UNIT-3

Dynamic Modeling.

System Sequence diagrams:-

A sequence diagram simply depicts interaction between objects in a sequential order (ie) the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram.

Use cases describe how external actors interact with the system. During this interaction the actors generate the events to the system. Due to which the system performs some operations in some sequence.

System sequence diagram is represented as flow of system events, system actions and messages between the objects or the components of the system.

Sequence diagram Notations:- 1. Actor

We use actors to depict various roles including human users & other external subjects.

An actor in a UML diagram represents a type of role where it interacts with the

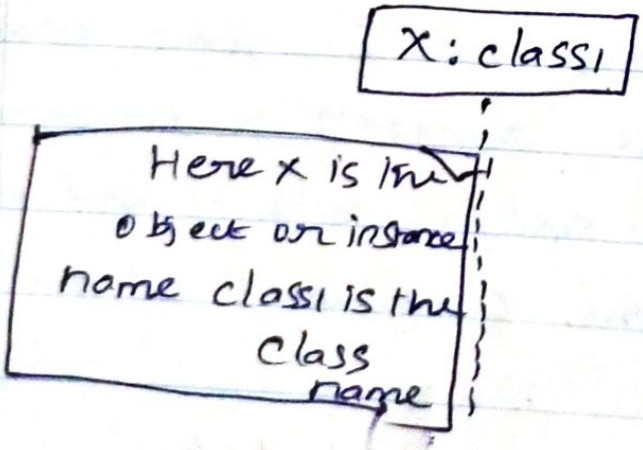
1. One particular scenario of use case.
2. The events that are generated by the external actors
3. Sequence of those events.
4. Events among the interacting objects of the system.

System and its objects.

2) Lifelines:-

A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format.

InstanceName: ClassName

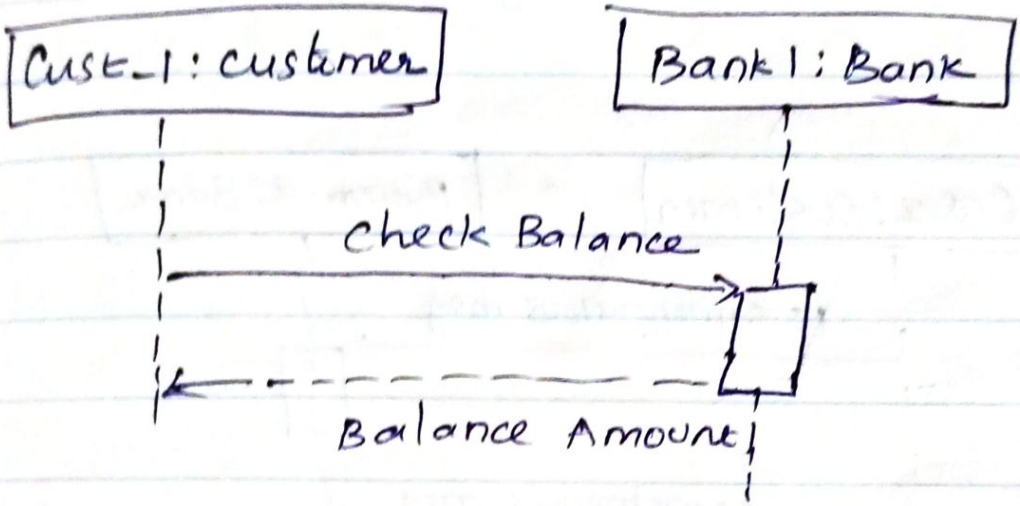


We display a lifeline in a rectangle called head with its name and type. The head is located on top of a vertical dashed

line (referred to as the stem).

Difference between a lifeline and an actor:-

A lifeline always portrays an object internal to the system whereas actors are used to depict objects external to the system.



3) Messages:-

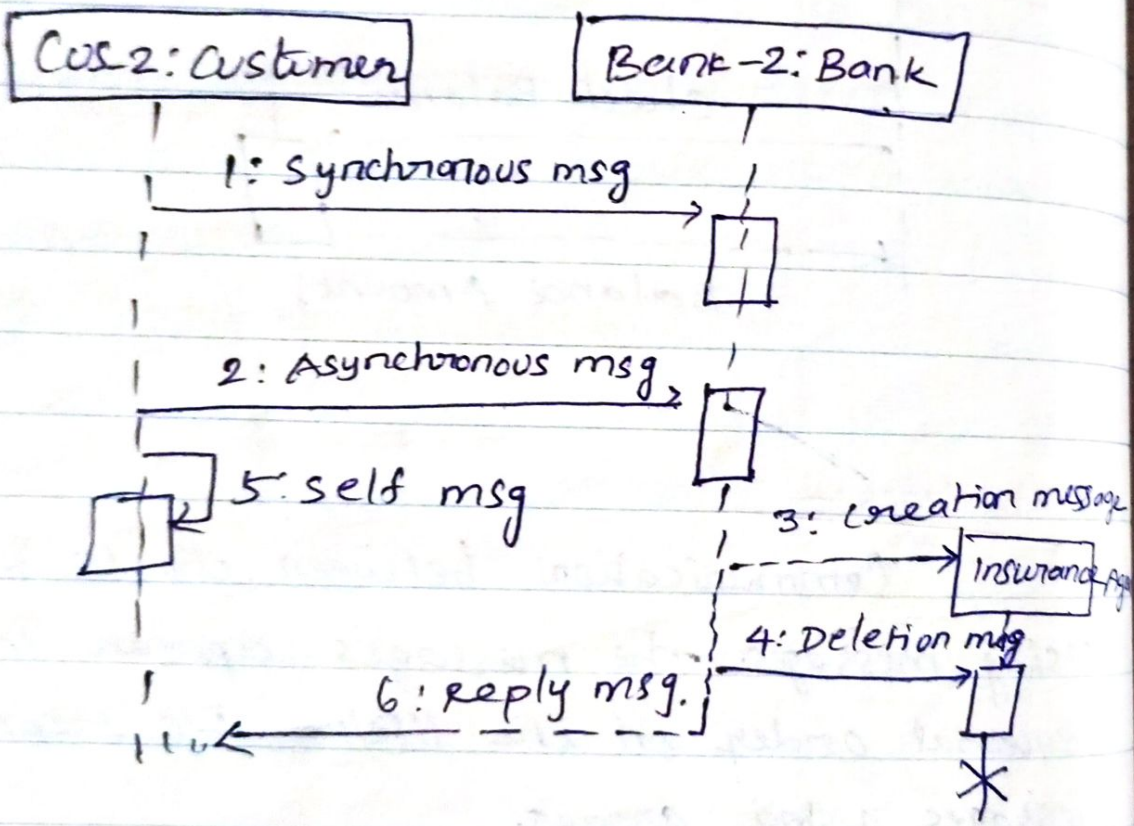
Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows.

Messages can be broadly classified into the following categories:

1) Synchronous messages
2) Asynchronous messages

3) Create message
4) Delete message
5) self message
6) Reply message
7) Found message
8) Lost message

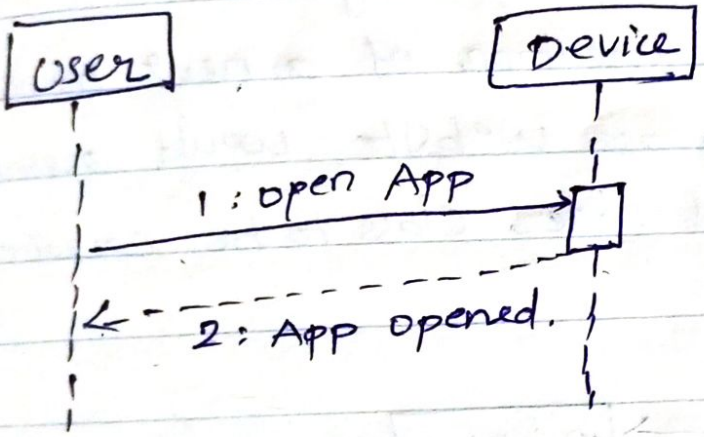
Ex:-



Synchronous messages:-

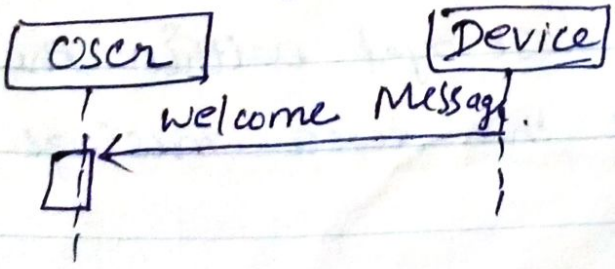
A synchronous message waits for a reply before the interaction can move forward.

The sender waits until the receiver has completed the processing of the message. The caller continues only when it knows that the receiver has processed the previous message (ie) it receives a reply message. A large number of calls in object oriented programming are synchronous. We use a solid arrow head to represent a synchronous message.



Asynchronous Messages:

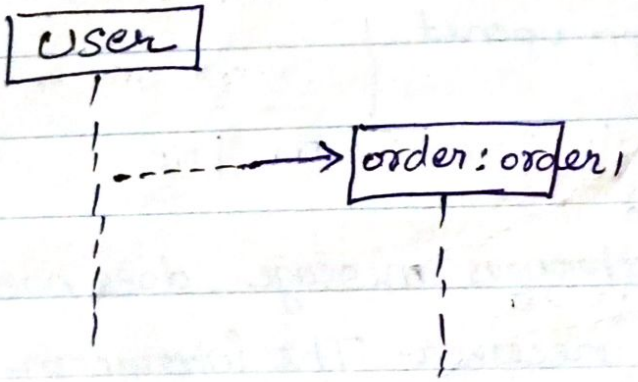
An asynchronous message does not wait for a reply from the receiver. The interaction moves forward irrespective of the receiver processing the previous message or not. We use a lined arrow head to represent an asynchronous message.



Create message:

Create message to instantiate a new object in the sequence diagram. There are situations when a particular message call requires the creation of an object. It is represented with a dotted arrow and create word labelled on it to specify that it is the create message symbol.

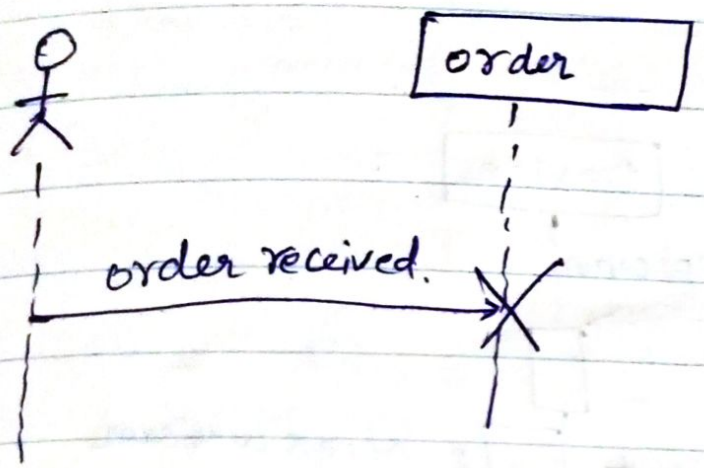
For Ex:- The creation of a new order on a e-commerce website would require a new object of order class to be created.



Delete Message:-

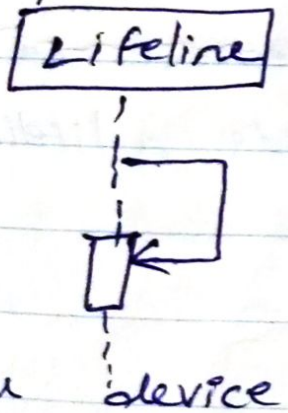
We use a Delete message to delete an object. When an object is deallocated memory or is destroyed within the system we use the delete message symbol.

It destroys the occurrence of the object in the system. It is represented by an arrow terminating with ax.



self message:-

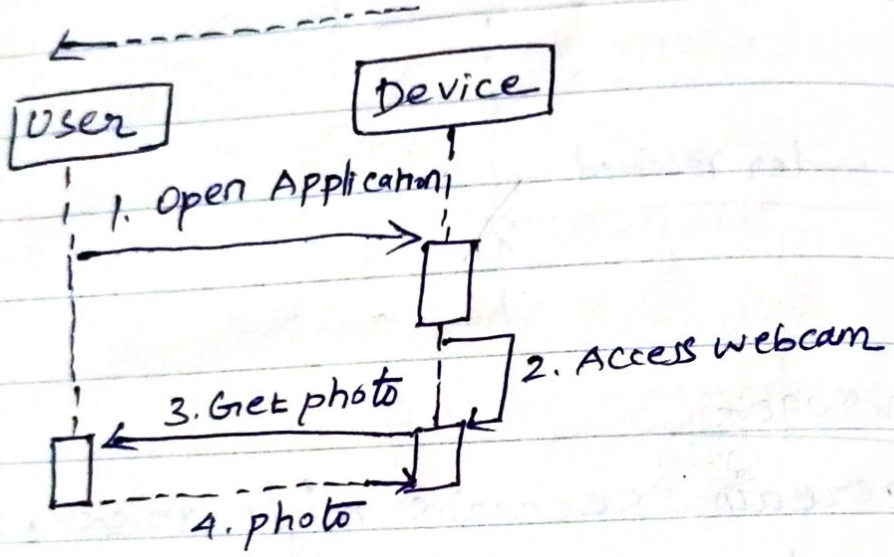
certain scenarios might arises where the object needs to send a message to itself. such messages are called self messages & are represented with a U shaped arrow.



where the device wants to access its webcam. Such a scenario is represented using a self message.

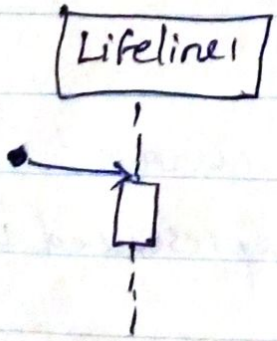
Reply message:

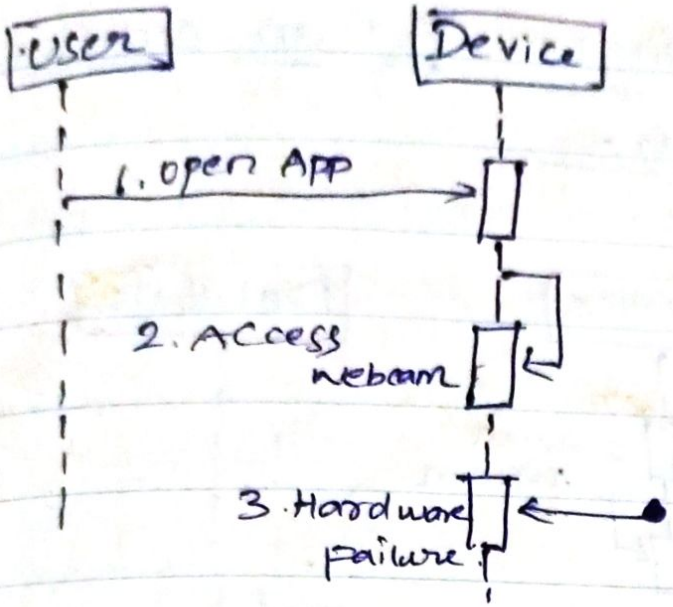
Reply messages are used to show the message being sent from the receiver to the sender.



Found Message:

A found message is used to represent a scenario where an unknown source sends the message. It is represented using an arrow directed towards a lifeline from an endpoint.



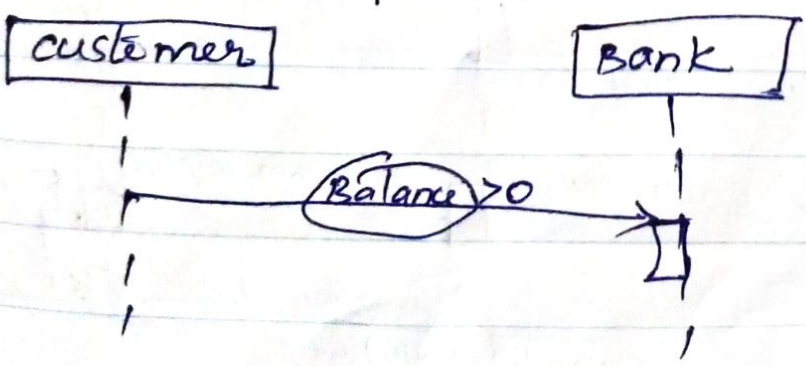


Lost message:-

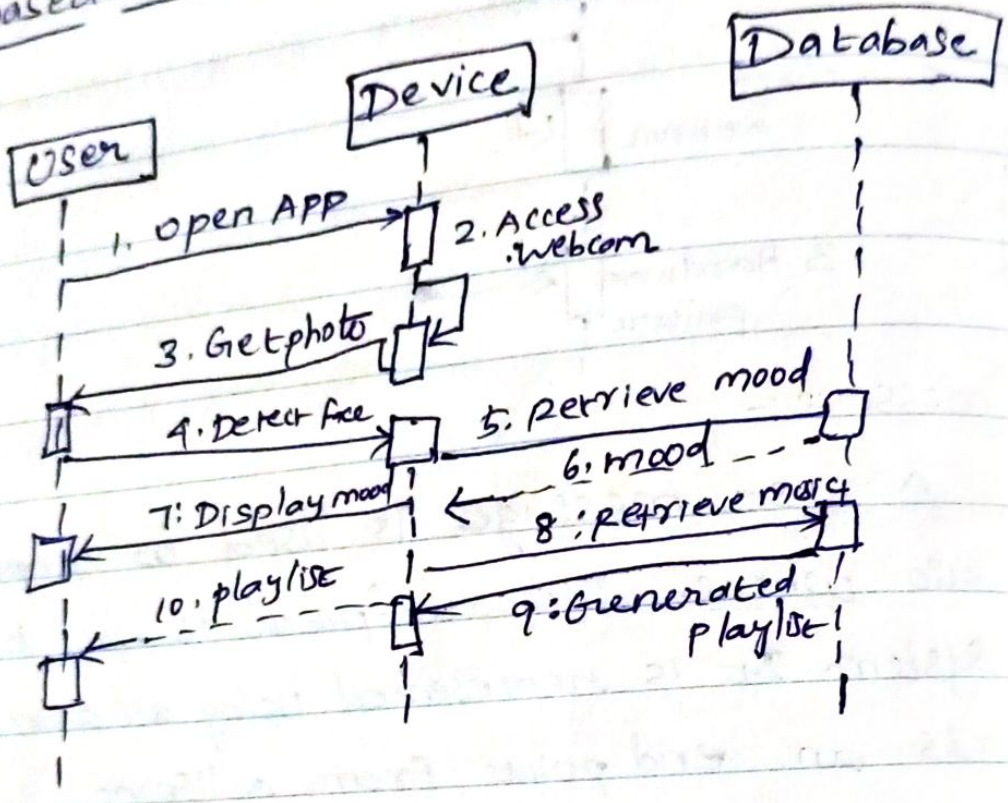
A Lost message is used to represent a scenario where the recipient is not known to the system. It is represented using an arrow directed towards an end point from a lifeline.

4. Guards:-

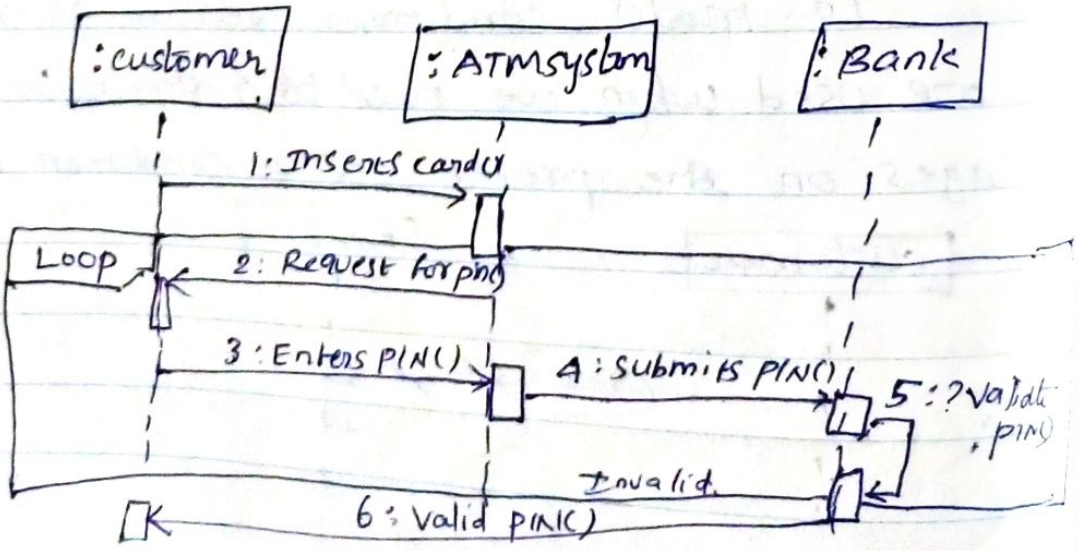
To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met.

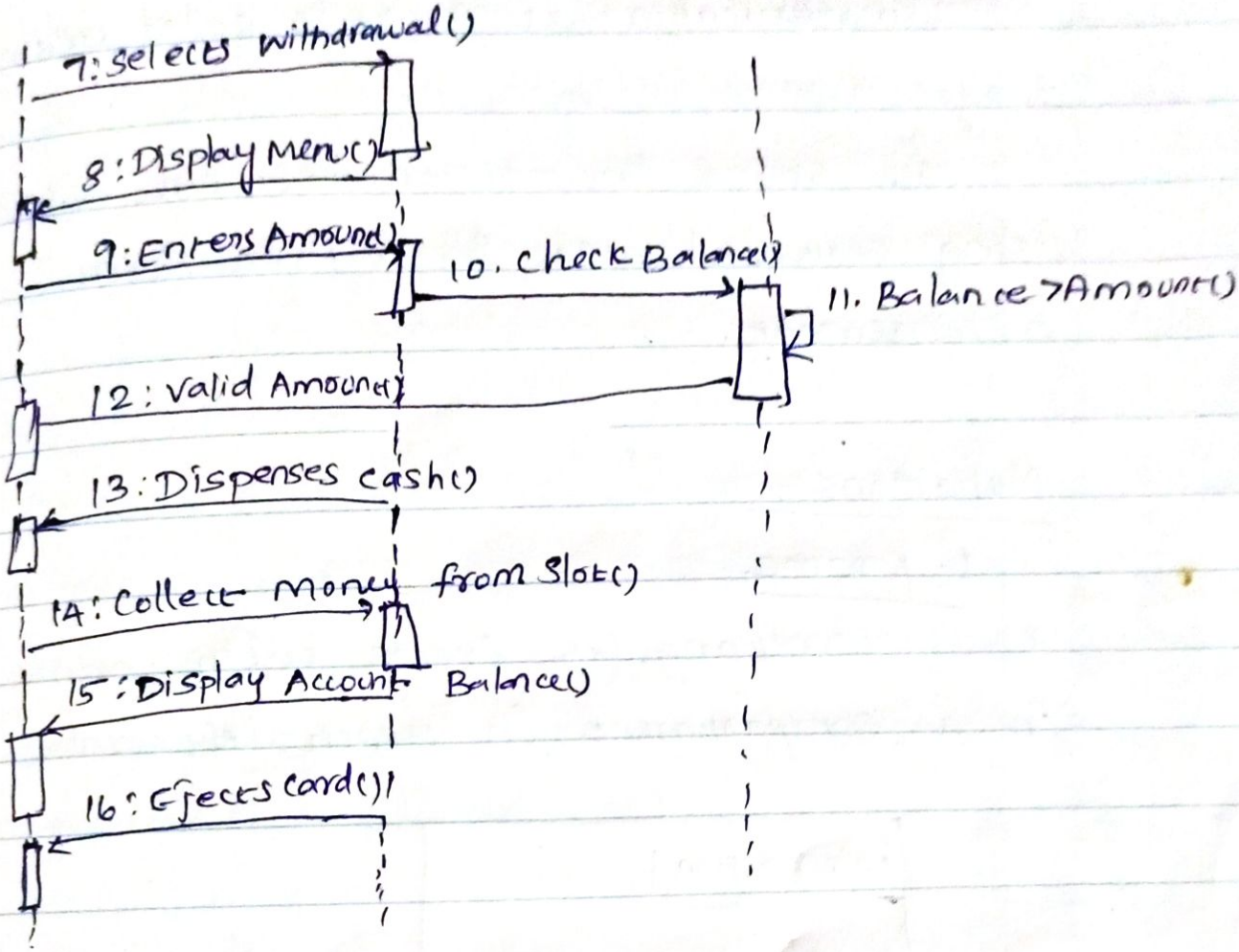


A seaverce diagram for an emotion based music player:-



ATM System sequence diagram





UML Interaction Diagram:-

The interaction diagrams are created to represent interactive behavior of the elements of the system.

1. sequence diagram
2. Communication diagram.

Communication diagram:

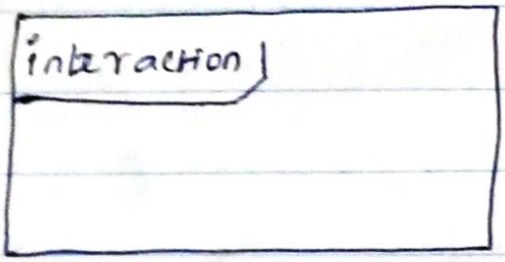
Communication diagram is called collaboration diagrams.

It shows interaction between objects using sequenced msgs in a free form arrangement.

Notations:-

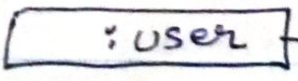
1. Frame

rectangular frame with name in a compartment in upper left corner

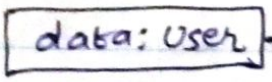


2. Lifeline:

Individual participant in the interaction.



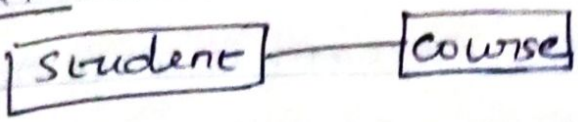
→ Anonymous lifeline



→ data of ols stack/user

3). Links

A link is a connection path between two objects. It indicates navigation and visibility between the objects. Ex:-

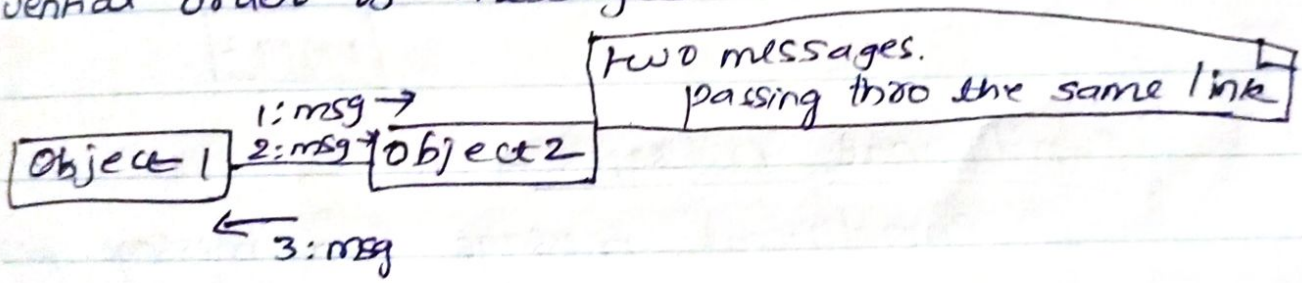


Messages:-

A line with sequence expression & arrow above the line.

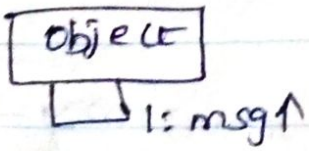
→ arrow indicates the direction of the communication

→ A sequence number is added to show the sequential order of messages.

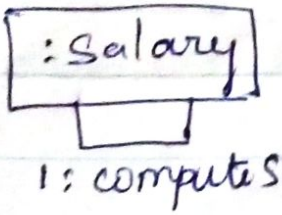


Message to 'self' or this:-

A message can be sent from an object to itself.



Ex:

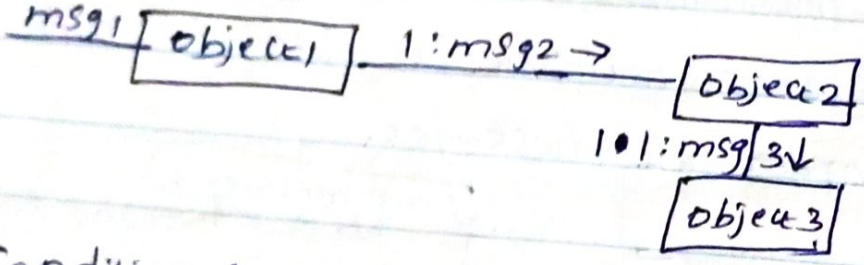


Message number sequencing:-

The numbers of message indicates the order of message within an interaction.

1. The first message is not numbered.
(ie) msg1 is unnumbered.

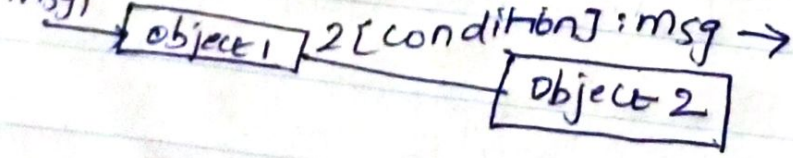
2. The order and nesting of subsequent messages is shown with a legal numbering scheme in which nested messages have a number appended to them.



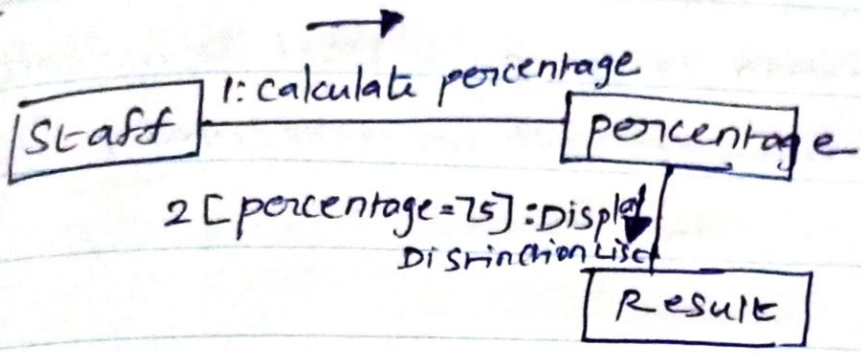
Conditional messages:-

Conditional message is a message passed between objects along with a condition to be checked before execution. Message is sent only if clause evaluates to true.

Notation:-

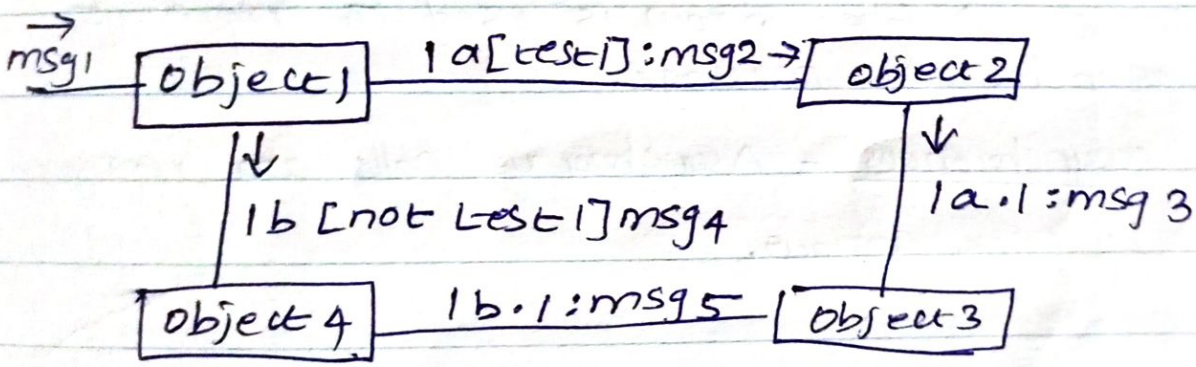


Ex:

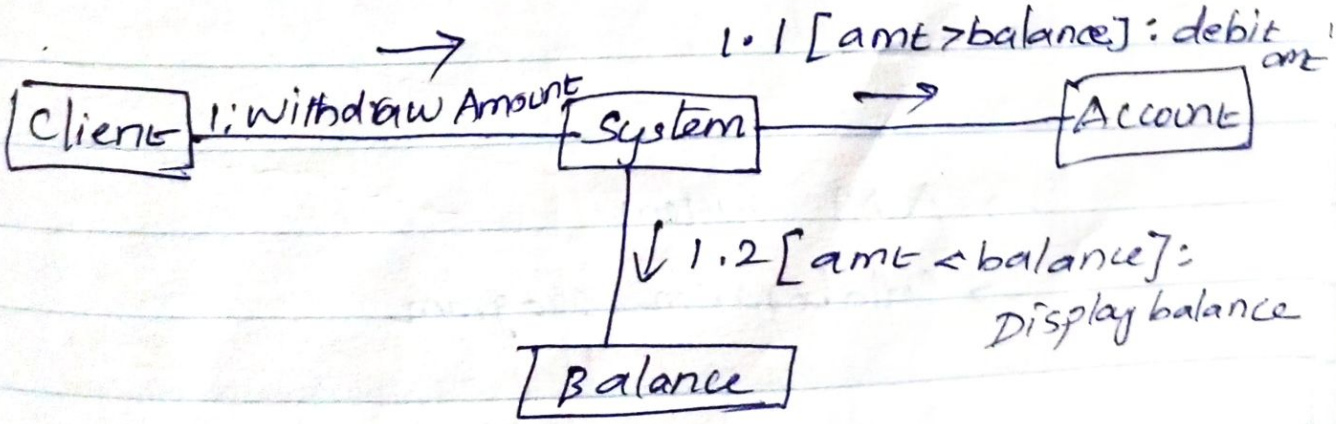


Mutually exclusive conditional paths:-

Mutually exclusive messages indicates that either one of the messages can be passed based on the condition.



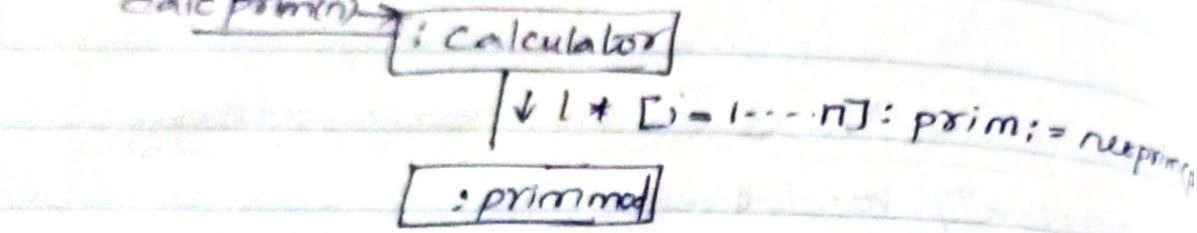
Ex:-



Iteration over a collection or Looping:-

Iteration or Loop is used to indicate certain operations will be executed n number of times.

Ex:-



Asynchronous and synchronous calls:-

A filled arrow is used to represent synchronous message.

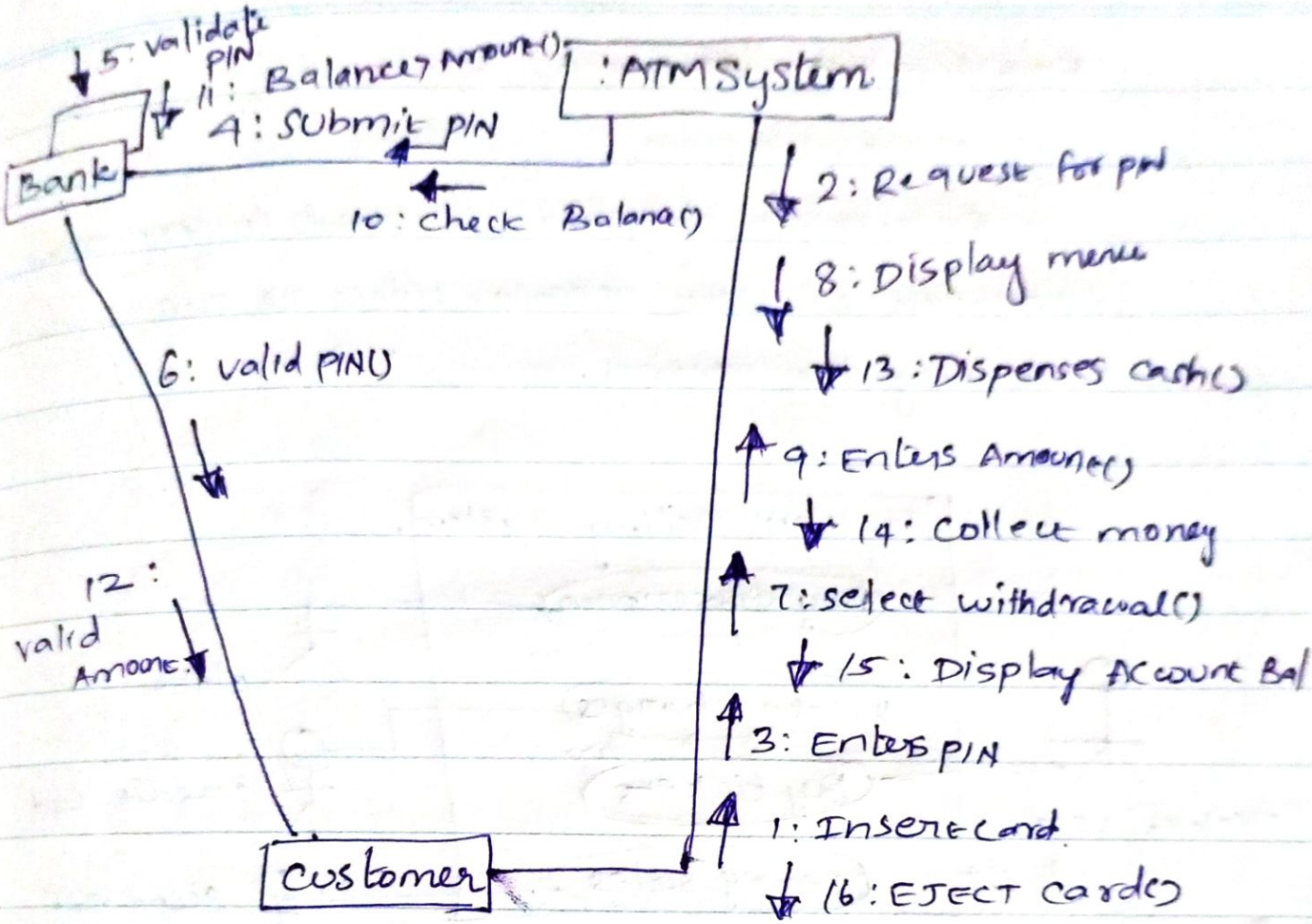
Asynchronous :- Asynchronous calls are represented by stick arrow.

Communication diagram:-

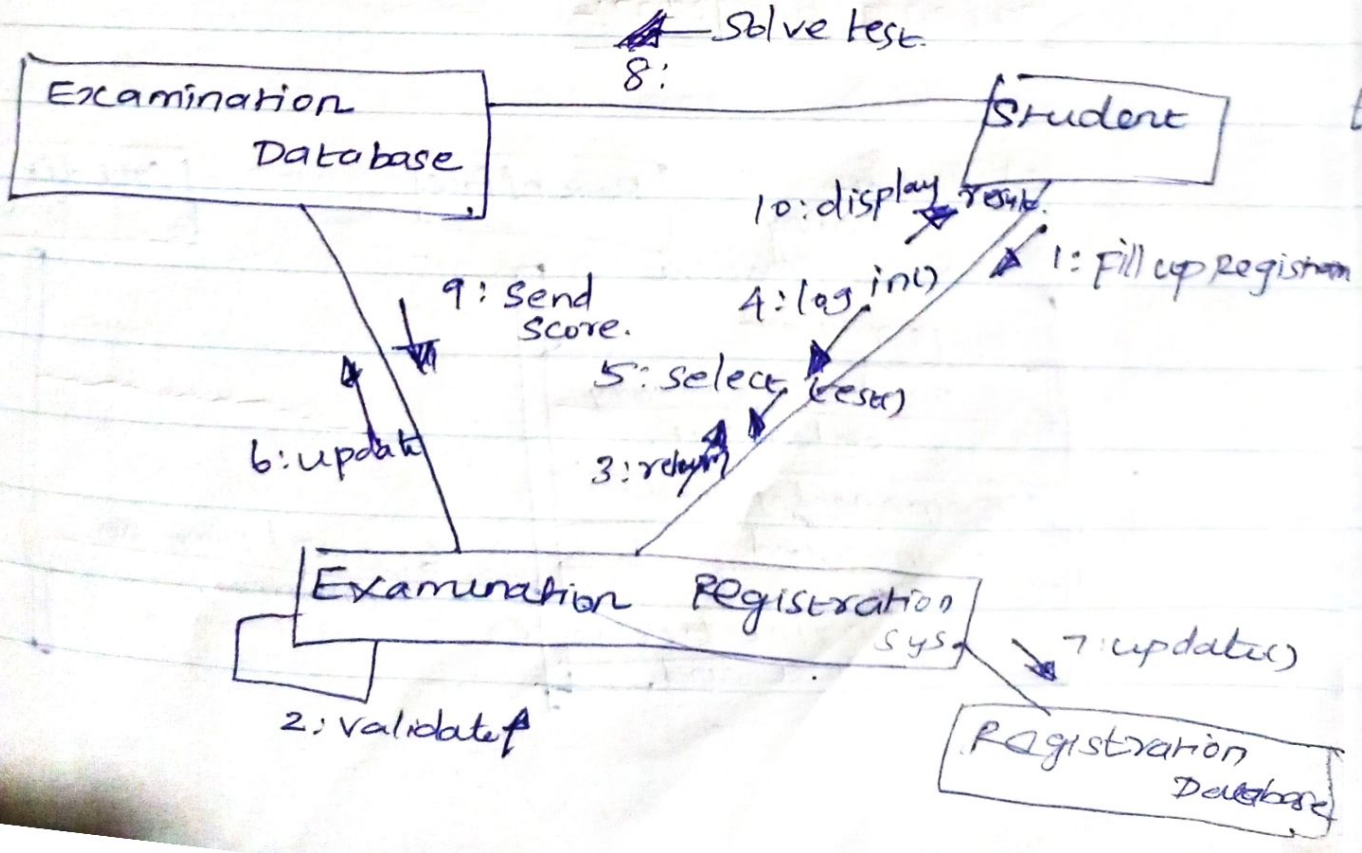
Ex:-

→ ATM System

→ Collaboration diagram



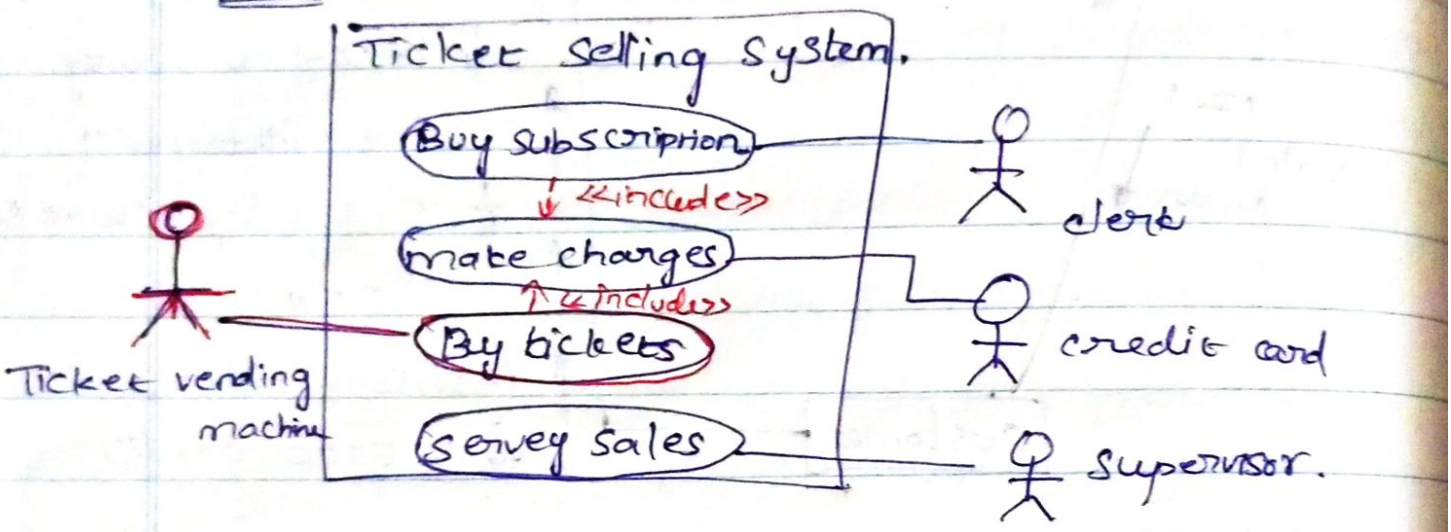
Exam registration:-



Relationship between Sequence diagram and Usecases diagram:

Sequence diagram is used to represent one scenario of the usecase. Hence it can be derived by observing the usecases.

Ex:



Sequence diagram:-