System Analysis And Design

Lecture (3)

2024

What After Gathering Information and requirements?

Requirements

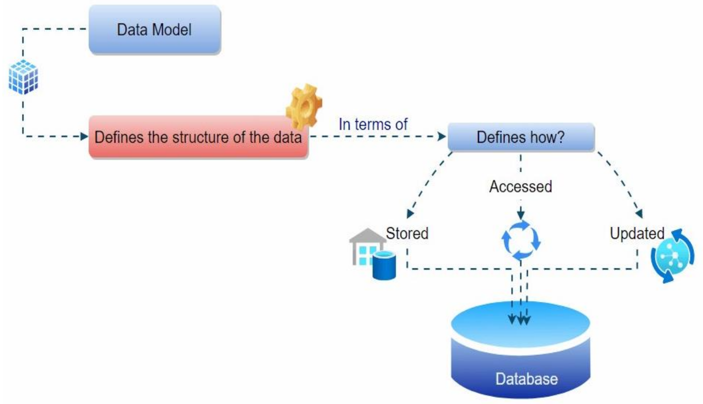
Model

Definition data modeling :

- Data modeling refers to the technique of representing the data requirements and structures of a system in a detailed and organized manner.
- It involves identifying the data entities (such as objects, events, or concepts), their attributes, relationships between entities, and constraints that govern the data within the system.

Benefits of using data model in SAD:

- Improved understanding of the system
- Better system design
- Reduced risk of errors
- Improved communication



Data and Process Model

Object Oriented

- *Object-oriented (O-O) analysis* combines data and the processes that act on the data into things called objects.
  - USE case – class diagram

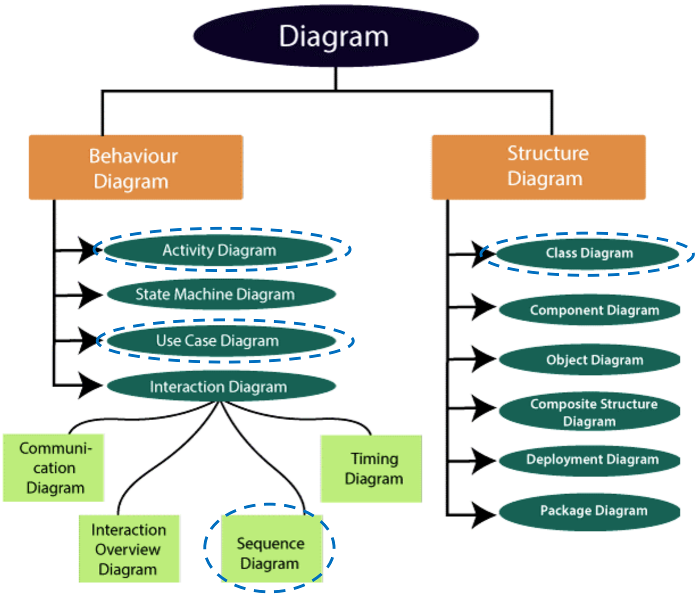
Structure

- *Structured analysis* identifies the data flowing into a process, the business rules that transform the data, and the resulting output data flow.
  - DFD Data flow Diagram
  - Relationship diagrams (ERD)

UML

Unified Modelling Language

UML Diagrams



Diagram

Behaviour Diagram

Activity Diagram

State Machine Diagram

Use Case Diagram

Interaction Diagram

Communi-cation Diagram

Interaction Overview Diagram

Sequence Diagram

Timing Diagram

Structure Diagram

Class Diagram

Component Diagram

Object Diagram

Composite Structure Diagram

Deployment Diagram

Package Diagram

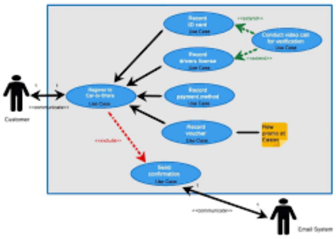
UML

Unified Modelling Language

Use Case Diagram

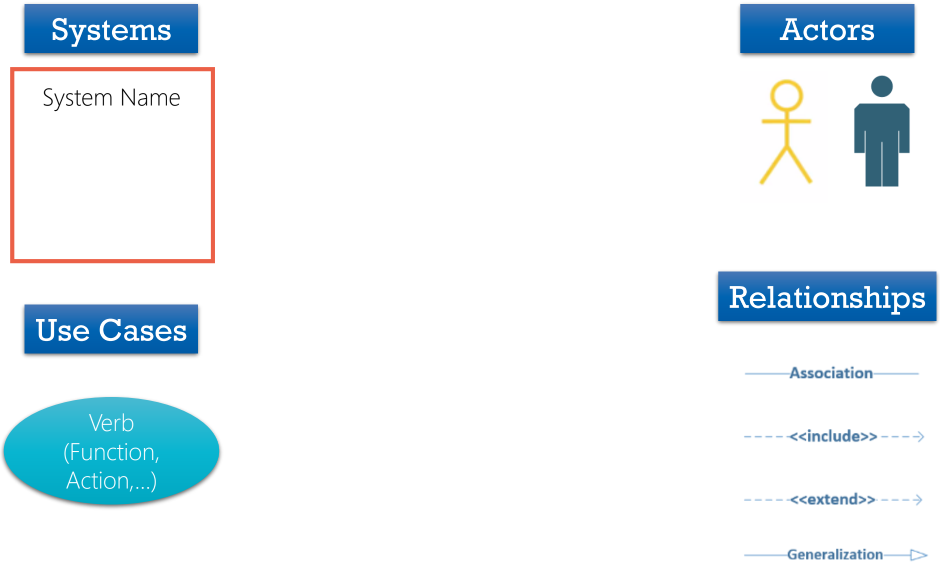
- ***A Use Case*** represents how a system interacts with its environment by illustrating the activities performed by the system's users and the system's responses.

- ***Use cases*** are created to help the development team more fully understand the steps involved in achieving the user's goals.


Record ID card Use Case
<<extend>>
Conduct video call for verification Use Case
Record drivers license Use Case
<<extend>>
Register to Car-to-share Use Case
Record payment method Use Case
Record voucher Use Case
New promo at Easter
<<include>>
Send confirmation Use Case
<<communicate>>
Customer
<<communicate>>
Email System

- ***The goal*** is to create a set of use cases that describe all the tasks that users need to perform with the system.

Use Case Diagram Elements



Systems

System Name

Use Cases

Verb (Function, Action,...)

Actors

Relationships

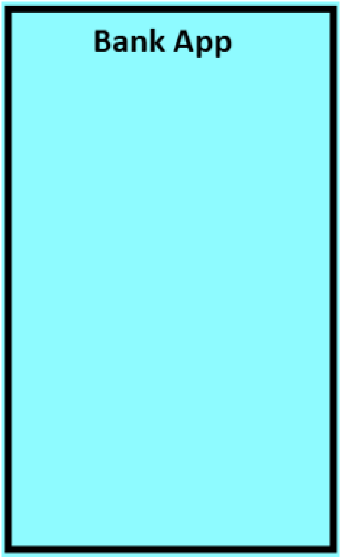
Association

<<include>>

<<extend>>

Generalization

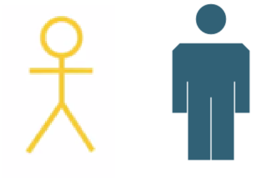
Systems

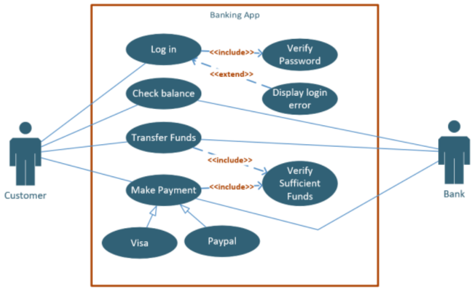


- System:
  - ➢ The software project that you will develop.
- Systems:
  - ➢ Website
  - ➢ Mobile app
  - ➢ Desktop app
  - ➢ Dashboard
- System:
  - ➢ The system is presented as rectangle.
  - ➢ The system name is written in the top of rectangle.

Actors

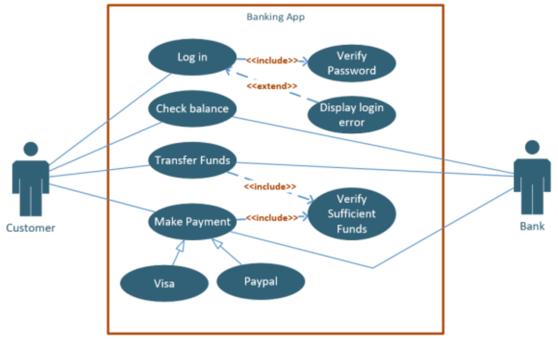
- Actor:-
  - ➢ Any one will use the system for specific goal.
- There is *two types* of Actors (*Primary , Secondary*).
- Primary Actor:-
  - ➢ Initiate the use of system (Customer).
  - ➢ The Primary actor placed to the left of system.
- Secondary Actor:-
  - ➢ The user that will react with system (Bank Employee).
  - ➢ The Secondary actor placed to the right of system.





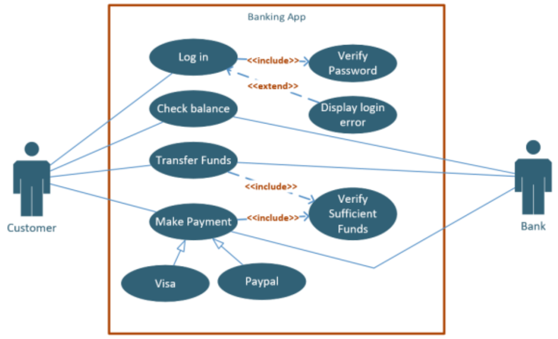
Use cases

- Use cases:-
  - ➢ Presents a set of actions, services, and Functions that the system needs to perform.
- The use case presented as ***oval*** and must be ***inside*** the system.



Use cases

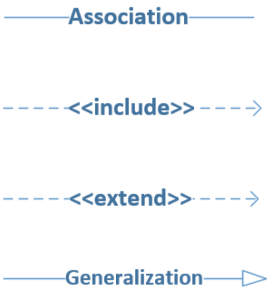
- The use cases ***start with verb***.
- Use cases have to be ***sorted*** according to the actions that the actor will do in system.
- Each actor has to interact with ***at least one*** of the use cases in the system.



Relationships

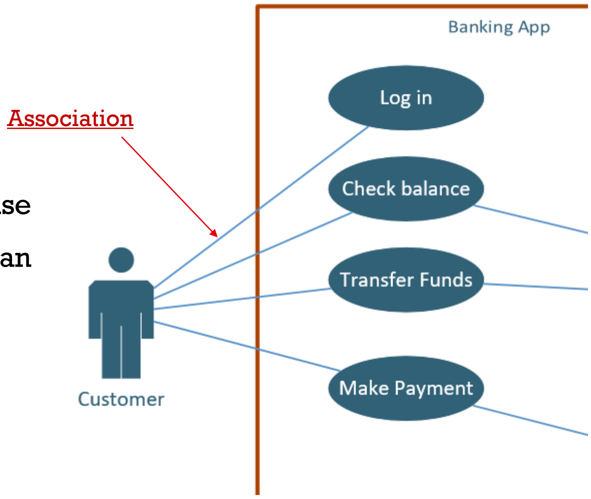
Relationship types:-

- Association
- Include
- Extend
- Generalization



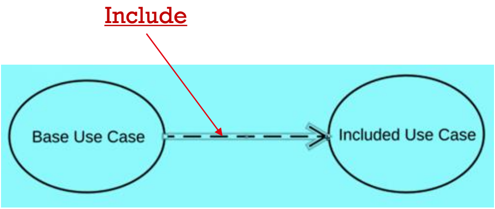
Relationships

- Association:-
  - The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link.

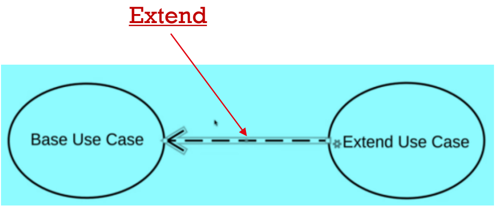


Relationships

- Include:-
  - ➢ When a use case is depicted as using the functionality of another use case, the relationship between the use cases is named as ***include or uses*** relationship.

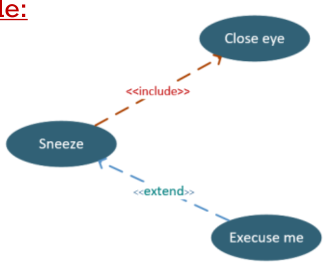


- Extend:-
  - ➢ Indicates that an use case may include (subject to specified in the extension) the behavior specified by base use case.
  - ➢ The extend use case *will only happen if certain criteria are met.*

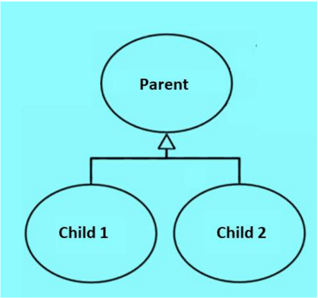


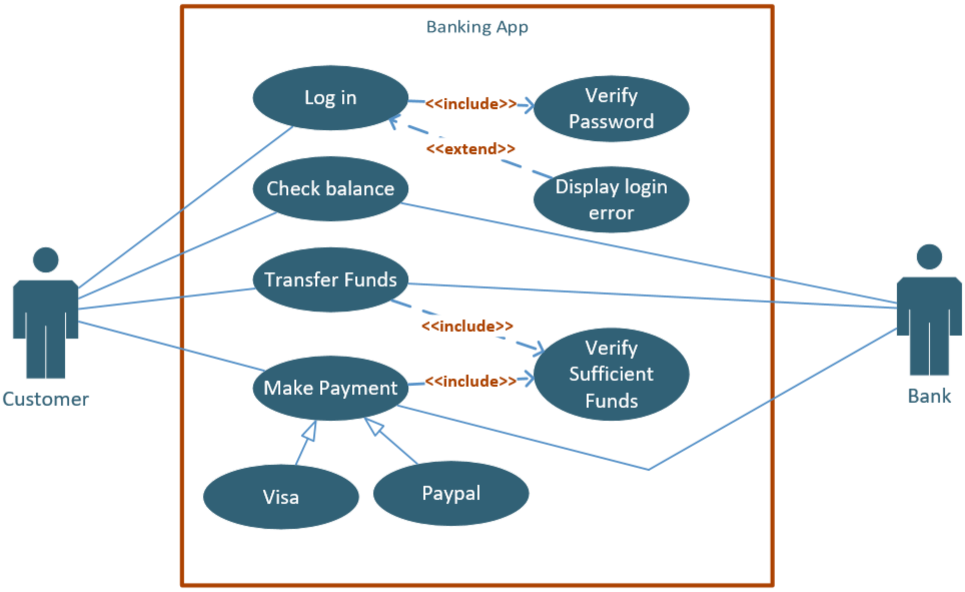
Relationships

- Include and Extend Example:



- Generalization :-
  - A generalization relationship is a parent-child relationship between use cases.

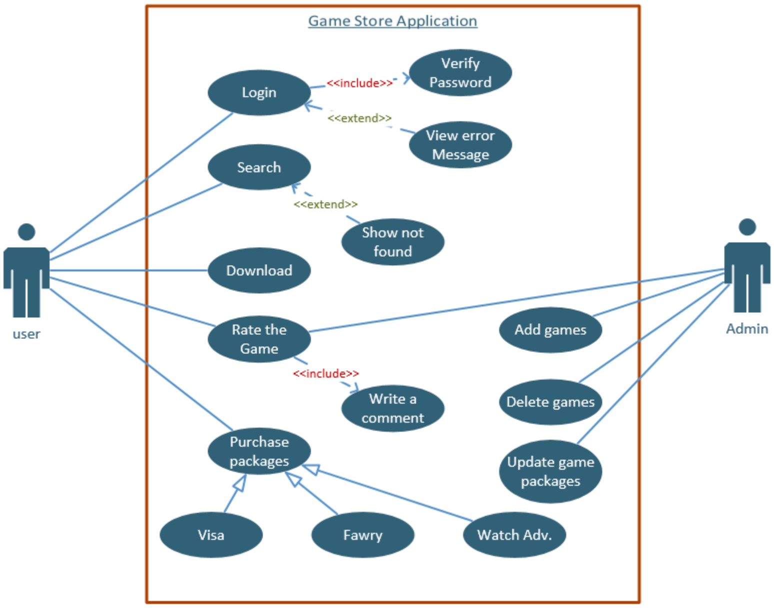




Case Study: Game Store Application

Scenario :

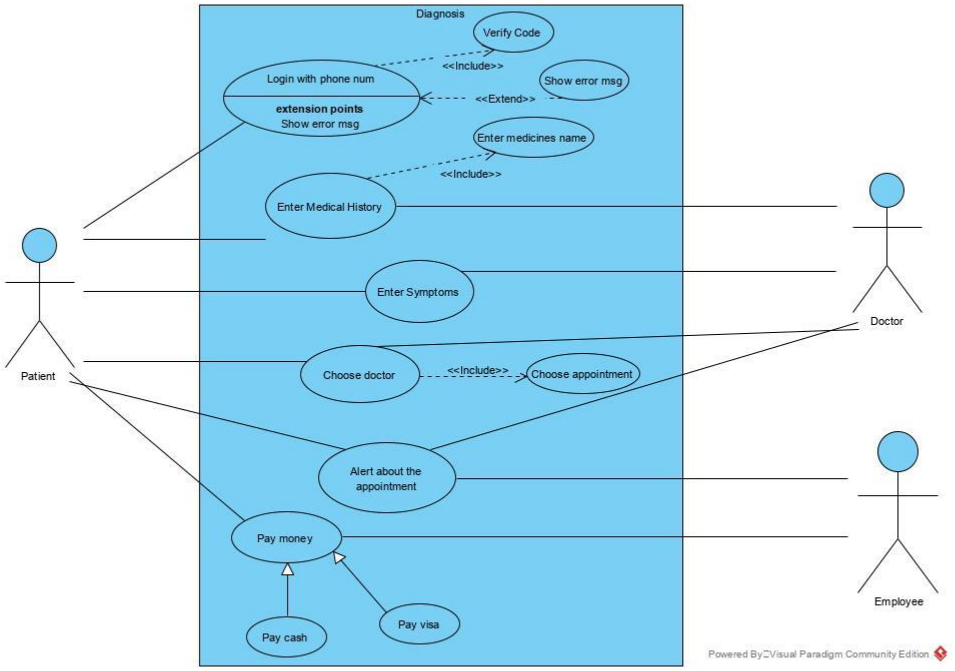
Design UML use case diagram for game store application. The **user** *login* to the application with an account. The **user** *search* for a specific game and *download* it. The **user** can also *purchase some game packs using (VISA, FAWRY, and watch advertisements)*. The **user** can also *rate the game*. The **admin** updates the game appearance according to users' rates. The **admin** can also (*add games, delete games, and update game packs*).



Case Study: Hospital Diagnostic system

Scenario :

Design UML use case diagram for a diagnostic system in a hospital. The ***patient*** *login* the system and give general information about his *medical history (medicine)*. The system ask users about *symptoms* in specific sequence due to the user answer for each question. The system *choose the* ***doctor*** after analyzing the patient data and *reserve an appointment*. The system also send a copy of *patient info to the chosen* ***doctor***. The patient *choose to pay the bill* (cache or by visa).



Case Study: Fix Car System

Scenario :

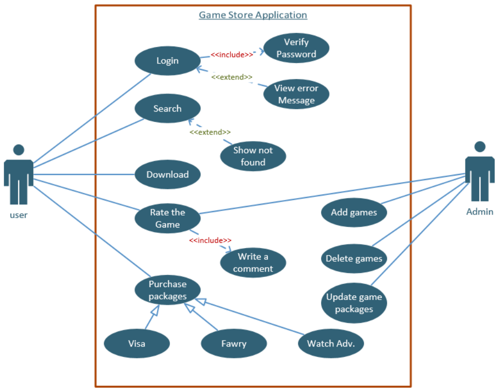
The customer can interact with the fix car system to create an account, request a repair, receive quotes, choose a mechanic, schedule an appointment, drop off their car, pick up their car, and pay for the repair. The fix car system provides the functionality for the customer to perform these actions.

1. the customer needs to be able to track the status of their car repair. This is important because it allows the customer to know when their car will be ready and to plan accordingly.
2. the mechanic needs to be able to access the customer's warranty information. This is important because it allows the mechanic to determine whether the repair is covered by warranty..

Summary

Use Case:

- Represents how a system interacts with its environment.
- Illustrates the activities that are performed by the users and the system's responses.
- Activities produce some output result.
- Each use case describes how an external user triggers an event to which the system must respond.



Thank You