Practical 7

AIM :To study UML user case diagram (ucd)

Introduction

- Use case diagram are dynamic in nature, dynamic behaviour
- It consists of internal and external factors for making the interaction
- These internal and external agents are known as actors
- It consists of actors, use cases and relationships

Purpose of use cases

- It's used to gather requirements of a system
- Used to get an outside view of a system
- It shows the interaction among the requirements and the actors

How to draw Use case diagram

- We have to first identify the actors actors can be defined as something that interacts with the system
- The actors can be human users some internal applications or may be some external applications

Where to use UCD

- It specify the events of a system and their flows but use case diagram never describe how they are implemented
- UCD can be an image as a black box where only the input output and the functions of the black box
- UCD can be in forward engineering process and backward engineering process

System boundary

- It defines the limit of a system
- It is represented by rectangular box
- Use case are inside the boundary
- Actors are outside the boundary

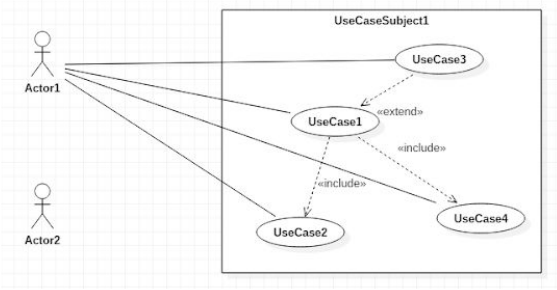

Fig 7.1

Relationships in UCD-

- Multiplicity of associations
- Multiplicity of an actors

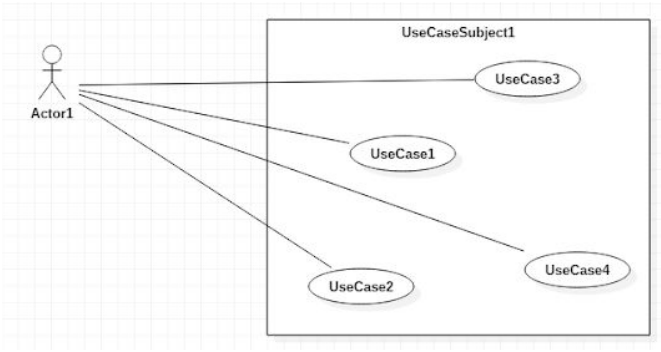

Fig 7.2

NOTE- only binary associations are allowed in actors and use cases

Include

- When a use case is depicted as using the functionality of another use case this relationship between the use cases is named as the include relationship
- The directed arrow having a dotted line
- The tip of the arrowhead points to the child use case and the parent use case is connected at the base of the arrow
- The stereotype "<<include>>" identifies the relationship as the

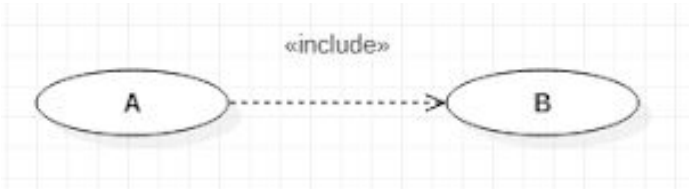

Fig 7.3

Extend

- Extend relationship between the child case adds to the existing functionality and characteristics of parent use case
- A directed arrow having a dotted line (same as include)
- The tip of the arrow have points to the parent use case and the child use case is connected at the base of the arrow

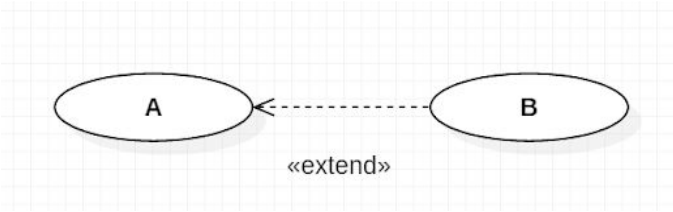

Fig 7.4

Generalisation

- Generalization is used when you find two or more use cases that have commonalities in behavior, structure, and purpose.
- When this happens, you can describe the shared parts in a new, often abstract, use case, that is then specialized by child use cases.

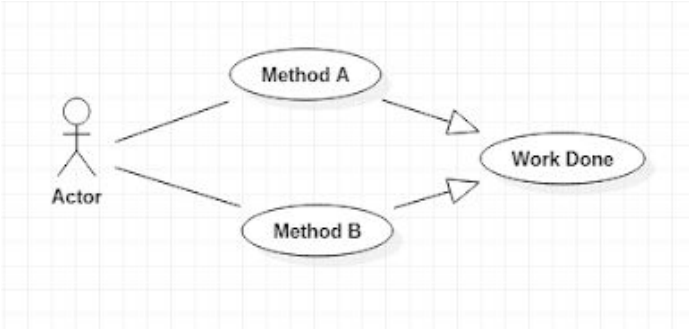

Fig 7.5

Practical - Use Case Diagram on ATM Machine

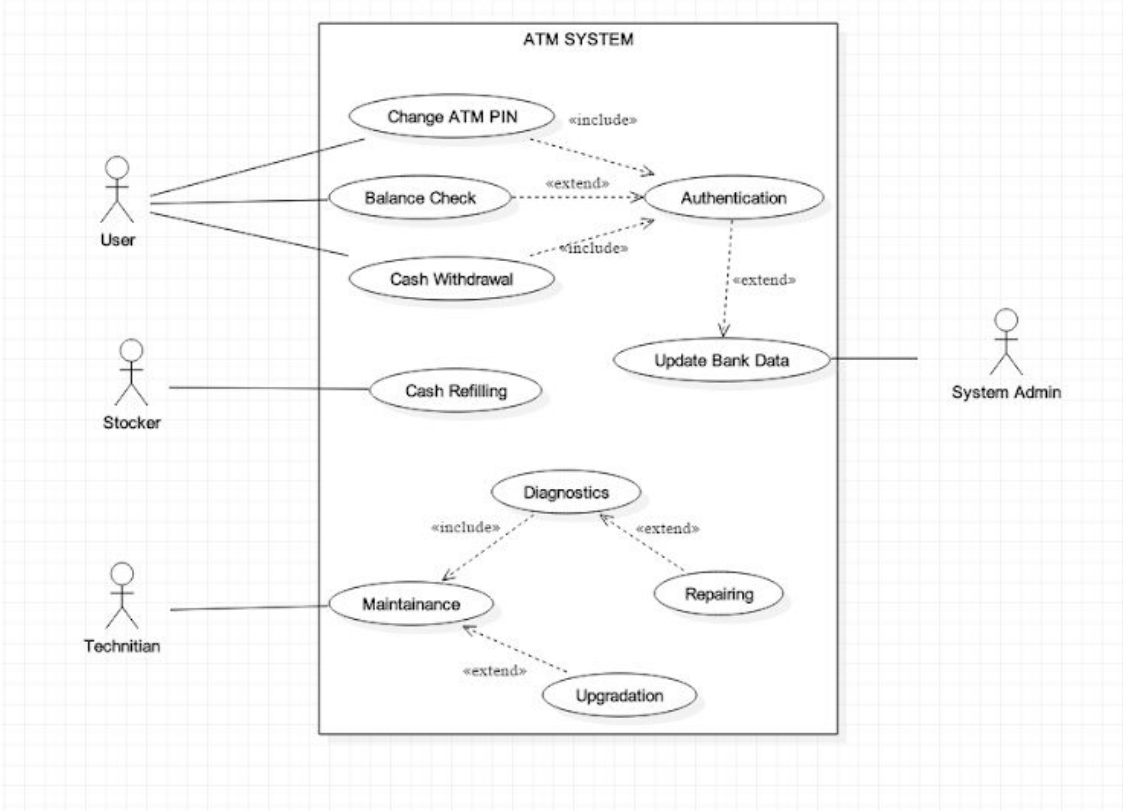

Fig 7.6