PROJECT MANAGEMENT

MODULE 3 – Winter 2023

HUMBER

WE ARE HUMBER

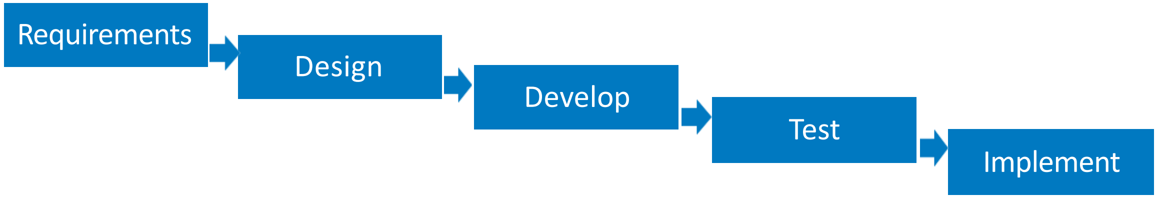
Agenda:

- Review of development approaches (Waterfall, Iterative, Incremental, Agile)
- Project Scope Management – using requirements to manage scope
- Identify use cases
- User Stories technique
- User Goal technique
- Use Case Diagrams

- Lab #1- Use Case Diagrams

Details: Predictive (Waterfall)

Fully plan-driven approach where the three main project constraints (time, scope, cost) are all determined at a detailed level at the start of the project



Additional Details:

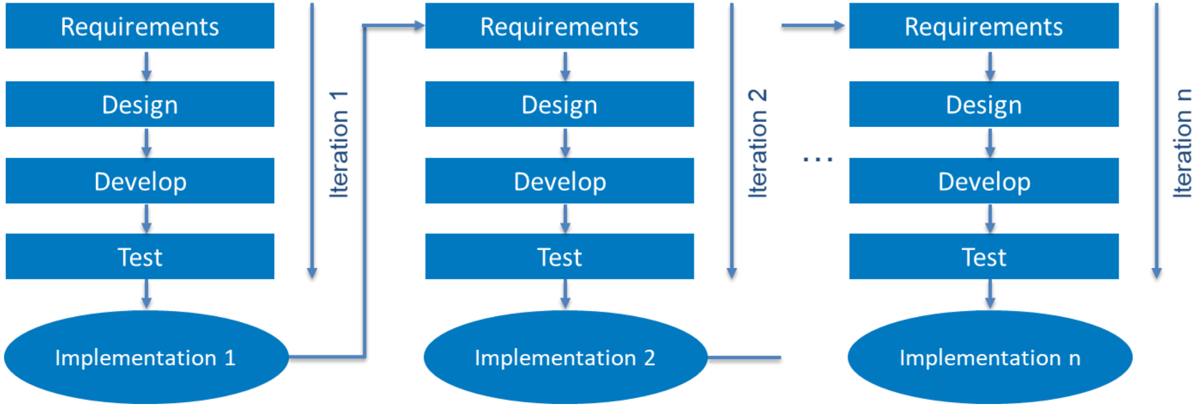
- You need to know your requirements going in
- The scope is fixed at the beginning
- There are distinct phases of activities: Requirements, Design, Development, Testing and Deployment
- A linear-sequential life cycle model; each phase starts after the previous phase completes
- The software is deployed into production after the completion of the Testing phase
- At the end of each phase, a review (or gate) takes place to determine if the project is on the right path and fit to move forward to the next phase of the project.

When to Use:

- Only when the requirements are very well known, clear and fixed
- Product definition is stable
- Technology is understood
- There are no ambiguous requirements
- Resources with required expertise are readily available

Details: Iterative

Breaking down the software development into smaller pieces, enabling better requirements definition at the start of each cycle.. Each release of is developed in a specific and fixed time period – called an “iteration”.



Additional Details:

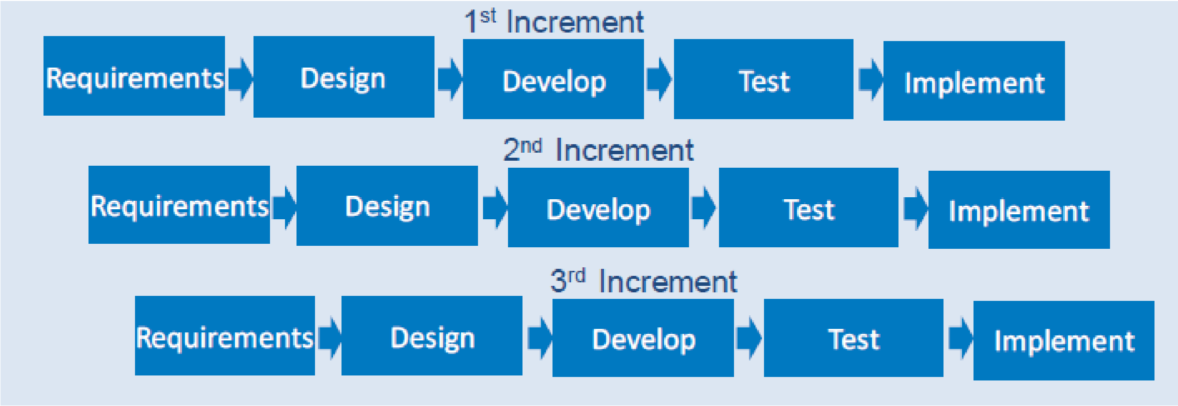
- There are the same five phases of activity
- Starts with a simple development of a small set of requirements, which iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed
- PMI (PMBOK Guide) recommends scope definition early in the project, but time and costs can be modified after each iteration when they are better understood
- This approach is comparable to many waterfall cycles with the customer verifying the work at the exit of each cycle
- It provides flexibility to address changes, which reduces risk
- The requirements are detailed for the next phase when you are done with the previous

When to Use:

- When requirements of the complete system are defined and understood
- Major requirements are defined, while some functionalities and requested enhancements evolve with the development process
- A new technology is being used and is being learned by the development team, while they are working on the project
- If there are some high-risk features and goals, which might change in the future

Details: Incremental

The incremental lifecycle approach develops a product through the implementation of incremental steps which have predetermined timeframes. Each increment delivers additional functionality for the product and is repeated until the final deliverable is produced.



Additional Details:

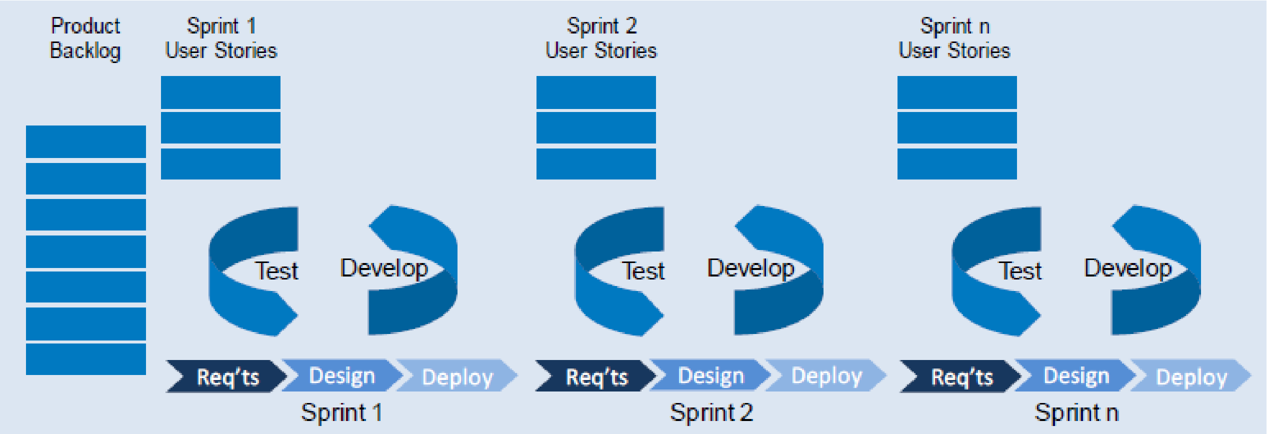
- Often, the Incremental approach is grouped with Iterative
- They are different but have some similarities
- Like with the Iterative approach, customers sign-off at each exit point

When to Use:

- When functionality can be split-up into increments (portions), whereby some functionality is delivered in each increment.
- When prototyping is a preferred approach

Details: Agile

The Agile Framework applies an incremental approach to work based on transparency, inspection, and adaptation, while also optimizing predictability and controlling risk.



Additional Details:

- Doesn't have separate phases of Requirements, Design, Development or Testing
- Combines both iterative and incremental approaches
- Requirements are represented by User Stories that are created and refined prior to being assigned to a Sprint; with the remaining User Stories sitting in the Backlog
- Design, development and testing are done together within Sprints
- Sprints generally last between two to four weeks each and include a limited number of User Stories
- There are many sprints within a product development release, depending upon the number of User Stories to complete

When to Use:

- Shorter planning based on iterations and multiple deliveries
- Flexible, cross-functional team composition
- Changes in deliverables are expected and less impactful
- Product delivered in functional stages
- Collaborative and interactive approach to requirements
- Customer is involved throughout the sprint
- Concurrent approach seeks to reduce dependencies

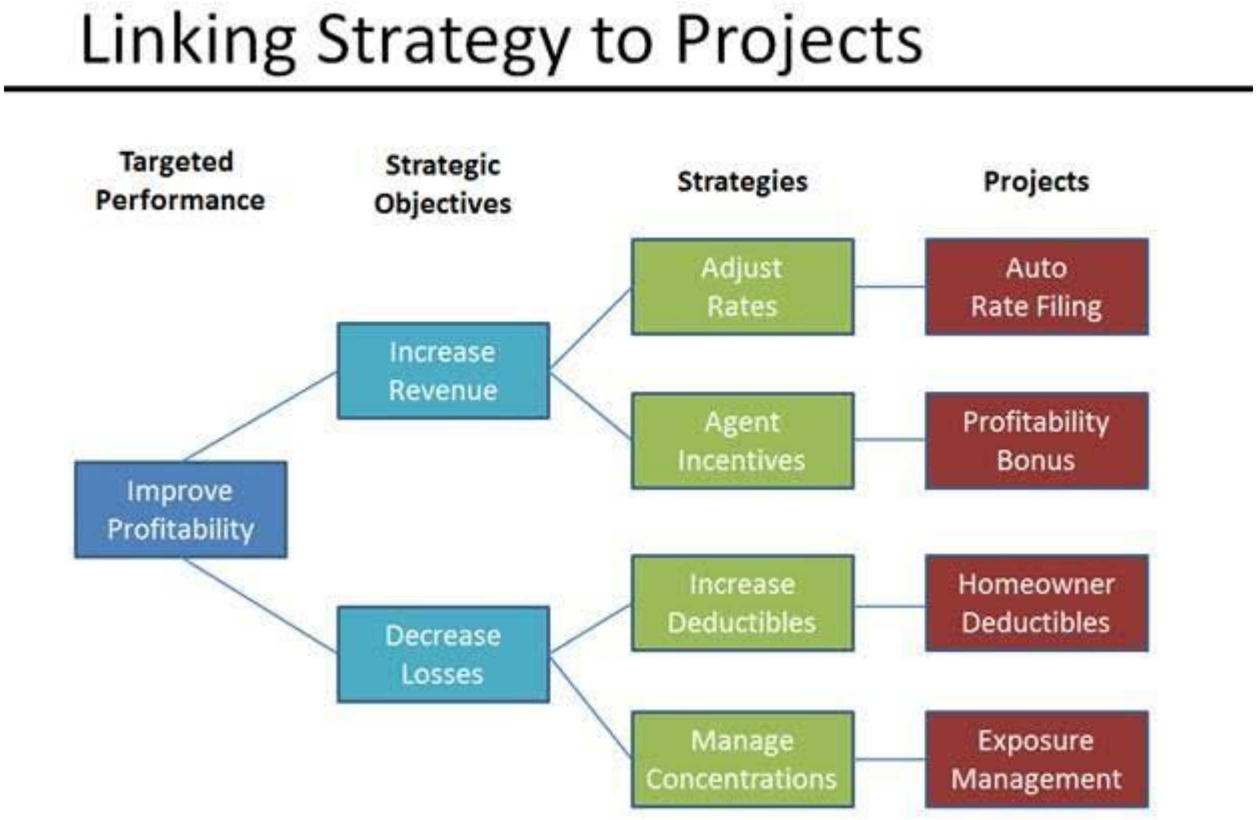
Summary of Characteristics by Framework

	Predictive (Waterfall)	Iterative/ Incremental	Agile
Requirements	Are complete, clearly defined, documented, and baselined in the planning phase of the project	Are for a set of features, rather than complete set of requirements for the project	Represented by the User Stories that are created and refined prior to being assigned to a specific sprint
Definition of Ready	Analysis & design is completed for all requirements before development begins	Analysis & design is completed for a 'set of requirements' before development begins	Requirements (user stories) that are not subject to change, that can be completed within the sprint will be considered for the sprint
Scoping	Sponsor decides project scope	Project Manager decides project scope for the iteration with Sponsor	Team decides the scope of the sprint based on the capacity & product backlog prioritization
Status Reports	Prescribed template		Daily Stand-up meeting (Scrum)
Task Assignment	PM assigns the tasks for the entire project		Team members are empowered to own the tasks for the entire sprint
Resource Utilization	Role specific; Resources only work on their assigned tasks	Role switchover is rare; Resources extend only on an 'as needed' basis	Role switchover is common; Everybody is ready to work on any task to complete sprint
Definition of Done	Delivery of artifacts that are developed, approved, and delivered at end of respective phase	Phase wise delivery of features (analysis & design in one phase, followed by development & testing in another phase) within the same iteration	A potentially shippable minimum viable product or the sprint
Quality Control	Detection and fix during the testing phase at the end of the project	Early detection and fix in each iteration followed by regression testing before deployment	Early detection and fix in each sprint
Delivery	Artifacts by phase and delivery of working software at end of project	Working software at end of each iteration	Working software at end of each sprint
Continuous Improvement (Lessons Learned)	At end of project; Lessons learned from previous projects implemented in next project	At end of every iteration; Lessons learned from previous iteration implemented in next iteration	Retrospective at end of every sprint; Lessons learned from previous sprint implemented in next sprint

What is Strategic Project Management?

Strategic Project Management (SPM) is the process of thinking about your Projects considering their connection to your strategic plan. In other words, Strategic Project Management is about forming clear links between your Projects and Strategic Objectives.

The premise of Strategic Project Management is that 'Projects' should actually work to achieve the goals and objectives outlined in your strategy.



Linking Strategy to Projects

Targeted Performance

Strategic Objectives

Strategies

Projects

Improve Profitability

Increase Revenue

Decrease Losses

Adjust Rates

Agent Incentives

Increase Deductibles

Manage Concentrations

Auto Rate Filing

Profitability Bonus

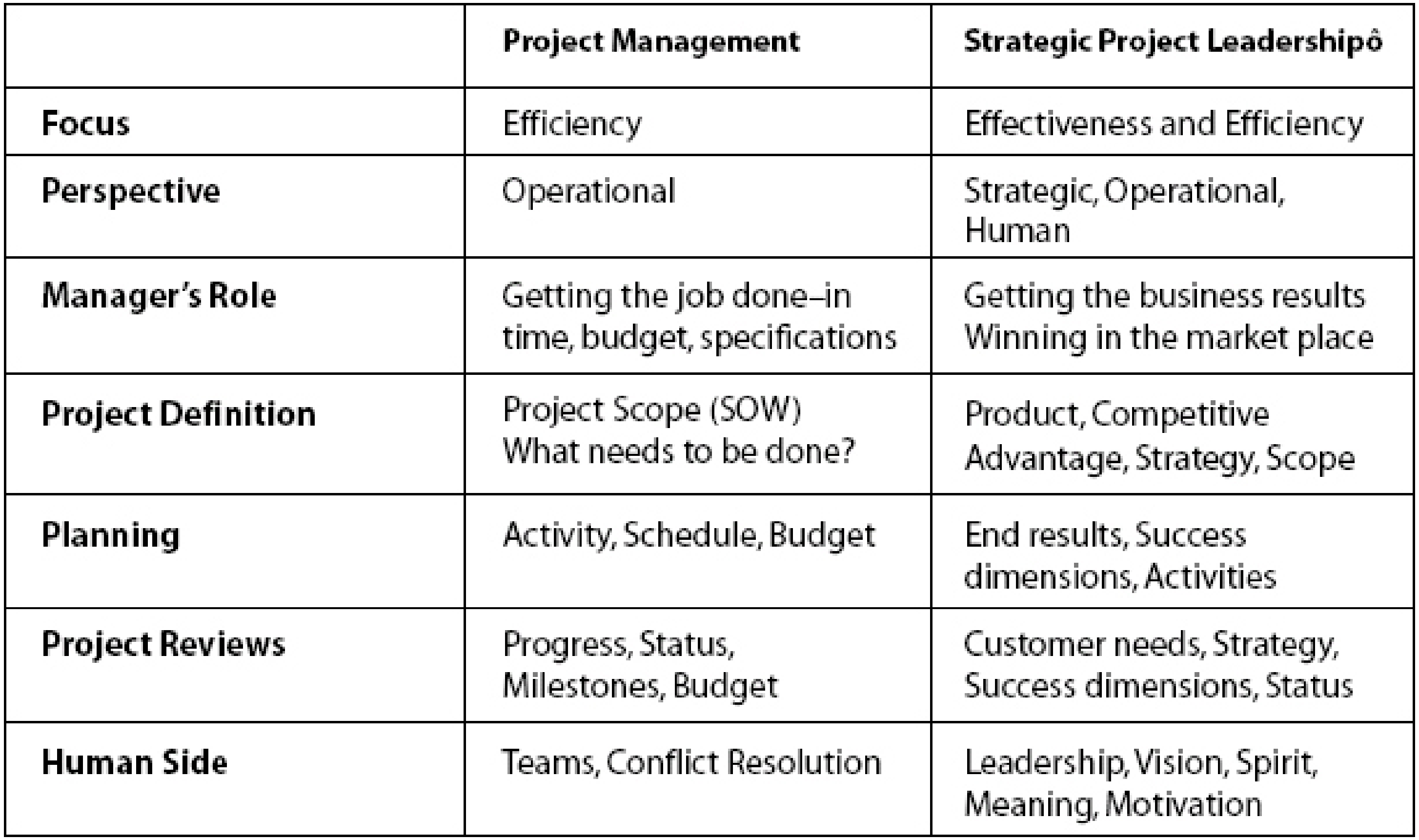
Homeowner Deductibles

Exposure Management

Insurance Company Example - Strategy to Projects Map

Source: projectriskcoach.com

From Project Management to Strategic Project Leadership



	Project Management	Strategic Project Leadershipô
Focus	Efficiency	Effectiveness and Efficiency
Perspective	Operational	Strategic, Operational, Human
Manager's Role	Getting the job done–in time, budget, specifications	Getting the business results Winning in the market place
Project Definition	Project Scope (SOW) What needs to be done?	Product, Competitive Advantage, Strategy, Scope
Planning	Activity, Schedule, Budget	End results, Success dimensions, Activities
Project Reviews	Progress, Status, Milestones, Budget	Customer needs, Strategy, Success dimensions, Status
Human Side	Teams, Conflict Resolution	Leadership, Vision, Spirit, Meaning, Motivation

Source: pmi.org

Project Justification – The Business Case

What is the problem, need, or opportunity we are trying to solve?

A business case **provides justification for undertaking a project, program**. It evaluates the benefit, cost and risk of alternative options and provides a rationale for the preferred solution.

Business Case

The Business Case includes the following key sections:

- Executive summary
- Background information
- Project definition
- Business Requirements
- Option presentation and evaluation
- Presentation of preferred option
- Strategic alignment
- Benefits

Inputs to Scope Management

- Key inputs for preparing the project scope statement include the project charter, requirements documentation, and organizational process assets such as policies and procedures related to scope statements as well as project files and lessons learned from previous, similar projects.

- The **Project charter** provides a basis for further defining the project scope. The charter describes the high-level scope, time, and cost goals for the project objectives and success criteria, a general approach to accomplishing the project s goals, and the main roles and responsibilities of important project stakeholders.

Scope Management using Requirements

- Collecting requirements involves defining and documenting the features and functions of the products produced during the project as well as the processes used for creating them. The project team creates stakeholder requirements documentation, a requirements management plan, and a requirements traceability matrix as outputs of the requirements collection process.

- Defining scope involves reviewing the project charter, requirements documents, and organizational process assets to create a scope statement, adding more information as requirements are developed and change requests are approved. The main outputs of scope definition are the project scope statement and updates to project documents.

Scope Management using Requirements

- Project scope management includes the processes required to ensure that the project addresses all the work required, and only the work required, to complete the project successfully. The main processes include collecting requirements, defining scope, creating the WBS, verifying scope, and controlling scope.

- The first step in project scope management is collecting requirements, a crucial part of many IT projects. It is important to review the project charter and meet with key stakeholders listed in the stakeholder register when collecting requirements. The main outputs of this process are requirements documentation, a requirements management plan, and a requirements trace- ability matrix.

Ways to gather Requirements

- Brainstorming
- Interviews
- Focus group
- Observing
- Prototyping
- Competition analysis / Reverse Engineering
- Surveys

Types of Requirements

- Business Requirements (BRD)
- User Requirements (UI/UX)
- System/Technical Requirements (SRS)
  - Functional Requirements (FR)
  - Non-Functional Requirements (NFR)

Gathering Requirements

Requirements engineering aims at defining the requirements of the system under construction.

- Define the purpose of the system
- Goal is to identify “actors” and “use cases”
- Actors:
  - Entities that interact with the system
  - Could be internal users, external systems, or process
- Use cases:
  - Sequence of events
  - All possible actions between actors and other use cases are identified

Requirements Elicitation

- Requirements elicitation is about communication among developers, clients, and users to define a new system
- Errors introduced during requirements elicitation are expensive to correct
  - Missing functionality
  - Incorrect functionality
  - Misleading user interface
  - Obsolete functionality
- Improving communication reduces requirement errors

Use Case Modelling

Use Cases modelling is an effective means of communicating with users and other

stakeholders about the system and what is intended to do.

- Support requirements engineering activities and the requirement process
- Capture what a system is supposed to do, i.e., systems functional requirements

- Documents the functionality of the system from the user's perspective
- Documents the scope of the system
- Describe sequences of actions a system performs that yield an observable result of value to a particular actor

User Stories and Use Cases

- A **use case** is an activity that the system performs, usually in response to a request by a user. A **user story** is one short sentence in everyday language of the end user that states what a user does as part of his or her work
- User stories and use cases are very closely related concepts. User stories are simple, one sentence descriptions of a user task that utilizes the system. User stories are normally created by the users themselves. User stories are used in Agile development projects where there is ongoing user involvement.

The template for a user story is in this form:

- **As a <*role played*>, I want to <*goal or desire*> so that <*reason or benefit*>**

It answers 3 questions:

- 1. Who? → *As a <type of user>,*
- 2. Does what? → *I want to <some behavior from the system>*
- 3. And why? → *so that <some value is achieved>*

Use Cases and User Goal Technique

One approach to identifying **use case**s, called the **user goal technique**, is to ask users to describe their goals for using the new or updated system.

The user goal technique for identifying use cases includes these steps:

1. Identify all the potential users for the new system.

2. Classify the potential users in terms of their functional role (e.g., shipping, marketing, sales).

3. Further classify potential users by organizational level (e.g., operational, management, executive).

4. Interview all users (or all types) - what are their specific goals?

5. Create a list of use cases, organized by type of user

6. Look for duplicates, resolve inconsistencies

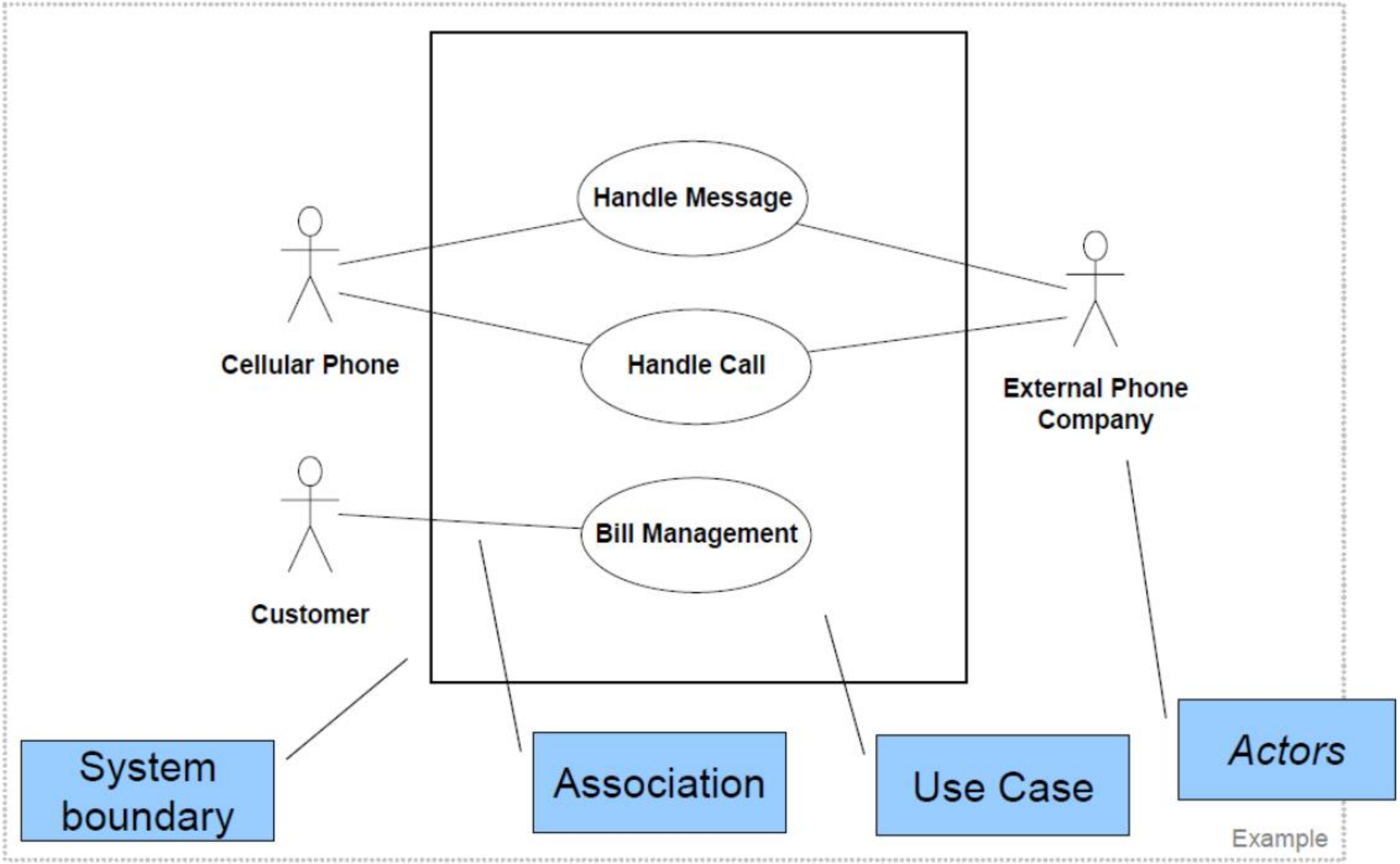
7. Identify where different users have the same use case

8. Review updated list with users and stakeholders

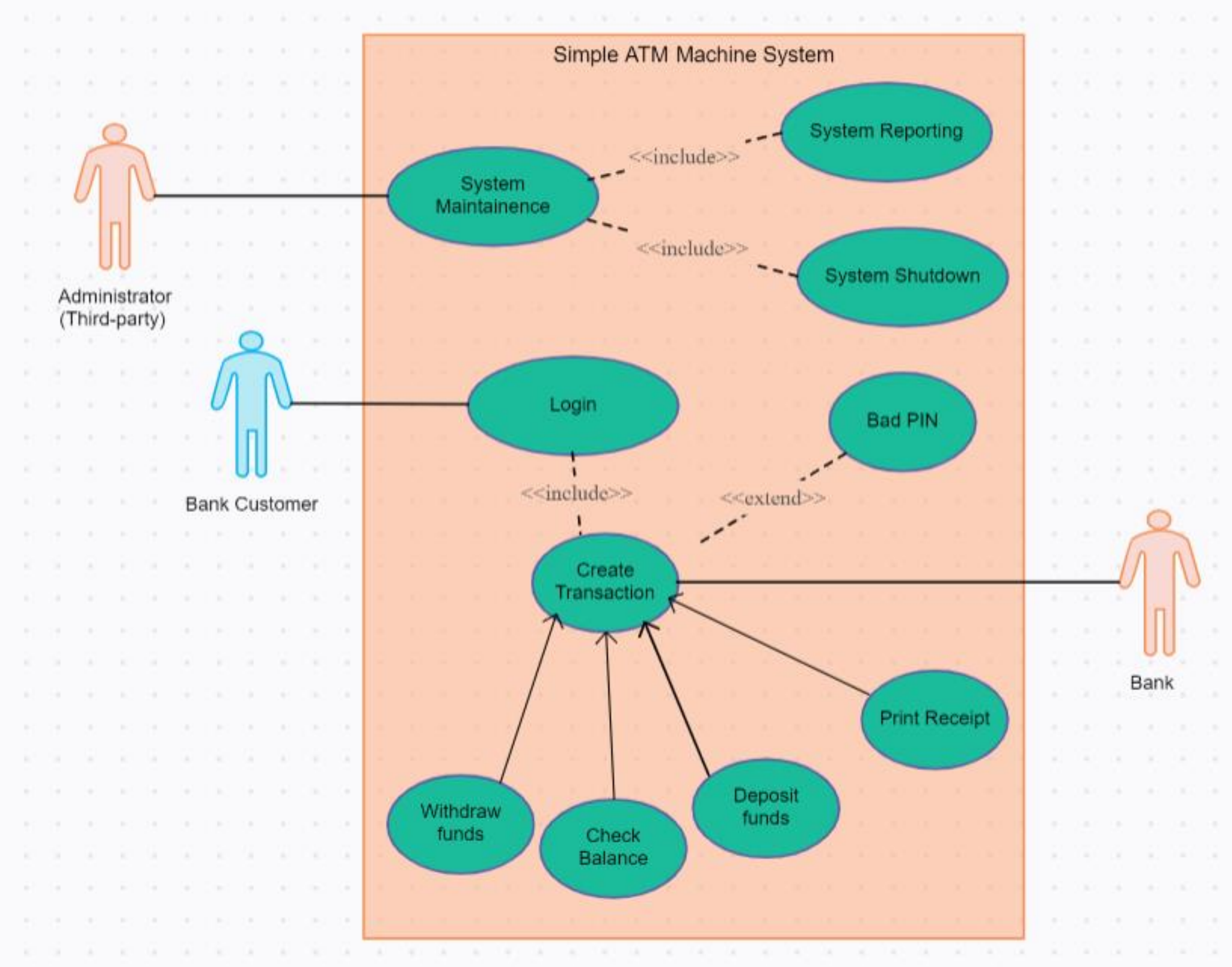
Use Case Diagrams

- **Actors**: An Actor is external to a system, interacts with the system, may be a human user or another system, and has a goals and responsibilities to satisfy in interacting with the system. Actors are represented as stick figures
- **Use Cases**: identify functional requirements, which are described as a sequence of steps describe actions performed by a system capture interactions between the system and actors. Use cases are represented as ellipses
- **Relationships**: Actors are connected to the use cases with which they interact by a line which represents a relationship between the actors and the use cases.
  - Include: required use cases
  - Extend: optional use cases
- **System Boundaries**: Identify an implicit separation between actors (external to the system) and use cases (internal to the system)

Use Case Diagram example



Use Case Diagram example



Building Use Case Diagrams

The steps to develop use case diagrams are:

1. Identify all the stakeholders and users who would benefit by having a use case diagram.

2. Determine what each stakeholder or user needs to review in a use case diagram. Typically, a use case diagram might be produced for each subsystem, for each type of user, for use cases with the includes relationship, and for use cases that are of interest to specific stakeholders.

3. For each potential communication need, select the use cases and actors to show and draw the use case diagram. There are many software packages that can be used to draw use case diagrams.

4. Carefully name each use case diagram and then note how and when the diagram should be used to review use cases with stakeholders and users.

Use Case Diagrams Tutorial

An excellent overview to learn more about Use Case Diagrams.

https://www.youtube.com/watch?v=zid-MVo7M-E

Creating Use Case Diagrams

Tools to create Use Case Diagrams:

- Lucidchart https://www.lucidchart.com/pages/
- Visual Paradigm https://www.visual-paradigm.com
- draw.io https://www.draw.io/
- Visio/PowerPoint

LAB 1: Use Case Diagram

“Antiques for you” a high-end antique store has added an online presence to their store due to Covid-19 restrictions. They are now able to sell all their merchandise through online shopping.

Create a Use Case Diagram for the following:

The customer can view or search for items. They must log in if they want to add items to their shopping cart. They can view their carts, and may remove items or go to the check-out to make their purchases. The customer reviews his/her purchases, chooses a payment method and pays.

A sales employee at receives the order and purchase confirmation from the system and sends the electronic order to the warehouse.

The warehouse employee updates the order status. The customer may check the order status.

LAB 1: submission Instructions

Submit via Blackboard / Assignments / Lab #1

- Ensure you save your diagram, so you don't lose it, copy it into a MS Word
- Submit your diagram in MS Word or PDF file

You have until end of class today to submit your Assignment

Once you have completed your lab, CLASS IS FINISHED! See you next week!

NEXT WEEK:

- Project Planning
- Estimating techniques
- Lab #1- MS Project introduction

Note: please ensure you have access to MS project

THANK YOU.

HUMBER

WE ARE HUMBER