**Use Cases**

Description of all ways of user want to use system capture all way the user and system can interact that result in user achieving the goal. They also capture all the things can go wrong along the way prevent user from achieving goal.

User Stories

* Focussed on the result
* benefit of system functions
* Small increments for getting feedback

Use Case

* More granular, describes how the system will act
* Detailed documentation
* Most done up front

Hybrid model

In Industry Use Cases used more often when projects have more stringent documentation requirements

**User goal technique**

* Most commonly used in industry
* User goals for interacting with the new system are identified
* Simple and effective

Event decomposition technique – Business events are identified

steps help identify use cases:

1. Identify all the user
   1. More comprehensive and complete technique
   2. Identify the events that occur to which the system must respond
      1. Event– something that occurs at a specific time and place, can be described, and should be remembered by the system
      2. Types of Events

An event that occurs outside the system, usually initiated by an external agent or actor

Eg: Wants some information - Customer wants to know product details

An event that occurs as a result of reaching a point in time

-Internal outputs needed at points in time:

Management reports (summary or exception)

Operational reports (detailed transactions)

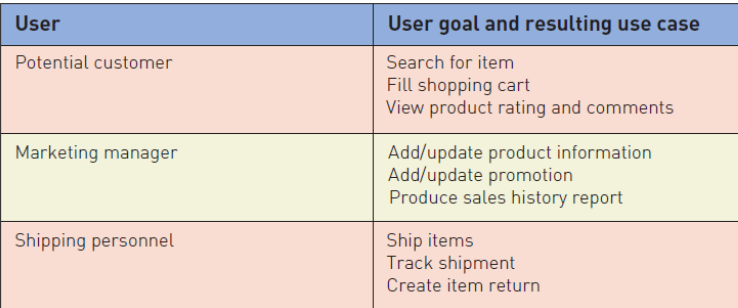
-External outputs needed at points of time:

Statements, status reports, bills, reminders

-an event that occurs when something happens inside the system that triggers some process

* 1. each event, name a use case describes what the system does when the event occurs

1. Classify them – Functional roles – Sales, Marketing – Organisational roles – Executive, Management, Operational
2. Interview each type of user to create an initial list of use cases
   1. Find out a list of specific goals
   2. Express them in VERB-NOUN format (e.g. Add customer)
   3. Interview and ask them to describe the tasks the system can help them with
3. Look for duplicates
4. Look for users with same needs
5. Review with users



Speak to potential customers through focus groups

Interview users from shipping and marketing departments

**Use Case Diagram Elements**

**ACTOR**

A person or another software entity (such as a system timer) that initiates the functionality provided by a Use Case, Roles not users or people

Primary vs. Secondary

Primary Actors: The Actor(s) using the system, The Use Case documents the interactions between the system and the actors to achieve the goal of the primary actor

Secondary Actors: Actors that the system needs assistance from to achieve the primary actor’s goal.

Eg: A bank loan officer wants to review a loan application from a customer, and part of the process involves a real-time credit rating check.

Primary Actor: Loan Officer; Secondary Actors: Credit Rating System

**USE CASE**

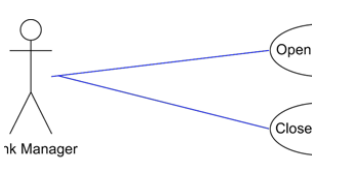
Use Case is an interaction between the system and a person or another system to achieve a result

“bit” of functionality

It yields an observable result of value to an actor

**ASSOCIATION**

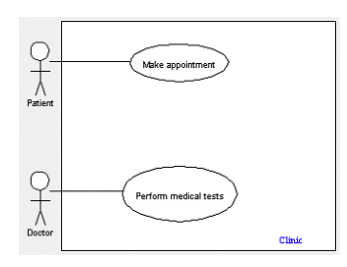
Association line indicates that a particular Actor makes use of the functionality provided by a particular Use Case.



**SYSTEM BOUNDARY**

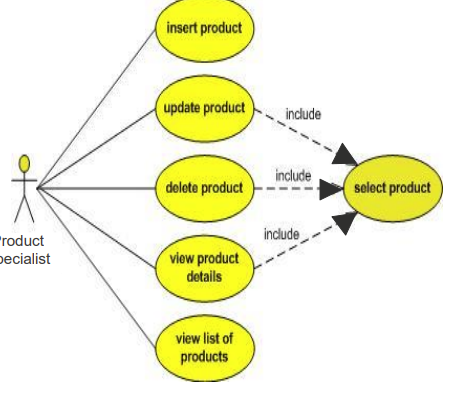
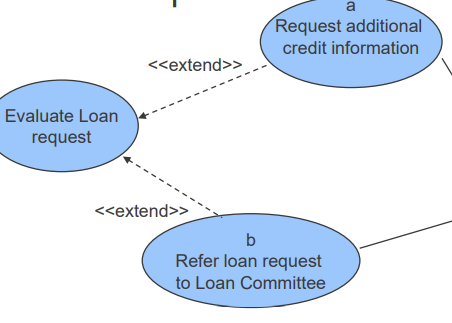
Defines the scope of what the system will be.

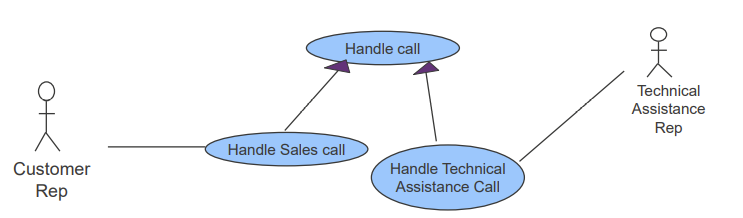
rectangle spanning



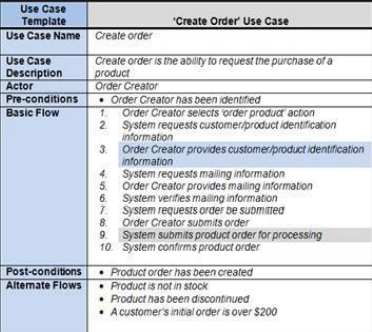
**USE CASE RELATIONSHIPS**

relationship between two use cases is basically a dependency between the two use cases

* Include- models encapsulated behaviors that can be inserted into a use case and possibly reused across multiple use cases
  + Captures commonality among use cases
  + Re-usability helps avoids repetition
  + An include relationship is depicted
  + 
* Extend- models significant extensions and behaviors that can occur as additions to the use case model
  + child use case adds to the existing functionality and characteristics of the parent
  + parent case can function without the child case
  + include relationship
  + It indicates that use case a and b may or may not be invoked. 
* Generalisation- models conceptual similarity between use cases.
  + If use cases have common behaviour, structure and similarities, their common parts can be factored out into a parent use case to optimise the model
  + generalization relationship between parent and child use cases is one in which the child is a more specialised form of the parent use case
  + ‘Handle Sale Call’ or ‘Handle Technical assistance call’ substitutes ‘Handle Call



**Use Case Description**



provides the details of the functionality that the system will support and describes how the actors will use the system

Writing: Don’t focus on perfection – be productive, Not about getting it right the first time, it is iterative process

1. Identify the Actors
2. Identify the Goal
   1. From the high-level scenario
   2. By brainstorming
   3. By asking: What does this Actor want to do?
3. Define the Pre-conditions
   1. before the Use Case can start
   2. Identify Pre-Conditions by asking: “What must be in place for the Use Case to begin?”
   3. Examples: User account exists, User has enough money in their account, There is enough disk space
4. Define the Post-conditions
   1. The result, or successful outcome of Use Case
   2. Identify the Post-Conditions by asking: “What is the successful result of this process or Use Case?
   3. Examples: Money was transferred to the user account, User is logged in, The file is saved
5. Describe the ‘Main Flow’
   1. Primary Scenario / Happy Day Scenario
      1. The simplest sequence – everything goes right Starts with pre-conditions and ends with post conditions
   2. Describe the Main Flow by asking: “What must happen to achieve the goal / outcome?”
   3. Example: 1. Admin enters Course name, code & description 2. System validates Course code 3. System adds course to the database and shows confirmation message
   4. Identify opportunities for reuse
   5. Only one side (actor or system) should do something in a single step
   6. Each step should lead to some progress
   7. Use simple sentences: Actor asks for money
6. Describe the Alternate Flows
   1. Describes the variations/exceptions to the Main Flow
      1. An additional flow, not necessarily error based
      2. An exception or error flow to any line item in your basic flow
   2. Asking: “What might affect this Use Case? “What could go wrong?”
   3. Examples: – While a customer places an order, their credit card failed – While a customer places an order, their user session times out

**How do you know when you have identified all your Use Cases**

* When all actors are specified
* When every functional requirement has a use case which satisfies
* The CRUD (Create, Read, Update, Delete)
* Use Cases and CRUD Technique
* For each type of data (data entity or domain class), verify that a use case has been identified that creates a new instance, updates existing instances, reads or reports values of instances, and deletes (archives) an instance 