# call the save() method to save the person to db
p.save()
```

DA LAYER EXAMPLE (JAVAEE)

```
// DA class
public class DAHelper{
    EntityManager em;
    ...
    public List<Person> searchPerson(String name) {
        Query q = em.createQuery("SELECT p FROM PERSON p
where p.name = :val");
        q.setParameter("val", name);
        return q.getResultList();
    }
}
```

```
DAHelper daHelper = ...;

// search for person by name
// return a list of person
public ArrayList<Person> search(String name) {
    return daHelper.searchPerson(name);
}
```

DA LAYER EXAMPLE

All database access (insert, search, delete, update) are handled by the framework (DA layer)

- Conversion of objects to database records
- Retrieval of database records to objects
- Developers for these classes only need to focus on the business logic and assume that the DA layer does its job

Achieve better separation of concerns

ORM EXAMPLE

id	name	age	contact_num	address_id	...
1	John	20	91234567	1	...
...					

id	address	postal
1	Blk 322 Clementi Avenue 5	120322
...		



```
class Person {  
    private long id;  
    private String name;  
    private int age;  
    private String contactNumber;  
    private Address address;  
    ...  
}
```

```
class Address{  
    private long id;  
    private String address;  
    private String postal;  
}
```

ORM EXAMPLE

id	name	age	contact_num	address_id	...
1	John	20	91234567	1	...
...					

person_id	address_id
1	1
...	

id	address	postal
1	Blk 322 Clementi Avenue 5	120322
...		

```
class Person {  
    private long id;  
    private String name;  
    private int age;  
    private String contactNumber;  
    private Address[] addresses;  
    ...  
}
```



```
class Address{  
    private long id;  
    private String address;  
    private String postal;  
}
```

BUSINESS LOGIC LAYER AND PRESENTATION LAYER



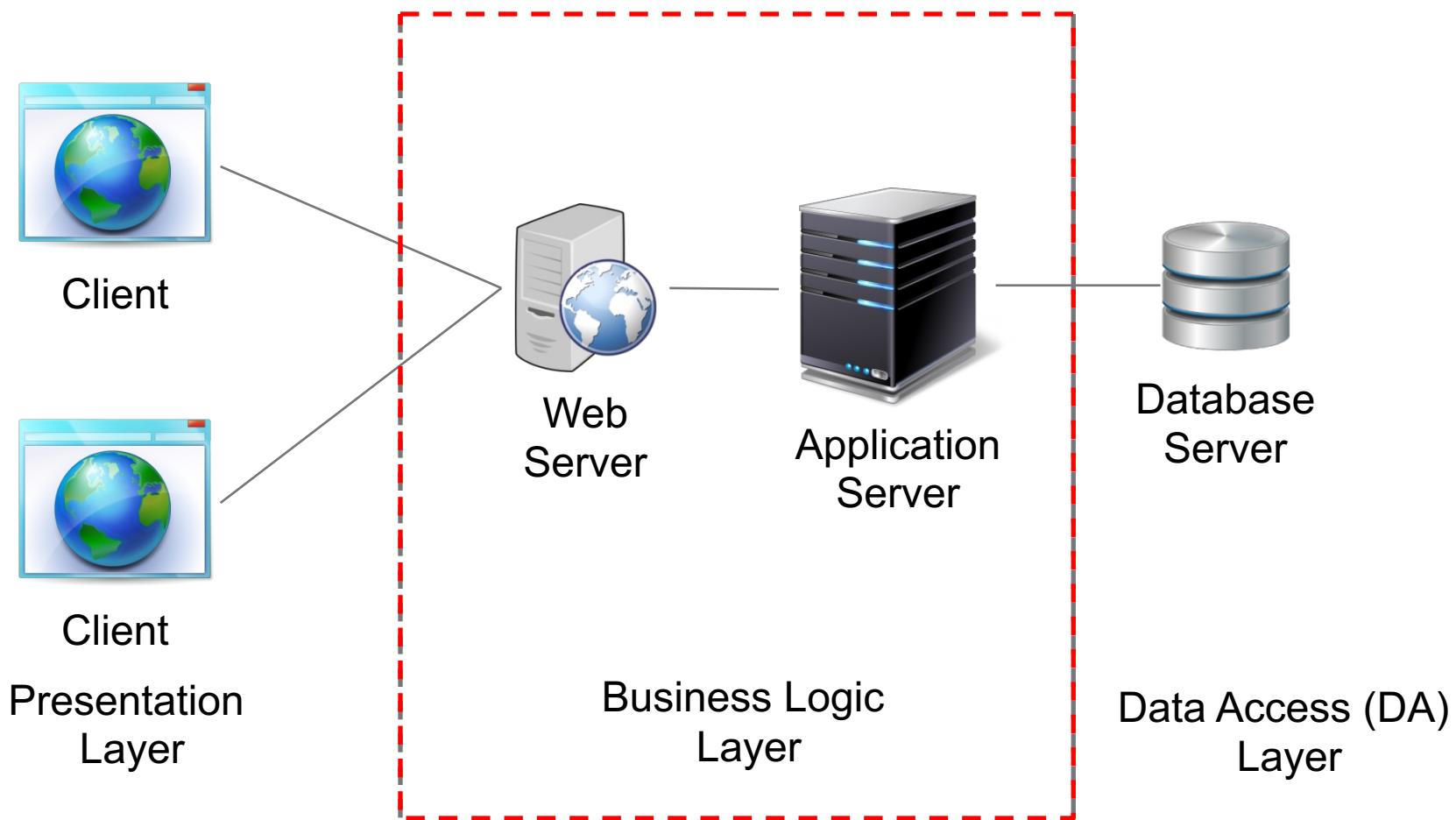
Introduction to
Enterprise
System

Data Access
Layer

Business Logic
Layer and
Presentation
Layer

UML diagrams

MULTI-TIER ARCHITECTURE



BUSINESS LOGIC LAYER

Receives request from the client

Fetches relevant data using the DA layer

Does computation on the data

Sends data to the presentation layer

PRESENTATION LAYER

Receives the raw data from business logic layer

Presents the data in a readable format

Provides user interface:

- User interacts with the system through the presentation layer
- Any request to do something is then directed to the business logic layer
- It often does not much logic beyond data validation

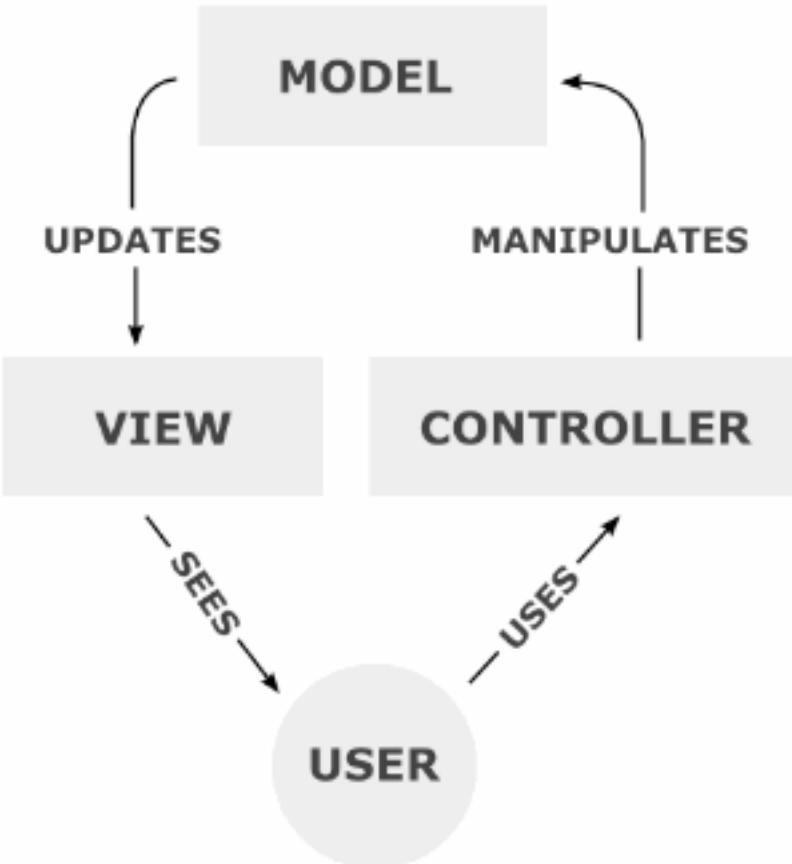
DESIGN PATTERNS

There are various ways to design the business logic layer and presentation layer

Software design patterns are general solution to a common problem

- Problem: how to have a clean design which separates the requests logic, display logic, business logic
- Web-based applications often use the **Model-View-Controller** design pattern

MODEL/VIEW/CONTROLLER (MVC)



Model : Domain object (updates the view)
View : Presentation
Controller : receives commands from user and invoke the model's methods (direct traffic)

One of the most popular design pattern nowadays, commonly used for web applications

UML DIAGRAMS

Introduction to
Enterprise
System

Data Access
Layer

Business Logic
Layer and
Presentation
Layer

UML diagrams

UML

Apart from **activity diagram** and **use case diagram**, UML also defines other diagrams useful for SDLC

2 main categories:

- Structural diagrams
 - Class diagram
 - Package diagram
- Behavioral diagrams
 - Activity diagram
 - Use case diagram
 - State Machine diagram / statechart
 - Sequence diagram

Here are the
list of the most
useful
diagrams

SUMMARY

Enterprise systems follows a server/client architecture

When designing the systems, it is important to “divide conquer” and try to achieve separation of concerns

For example, database access should be separated into a tier/layer

Likewise for business logic and presentation logic

WHAT'S NEXT?

Data Modeling

LECTURE 6

DATA MODELING

LEK HSIANG HUI

LEARNING OBJECTIVES

At the end of this lecture you should understand:

- What structural modeling is
- How to perform structural modeling using domain model class diagram

INTRODUCTION TO STRUCTURAL MODELING

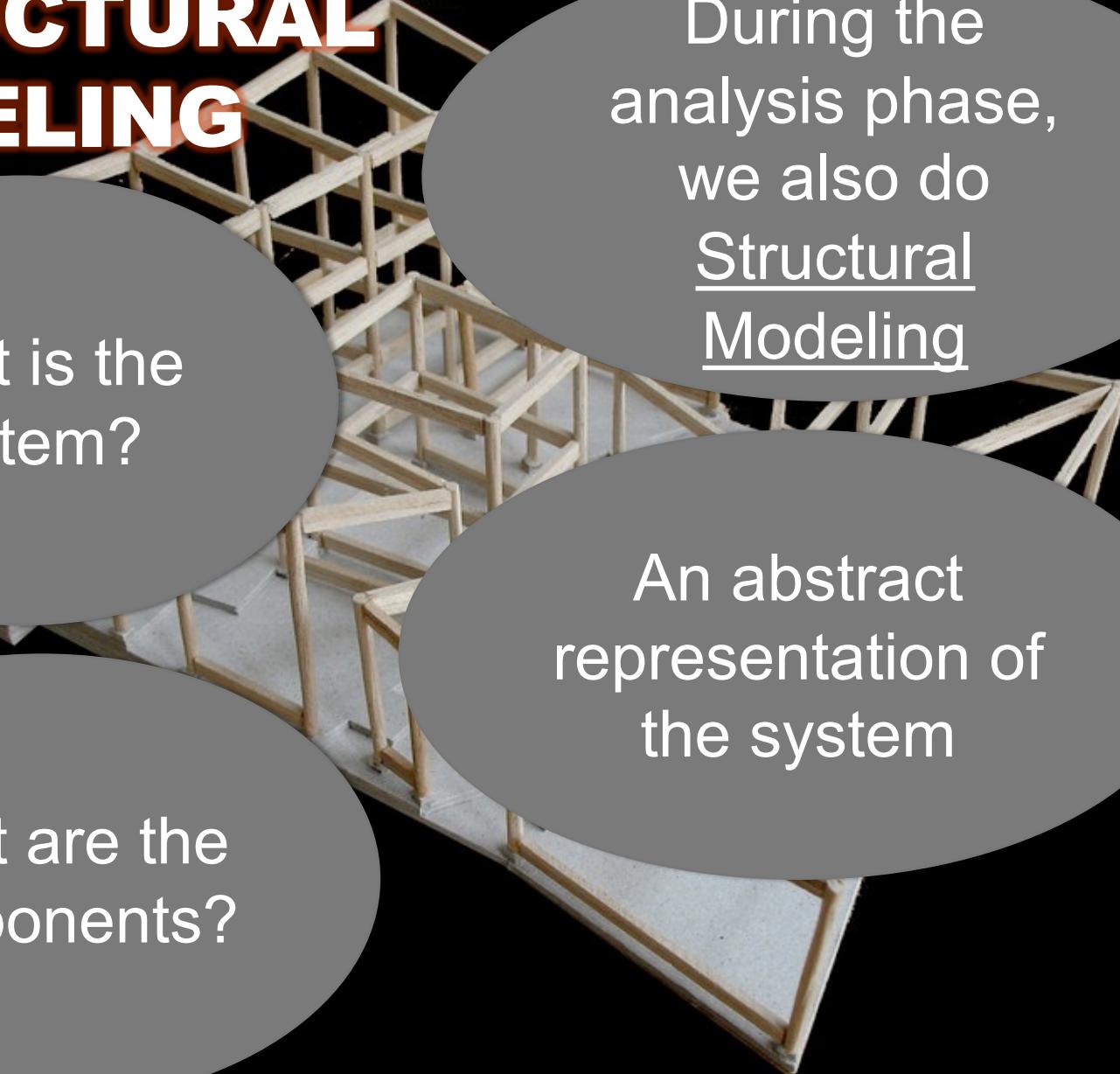
Introduction
to Structural
Modeling

Introduction
to Entity
Class

Class
Diagram
Notation

Domain
Model Class
Diagram

STRUCTURAL MODELING



What is the system?

What are the components?

During the analysis phase, we also do Structural Modeling

An abstract representation of the system

STRUCTURAL MODEL

Main diagram = class diagram

Will be performing object-oriented structural analysis:

- Attributes (of business objects)
- Operations (of business objects)
- Numerical relationships (between business objects)
 - E.g. How many customers may co-own a particular account

Focuses on the “nouns**” of the system**

STRUCTURAL MODEL

Ensures internal consistency within the BRD

Each use case description is verified against the structural model

STEPS IN STRUCTURAL ANALYSIS

- 1. Identify Entity Classes and Add Attributes**
- 2. Model Generalizations**
- 3. Model Whole/Part Relationship**
- 4. Analyze Associations**
- 5. Analyze Multiplicity**

INTRODUCTION TO ENTITY CLASS



Introduction
to Structural
Modeling

Introduction
to Entity
Class

Class
Diagram
Notation

Domain
Model Class
Diagram

STEPS IN STRUCTURAL ANALYSIS



1. Identify Entity Classes
and add Attributes

WHAT TO KEEP TRACK?



If you are building an e-commerce website, what do you need to keep track/record?

List of transactions?

List of Members?

ENTITY CLASSES

Entity Classes
= Business objects that
need to be
recorded

Transaction,
Member are
examples of
Entity Classes

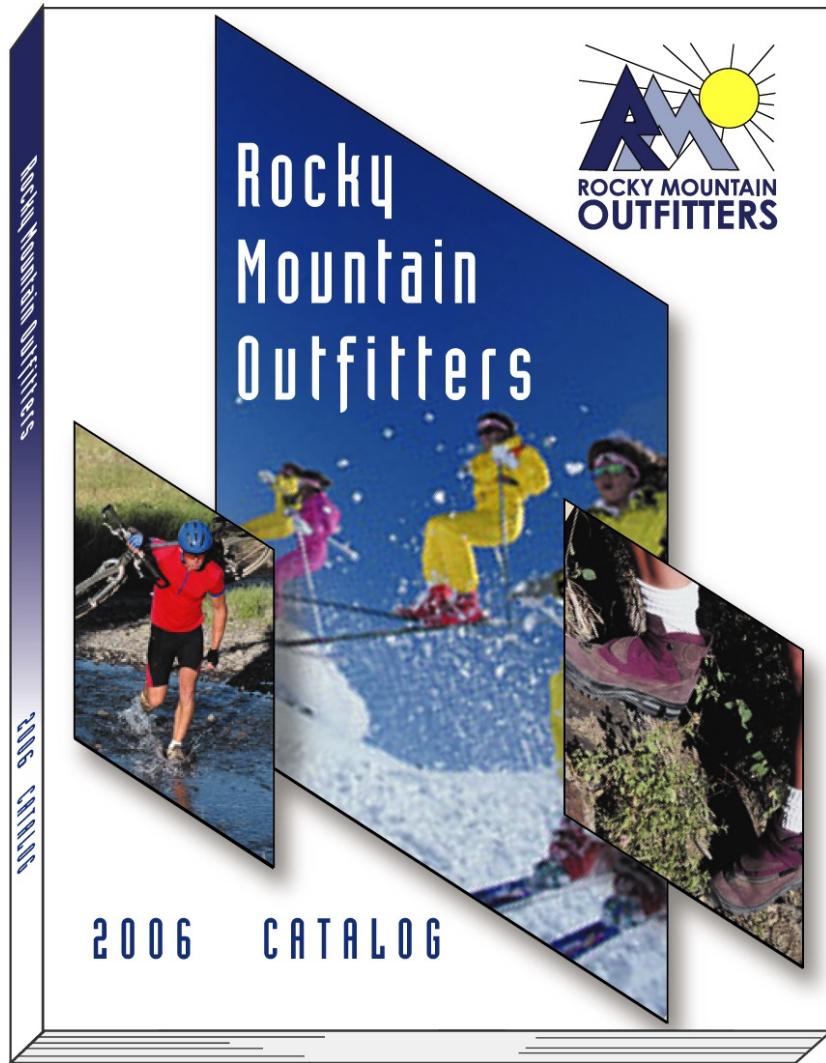
Need to
understand
the problem
domain

PROCEDURE FOR DEVELOPING AN INITIAL LIST OF THINGS

Review the **use case documentation** and **human-interface requirements**

- Any **noun phrase** appearing in these are potential entities
- E.g. Member, Transaction, Item, Person, Address, etc

ROCKY MOUNTAIN OUTFITTER CASE STUDY



Rocky Mountain Outfitters (RMO) is a sports clothing distributor

- Significant regional distributor
- Employs more than 600 people
- \$100 million annual sales

PARTIAL LIST OF “THINGS” BASED ON NOUNS FOR RMO

IDENTIFIED NOUN	NOTES ON INCLUDING NOUN AS A THING TO STORE
Accounting	We know who they are. No need to store it.
Back order	A special type of order? Or a value of order status? Research.
Back-order information	An output that can be produced from other information.
Bank	Only one of them. No need to store.
Catalog	Yes, need to recall them, for different seasons and years. Include.
Catalog activity reports	An output that can be produced from other information. Not stored.
Catalog details	Same as catalog? Or the same as product items in the catalog? Research.
Change request	An input resulting in remembering changes to an order.
Charge adjustment	An input resulting in a transaction.

PARTIAL LIST OF “THINGS” BASED ON NOUNS FOR RMO

IDENTIFIED NOUN	NOTES ON INCLUDING NOUN AS A THING TO STORE
Color	One piece of information about a product item.
Confirmation	An output produced from other information. Not stored.
Credit card information	Part of an order? Or part of customer information? Research.
Customer	Yes, a key thing with lots of details required. Include.
Customer account	Possibly required if an RMO payment plan is included. Research.
Fulfillment reports	An output produced from information about shipments. Not stored.
Inventory quantity	One piece of information about a product item. Research.
Product item	Yes, what RMO includes in a catalog and sells. Include.
Management	We know who they are. No need to store.
Marketing	We know who they are. No need to store.

PARTIAL LIST OF “THINGS” BASED ON NOUNS FOR RMO

IDENTIFIED NOUN	NOTES ON INCLUDING NOUN AS A THING TO STORE
Merchandising	We know who they are. No need to store.
Order	Yes, a key system responsibility. Include.
Payment method	Part of an order. Research.
Price	Part of a product item. Research.
Promotional materials	An output? Or documents stored outside the scope? Research.
Prospective customer	Possibly same as customer. Research.
Return	Yes, the opposite of an order. Include.
Return confirmation	An output produced from information about a return. Not stored.
RMO	There is only one of these! No need to store.
Season	Part of a catalog? Or is there more to it? Research.
Shipment	Yes, a key thing to track. Include.

(continued)

ATTRIBUTES OF THINGS

Specific details of things are called **attributes**

Identifier (key): attribute uniquely identifying thing

- Examples: vehicle ID number, or product ID number

ALL CUSTOMERS HAVE THESE ATTRIBUTES:	EACH CUSTOMER HAS A VALUE FOR EACH ATTRIBUTE:		
Customer ID	101	102	103
First name	John	Mary	Bill
Last name	Smith	Jones	Casper
Home phone	555-9182	423-1298	874-1297
Work phone	555-3425	423-3419	874-8546

CLASS DIAGRAM NOTATION

Introduction
to Structural
Modeling

Introduction
to Entity
Class

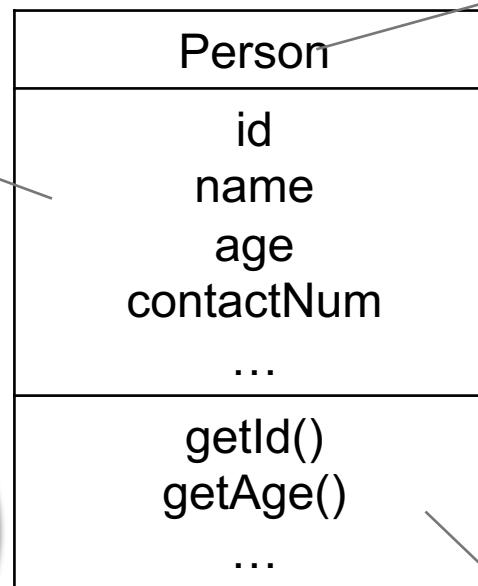
Class
Diagram
Notation

Domain
Model Class
Diagram

NAMING CONVENTIONS

Person Table

id	name	age	contact_num	...
1	John	20	91234567	...
2	Mary	18	97654321	...
...				



Attribute list

Class name

For all the names,
make sure you follow
how you would do it in
the sourcecodes
(e.g. no spaces)

Name class with a
singular noun
(e.g. Student
instead of
Students)

Method list

INTERVIEW QUESTIONS FOR FINDING CLASSES

What types of people and organizations do the system keep track of?

- Example: Customer, CardHolder, and BoardMember

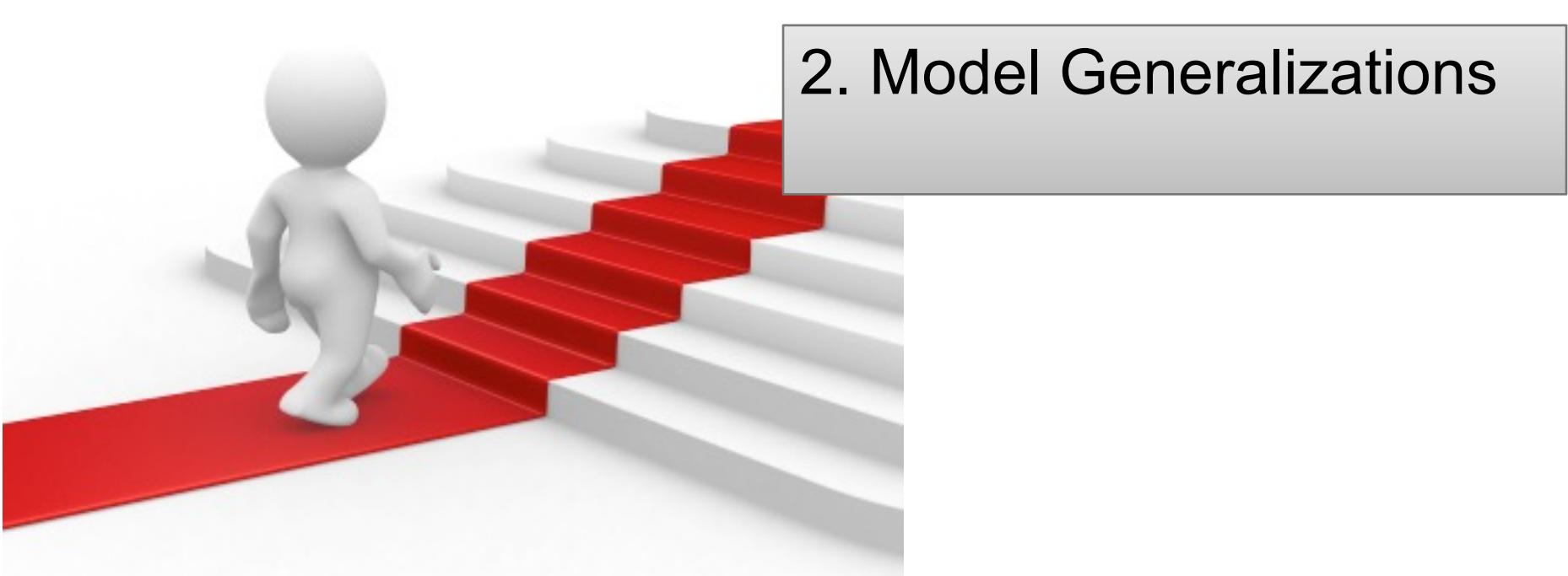
What events and transactions does the system keep a record of?

- Example: Promotion, Sale

What products and services does the system keep a record of?

- Example: Products (keep the stock count for accounting)

STEPS IN STRUCTURAL ANALYSIS



RECALL: GENERALIZATION AND SPECIALIZATION

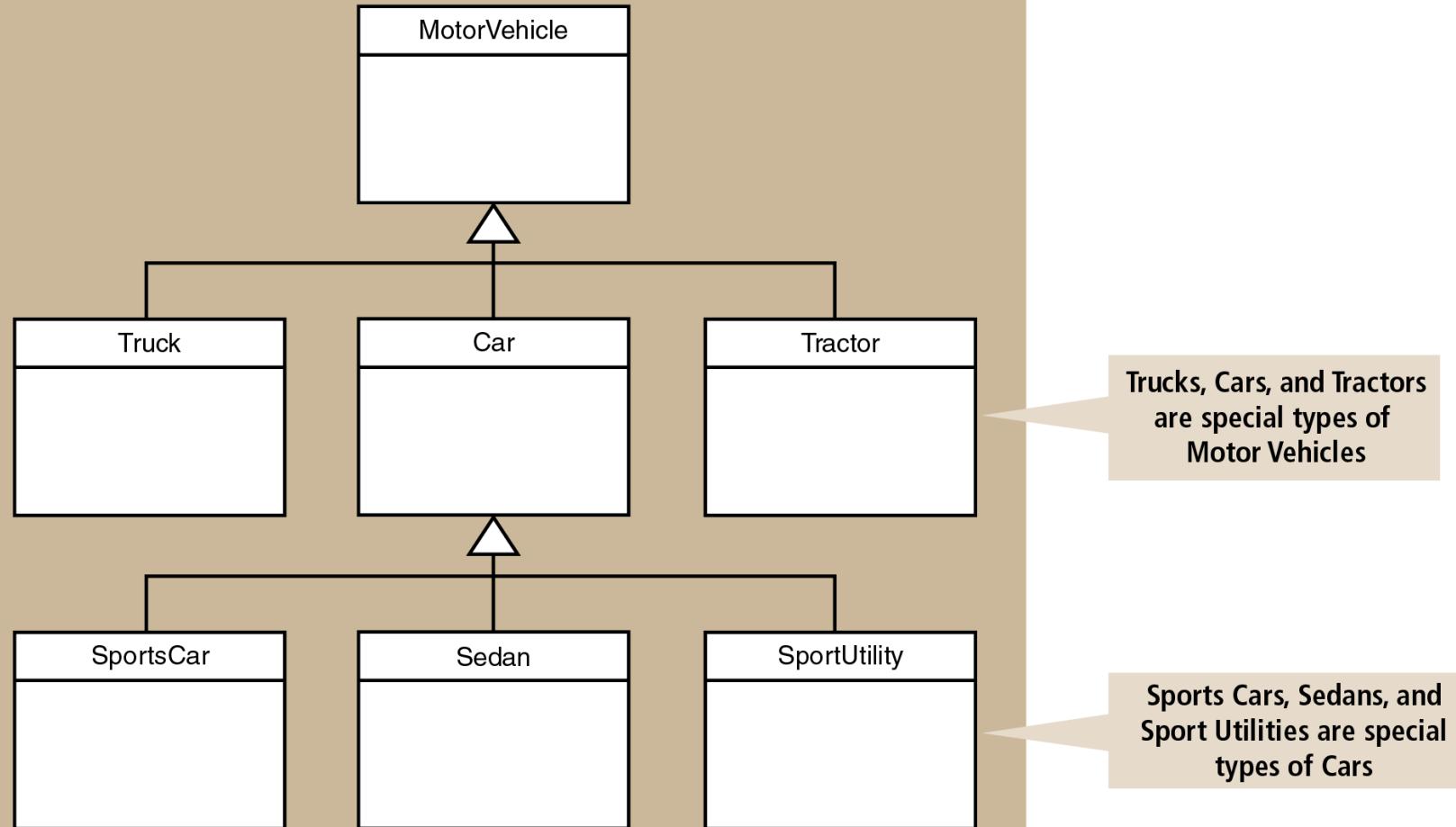
Specialized object inherit the ability to do all the things that a generalized object can do

Think Object-Oriented programming

Clerk extends (inherits) Employee

- Employee = Generalized actor (Superclass)
- Clerk = Specialized actor (Subclass)

A GENERALIZATION/SPECIALIZATION HIERARCHY FOR MOTOR VEHICLES



STEPS IN STRUCTURAL ANALYSIS



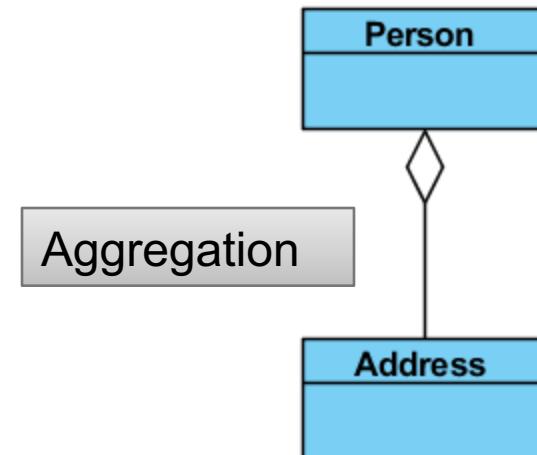
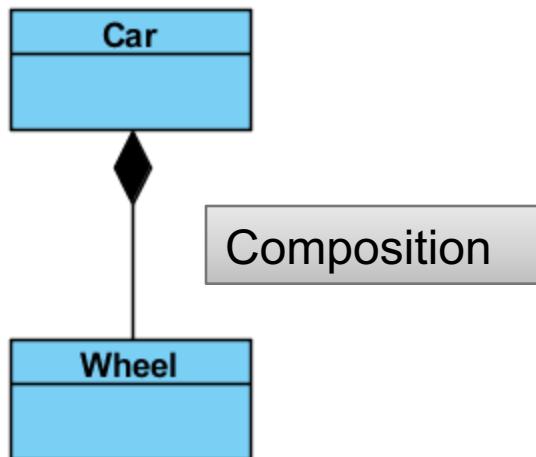
3. Model whole/part relationship

MODEL WHOLE/PART RELATIONSHIPS

Some objects consist of other objects

Model these relationships using
composition and **aggregation**

- Composition and Aggregation describe the relationship between a whole and its parts

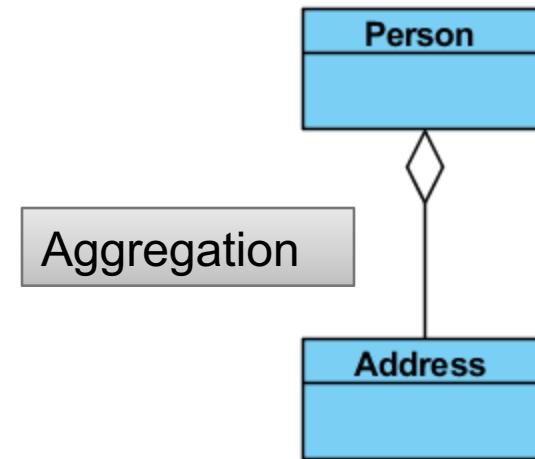
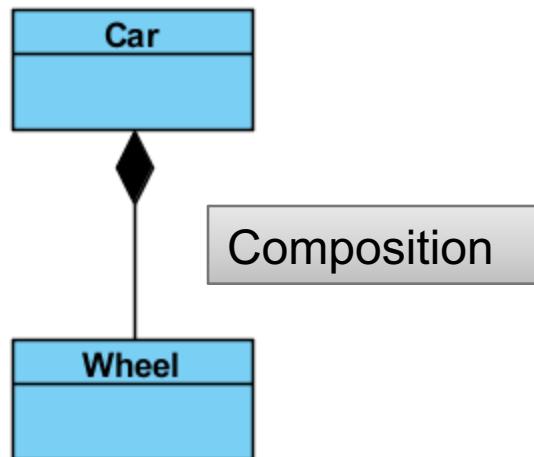


MODEL WHOLE/PART RELATIONSHIPS

Composition means that the whole owns the part entirely

- the part may not belong simultaneously to any other whole

Aggregation means that there is some kind of whole/part relationship

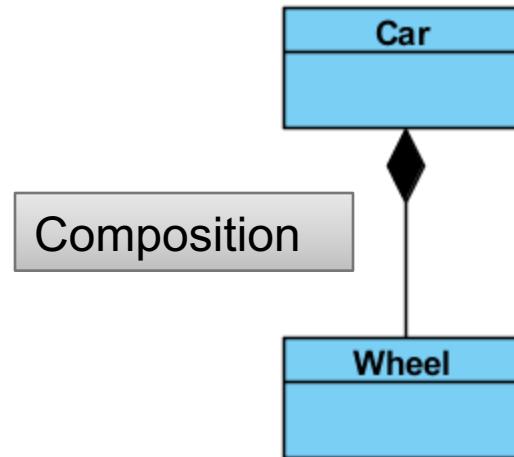


COMPOSITION

Car has one (or more) Wheel(s)

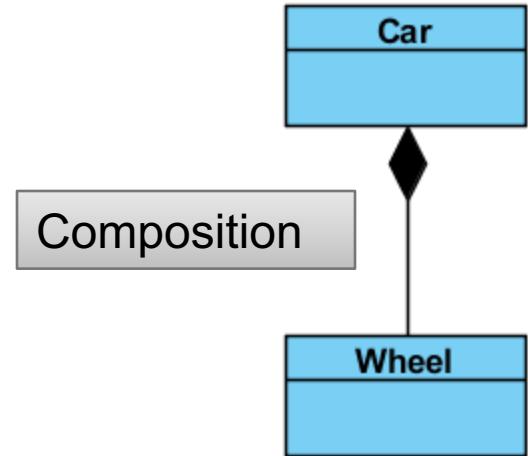
When a Car object is created, the associated Wheel objects are created (together with the Car)

When a Car object is destroyed, the associated Wheel objects are also destroyed



COMPOSITION

```
class Wheel:  
    def __init__(self, num):  
        self.num = num  
  
    def __str__(self):  
        return repr("Wheel " + str(self.num))
```



COMPOSITION

```
from wheel import Wheel
```

```
class Car:
```

```
    wheels = []
```

```
    def __init__(self):
```

```
        self.wheels.append(Wheel(1))
```

```
        self.wheels.append(Wheel(2))
```

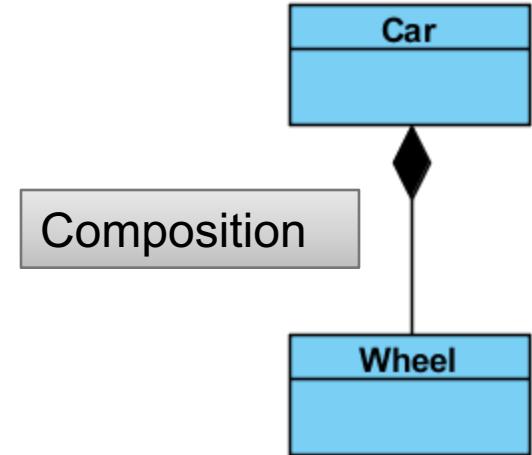
```
        self.wheels.append(Wheel(3))
```

```
        self.wheels.append(Wheel(4))
```

```
    def __str__(self):
```

```
        return repr("Car:: Wheels : " +
```

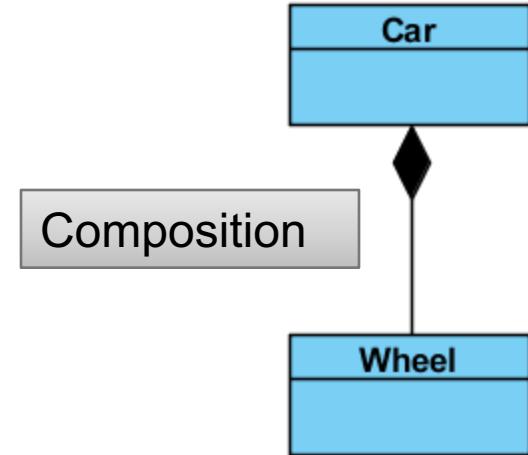
```
                   ', '.join(str(w) for w in self.wheels))
```



COMPOSITION

```
from car import Car
```

```
#wheels created with the Car
c = Car()
print(c)
```

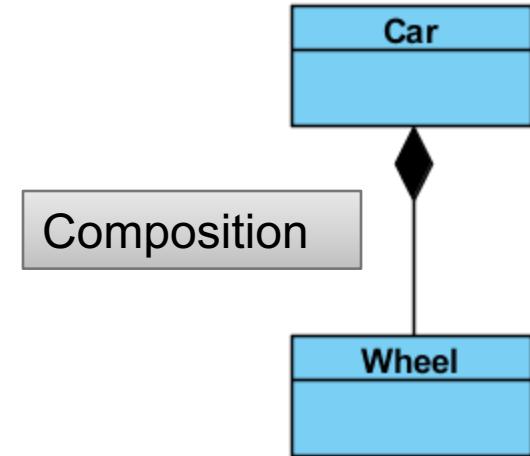


"Car:: Wheels : 'Wheel 1', 'Wheel 2', 'Wheel 3', 'Wheel 4'"

COMPOSITION (JAVA)

```
public class Wheel{  
    . . .  
}
```

```
public class Car{  
    ArrayList wheels;  
  
    public Car(){  
        wheels = new ArrayList();  
        wheels.add(new Wheel());  
        wheels.add(new Wheel());  
        wheels.add(new Wheel());  
        wheels.add(new Wheel());  
    }  
}
```

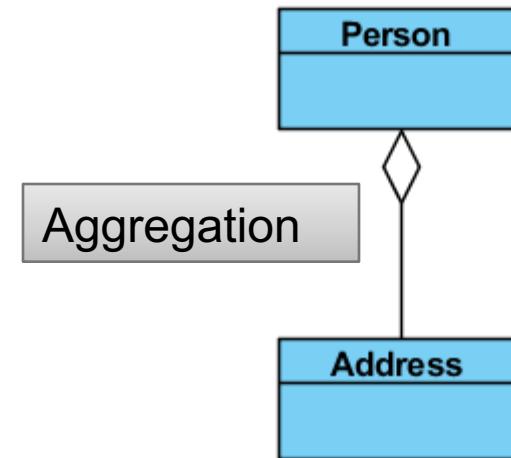


AGGREGATION

Person has one (or more) Address(es)

When a Person object is destroyed/ceased to exist, the associated Address object is **not** destroyed

Think of it as John (Person) bought a house (Address), this house is added to the John “object”. John died (assume the John object is destroyed), the house (Address) is not automatically destroyed because he died



AGGREGATION

```
class Address:
```

```
    def __init__(self, street, unit, postal):
```

```
        self.street = street
```

```
        self.unit = unit
```

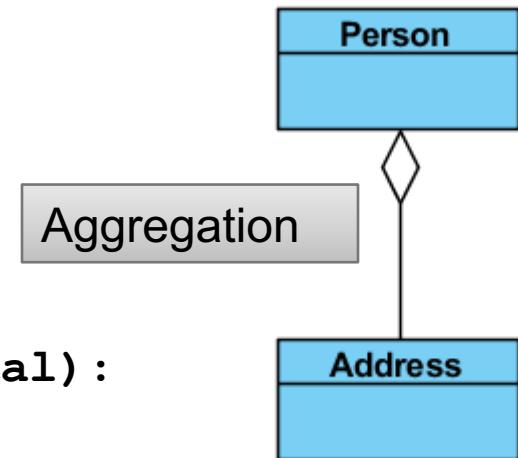
```
        self.postal = postal
```

```
    def __str__(self):
```

```
        return repr("Street : " + self.street
```

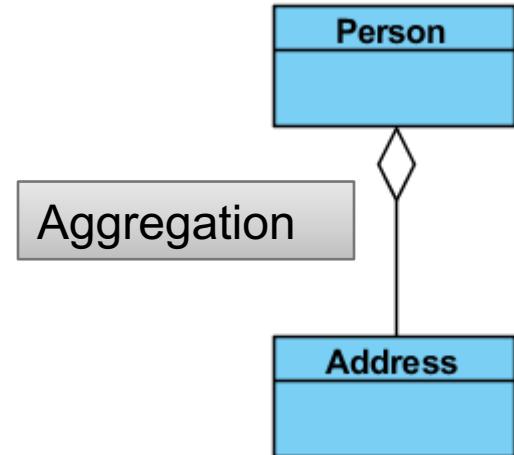
```
                    + ", Unit : " + self.unit
```

```
                    + ", Postal : " + self.postal)
```



AGGREGATION

```
class Person:  
  
    def __init__(self, address):  
        self.address = address  
  
  
    def __str__(self):  
        return repr("Person with address : "  
                   + str(self.address))
```



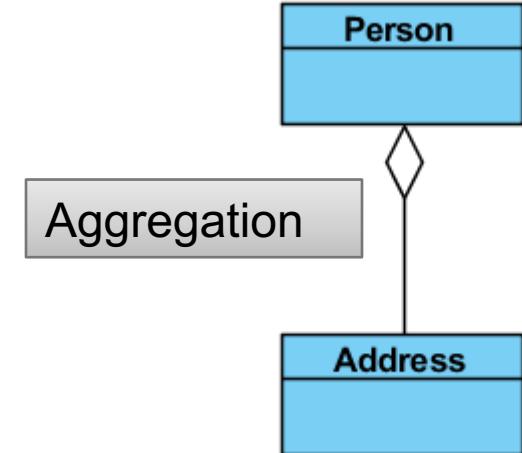
AGGREGATION

```
from person import Person  
from address import Address
```

```
add1 = Address("13 Computing Drive", "#01-01", "117417")
```

```
john = Person(add1)
```

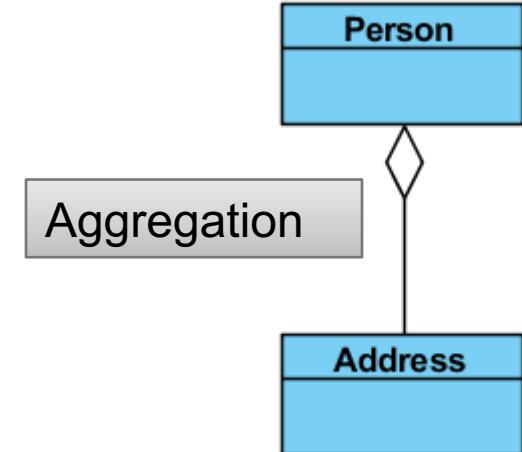
```
print(john)
```



"Person with address : 'Street : 13 Computing Drive, Unit : #01-01, Postal : 117417'"

AGGREGATION (JAVA)

```
public class Address{  
    . . .  
}
```



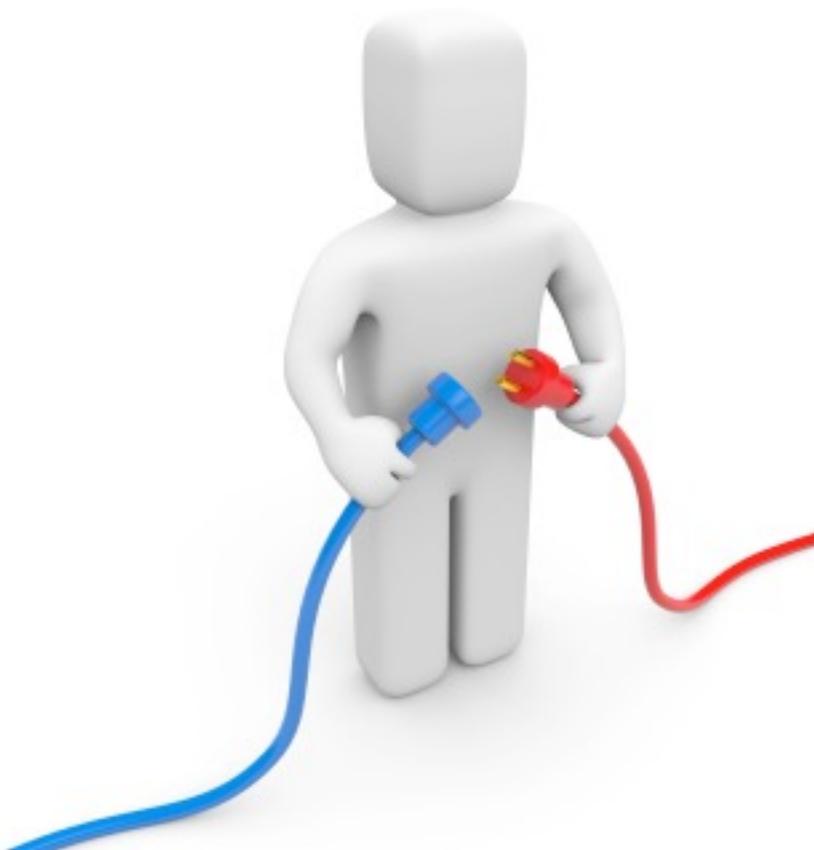
```
public class Person{  
    private Address address;  
  
    public Person (Address address) {  
        this.address = address;  
    }  
}
```

STEPS IN STRUCTURAL ANALYSIS



4. Analyze associations

STEPS IN STRUCTURAL ANALYSIS



Discover the ways how one business object links with others

- Each of these relationships is an association

ANALYZE ASSOCIATIONS

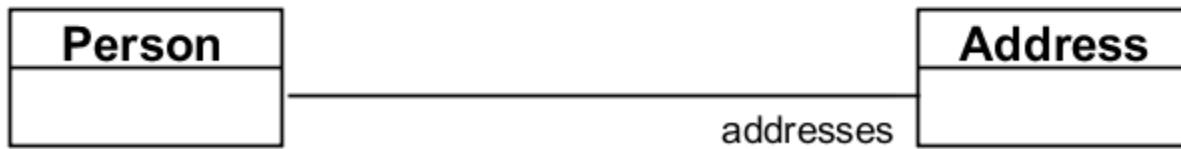
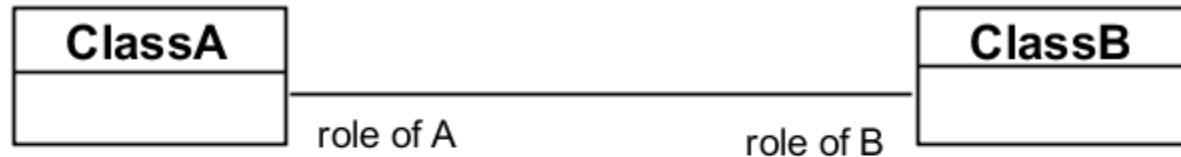
Information about one object refers to information about another object

- e.g. invoice data refers to product information (such as description and price), so Invoice is associated with Product

To carry out a business operation relating to one object, the operation affects related objects

- e.g. when a booking is canceled (an operation of Booking), flights must be updated to reflect the newly available seat. Booking is thus associated with Flight

INDICATE ASSOCIATIONS



class Address:

...

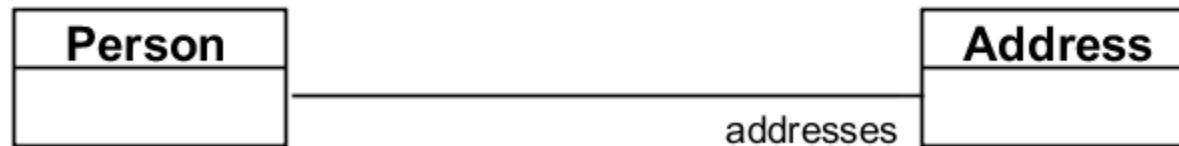
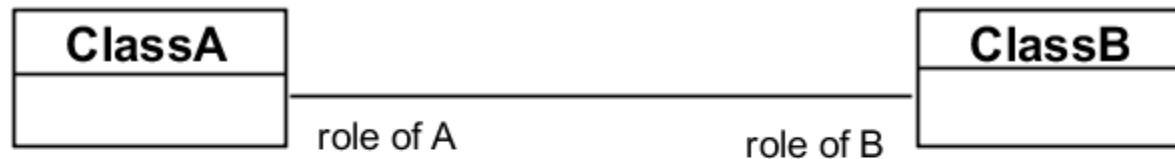
rolename

```
class Person:  
    def __init__(self, addresses):  
        self.addresses = addresses  
    ...
```

Or

```
class Person:  
    addresses = []  
    ...
```

INDICATE ASSOCIATIONS (JAVA)

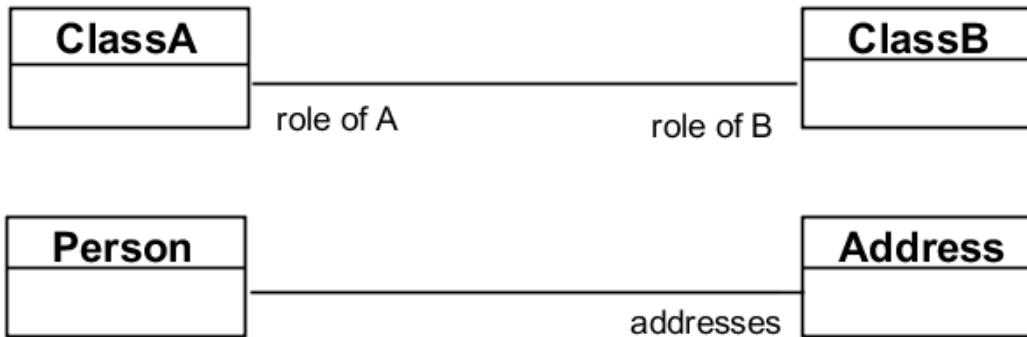


```
public class Address{  
    . . .  
}
```

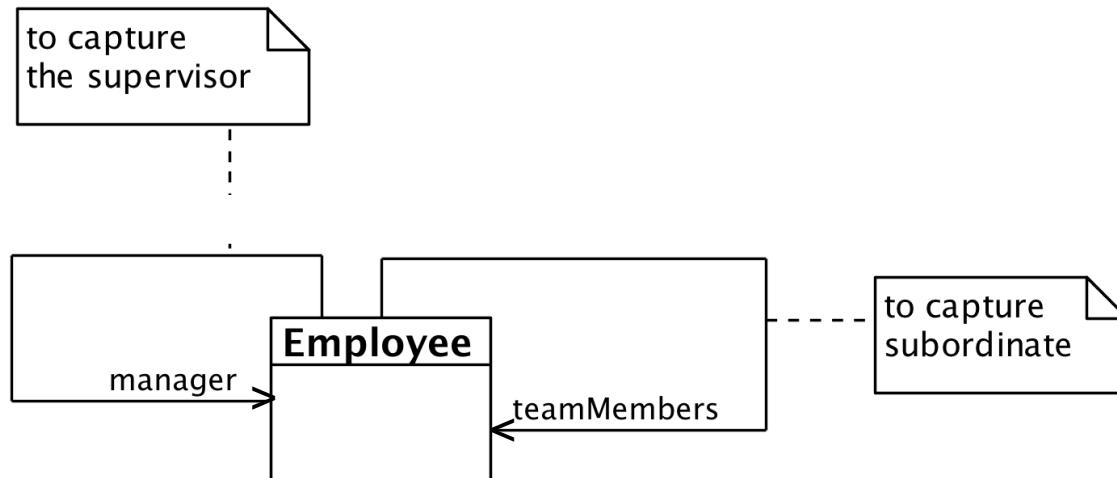
```
public class Person{  
    private Address[] addresses;  
    ...  
}
```

rolename

INDICATE ASSOCIATIONS



Binary Association



Reflexive Association

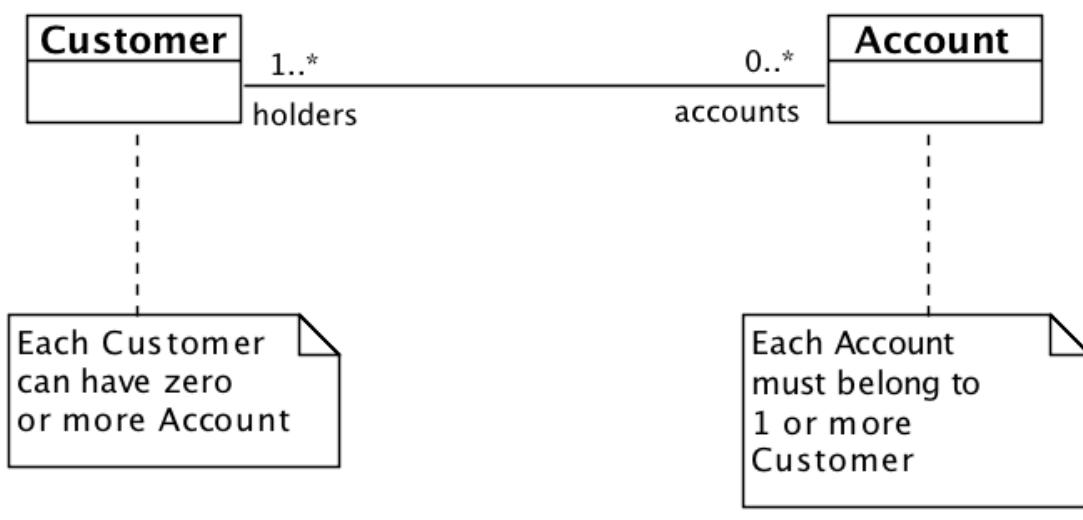
STEPS IN STRUCTURAL ANALYSIS



5. Analyze Multiplicity

- Model business rules that deal with the number of business objects that may be linked to each other

INDICATE MULTIPLICITY



1	Exactly one
*	Unlimited number (zero or more)
0..*	Zero or more
1..*	One or more
0..1	Zero or one
3..5	Specific Range (3 to 5 inclusive)

DOMAIN MODEL CLASS DIAGRAM

Introduction
to Structural
Modeling

Introduction
to Entity
Class

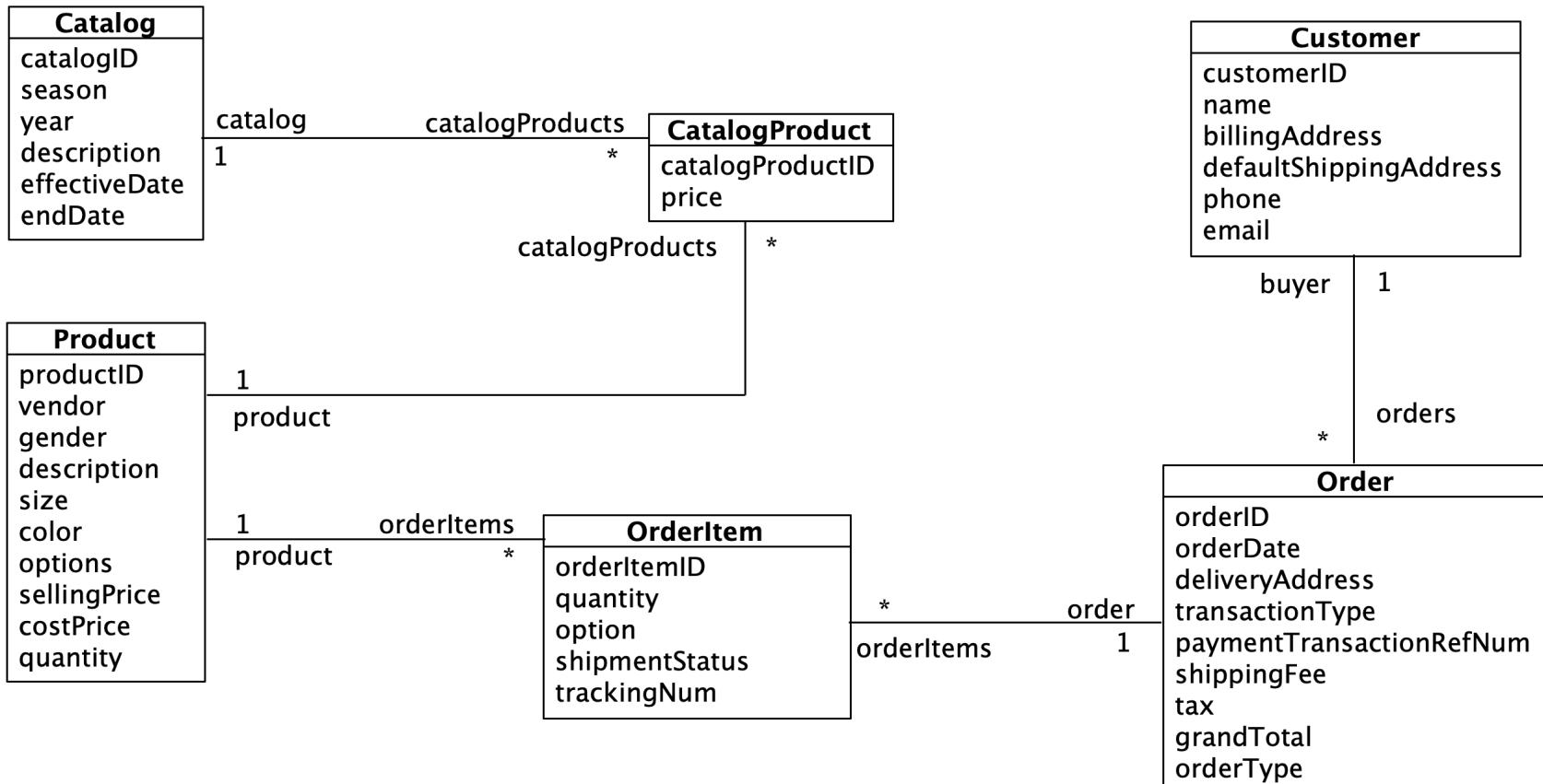
Class
Diagram
Notation

Domain
Model Class
Diagram

SIMPLIFIED DOMAIN MODEL CLASS DIAGRAM OF RMO

Domain Model Class Diagram only contains the Classes, Attributes, Rolenames, Associations

It doesn't have the operations/methods



A close-up photograph of a man with dark hair and glasses, wearing a grey shirt. He is holding a plain white rectangular card in front of his chest with both hands. His fingers are visible at the edges of the card. The background is a soft-focus green.

It's your turn...

TASK

IS2102 Exam Question (Nov/Dec 2012 Q45)

SUMMARY

**Structural Modeling using UML Class Diagram
(Domain Model Class Diagram)**

WHAT'S NEXT?

Modeling States

LECTURE 7

MID SEM SUMMARY

LEK HSIANG HUI

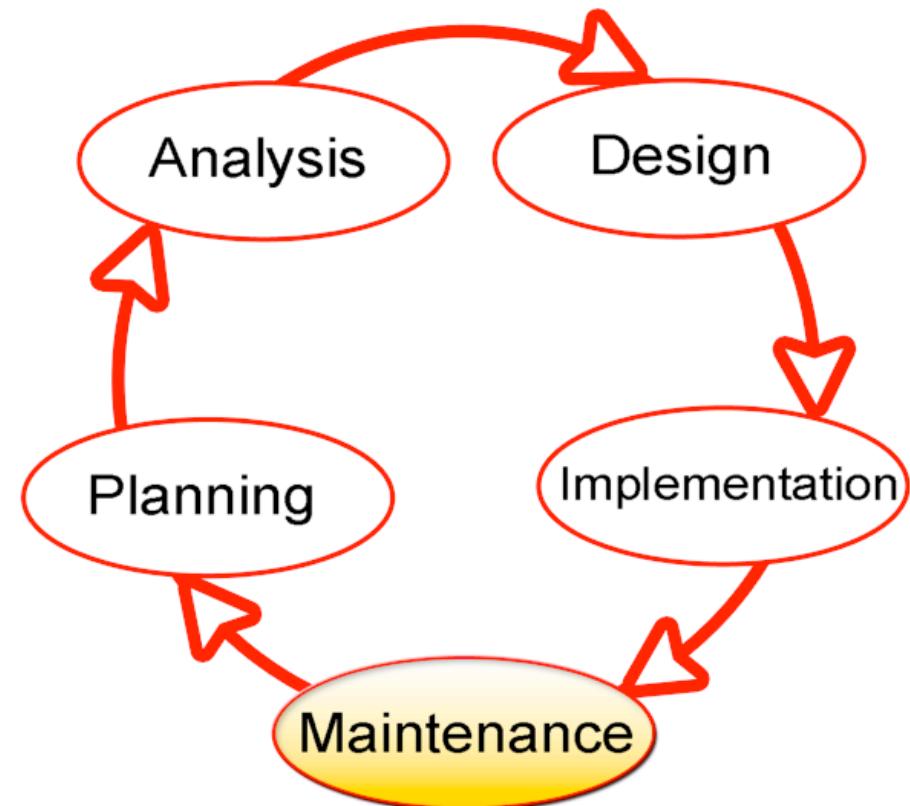
LEARNING OBJECTIVES

At the end of this lecture, you should better appreciate how the diagrams covered so far support the planning and analysis phase of a typical Systems Development Life Cycle (SDLC)

SYSTEMS DEVELOPMENT LIFE CYCLE (SDLC)

The development life cycle does not only involve coding

In fact, implementation is only one of the phase



PLANNING PHASE

Planning Phase

- Be careful not to think that SDLC is just about drawing diagrams + coding
- The UML diagrams introduced is just tools to guide the process
- Ultimately, it is about gathering and analyzing the requirements that matters (at this stage)
- So, realistically before we go ahead and gather the requirements, it is important to **establish the scope of the system**

GROUP DISCUSSION 1 : ESTABLISHING SCOPE



Suppose we are designing an online bookstore portal (like Amazon Books)

- 1. Discuss in groups and identify who are the stakeholders (different types of users for such a system)**
- 2. Post your list onto Canvas discussion forum**

GROUP DISCUSSION 1 : ESTABLISHING SCOPE



Questions to think about (you are free to design the system according to how you would like it to be?)

- 1. Who is selling the books? This company (that is maintaining this website?) or do they just provide a marketplace for book publishers to sell on this platform?**

**What are the
diagrams/tables we have
covered so far?**

DIAGRAMS/TABLES

Planning Phase

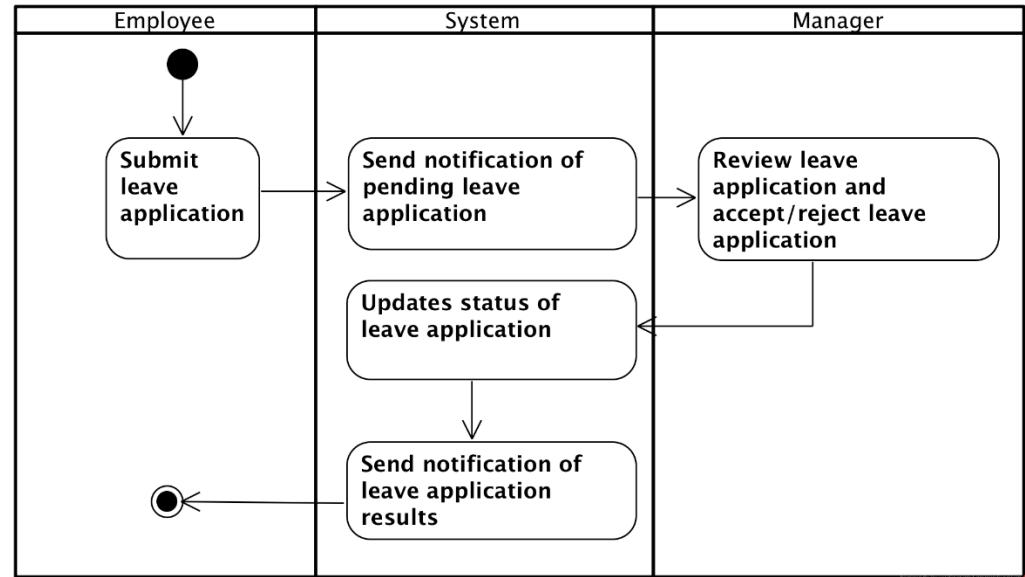
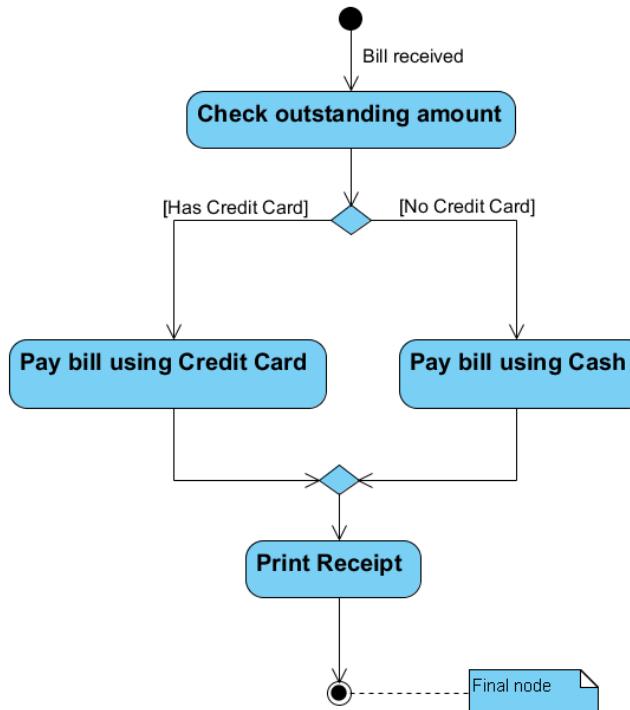
- Initial stage of the project: understand the company, identify the project objective/scope, etc
- **Activity Diagram, Use Case Diagram, Role Map**

DIAGRAMS/TABLES

Swimlane approach is preferred

Planning Phase

- **Activity Diagram (use to model workflow : general business workflow or system use case flow)**



BUSINESS PROCESSES

Before we produce the use cases, it is often helpful to identify the major business processes

This allows us to have better clarity about whether the stakeholders we have identify earlier is realistic and what the interaction between the different stakeholders will be like

- I.e. difficult to just sit down and list out users without mentally visualizing how each stakeholder talk to each other

GROUP DISCUSSION 2 : IDENTIFY MAJOR BUSINESS PROCESSES



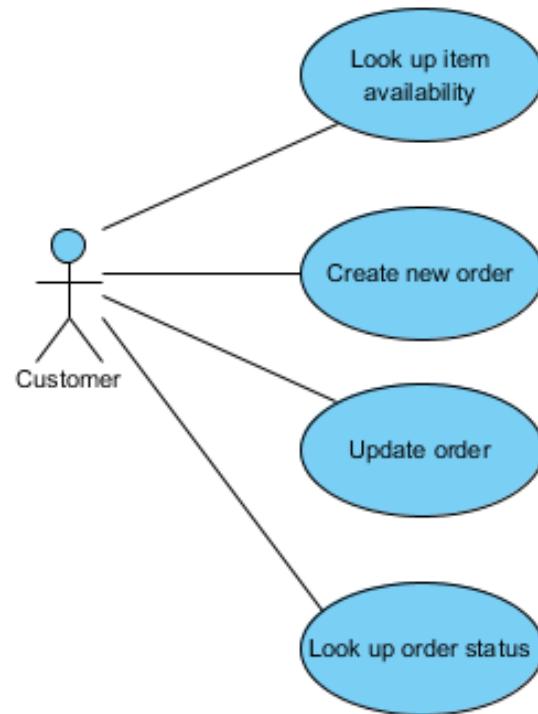
Next step is to identify the major business processes

- 1. Discuss without your group and identify the major business processes and the interaction between different stakeholders**
- 2. Consider using ChatGPT to guide you if you need help**
- 3. Draw multiple activity diagrams (one for each major business process) to encode this information**
- 4. Embed your activity diagrams onto a forum post**

DIAGRAMS/TABLES

Planning Phase

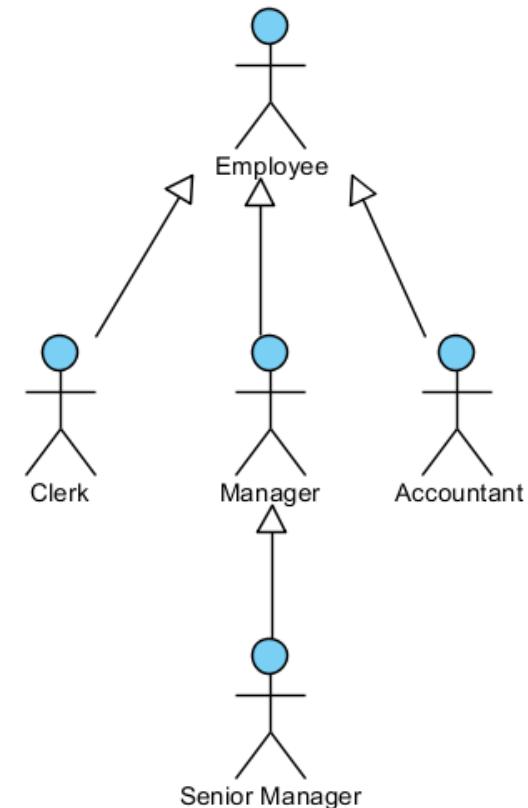
- Use case diagram



DIAGRAMS/TABLES

Planning Phase

- **Role Map (identify the relationship between the different actors – potential users of the system)**
- **Usually, we will embed this information in the use case diagrams we draw rather than draw the use case diagrams and rolemaps separately**



GROUP DISCUSSION 3 : PRODUCE USE CASE DIAGRAMS



Next step is to produce the use case diagrams

- 1. As there will be many use cases, you might want to produce the use case diagram by subsystems (1 use case diagram for each subsystem)**
- 2. What this means is that potentially you might need to first identify what are the subsystems**

GROUP DISCUSSION 3 : PRODUCE USE CASE DIAGRAMS



- 3. (Due to time constraint) In your group, choose one subsystem and produce the use case diagram for that subsystem**
- 4. Consider using ChatGPT to guide you if you need help but you should verify that it is giving you proper use cases**
- 5. Make sure you indicate which subsystem your group is working on**
- 6. Embed your use case diagram onto a forum post**

GUIDE FOR IDENTIFYING USE CASES

1. Need to identify all the actors of this system

- Note that actors need not be just human users, they can be **other systems** that interact with this system

2. A use case must be a system function/interaction

- E.g. clerk calling a customer is NOT a use case - it's not a system function

3. A use case should fulfil a business objective

- E.g.
Actor: Student
Use case: View list of enrolled courses

GUIDE FOR IDENTIFYING USE CASES

4. While a use case is a system function/interaction, need to know how to differentiate between a use case vs a step of a use case
 - Note that a use case is a high-level system function
 - We will derive the individual steps of a use case in future phases (not now)
 - Do NOT try to list out the individual steps of a use case
5. Use cases are not just a function in code. It should be seen from a point of view of an actor
 - E.g. “check password during login”, “send email confirmation”, “add to database”, etc are not use cases. They are what the system/code does

GUIDE FOR IDENTIFYING USE CASES

- 6. Use cases are seen from the perspective of a user rather than from the perspective of the system**
 - i.e. “get a list of chat messages” rather than “receive a chat message”
- 7. A use case diagram should not have this system itself as an actor**
 - Usually, we omit time event in the use case diagram
 - But if you really want to draw, you can create an “actor” called TIME

DIAGRAMS/TABLES

Analysis Phase

- Go into detailed analysis of the project
- **Use Case Description, Domain Model Class Diagram**