Sequence Diagrams in UML

Friday, December 14, 2018 11:21 AM

Sequence Diagrams :

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. The terms like event diagrams or event scenarios can also be used to refer to a sequence diagram.

They describe how and in what order the objects in a system function.

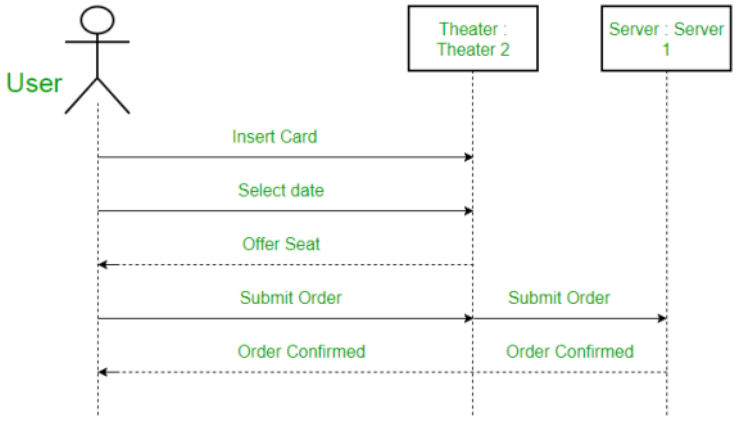
Notations:

1. Actors:

   An actor is used to represent a role to interact with the system and its objects. An actor is always outside the scope of the system.

   This is used to depict various roles including human users and other external subjects. It is represented using a stick person notation. It is possible to have multiple actors in a sequence diagram.

   Example : Seat reservation system (Note the actor is outside the system)

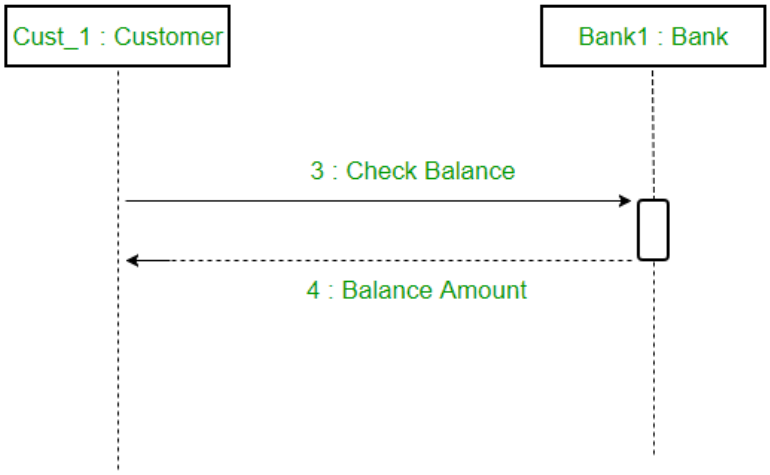


2. Lifelines:

   A lifeline is a named element which depicts an individual participant in a sequence diagram. The standard format to represent a lifeline is Instance Name : Class Name. Lifeline is represented in a rectangle called head with its name. They are located on top of a vertical dashed line.

   Difference between a lifeline and an actor is that lifeline always represents the objects internal to the system whereas the actor represents the object external to the system.

   Example for the lifelines:



3. Messages:

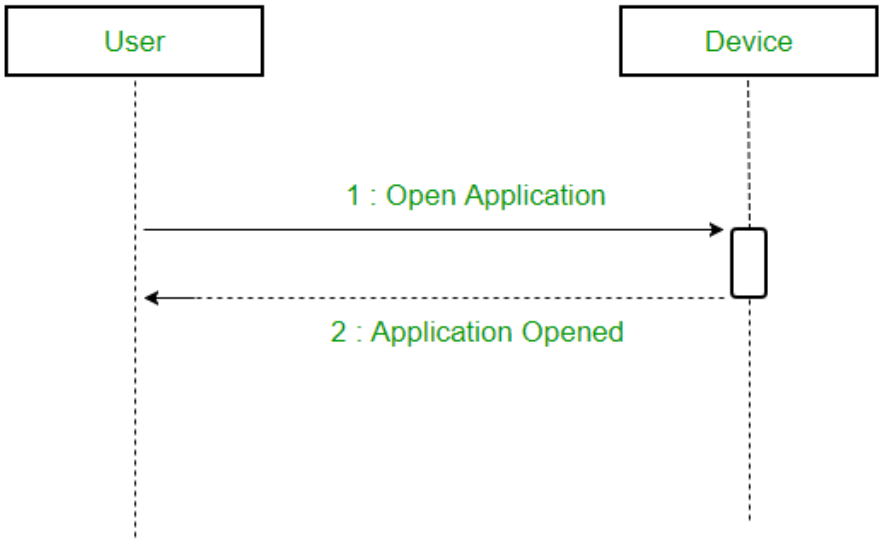
Communication between objects can be described using messages. The messages are represented using the arrows. They appear in a sequential order of the lifeline. Lifelines and Messages are the core of a system.

Different types of messages that can be used are explained with the examples below.

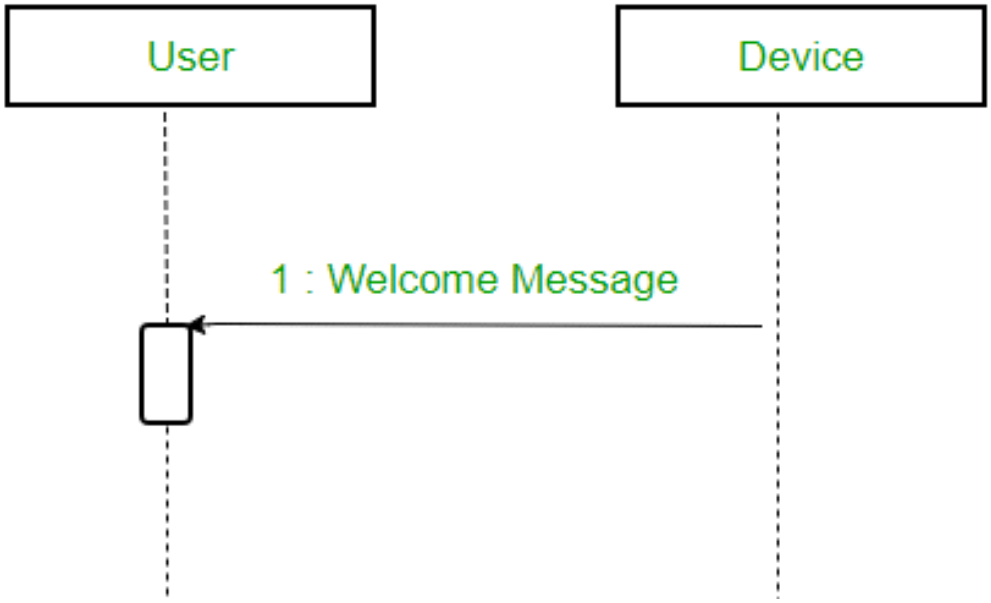
Message type	Description	Example
Create	A create message represents the creation of an instance in an interaction. The create message is represented by the keyword «create». The target lifeline begins at the point of the create message.	In a banking scenario, a bank manager might start a credit check on a client by sending a create message to the server.
Destroy	A destroy message represents the destruction of an instance in an interaction. The destroy message is represented by the keyword «destroy». The target lifeline ends at the point of the destroy message, and is denoted by an X.	A bank manager, after starting a credit check, might close or destroy the credit program application for a customer.
Synchronous call	Synchronous calls, which are associated with an operation, have a send and a receive message. A message is sent from the source lifeline to the target lifeline. The source lifeline is blocked from other operations until it receives a response from the target lifeline.	A bank teller might send a credit request to the bank manager for approval and must wait for a response before further serving the customer.
Asynchronous call	Asynchronous calls, which are associated with operations, typically have only a send message, but can also have a reply message. In contrast to a synchronous message, the source lifeline is not blocked from receiving or sending other messages. You can also move the send and receive points individually to delay the time between the send and receive events. You might choose to do this if a response is not time sensitive or order sensitive.	A bank customer could apply for credit but can receive banking information over the phone or request money from an ATM, while waiting to hear about the credit application.
Asynchronous signal	Asynchronous signal messages are associated with signals. A signal differs from a message because no operation is associated with the signal. A signal can represent an interrupt condition or error condition. To specify a signal, you create an asynchronous call	A credit agency could send an error signal message to the bank manager that states a failure to connect to the credit bureau.

	message and change the type in the message properties view.	
Lost and found	A lost message is a message that has a known sender but the receiver is not known. A found message is a message that does not have a known sender but has a receiver.	An outside actor sends a message to a bank manager. The actor is outside the scope of the sequence diagram and is therefore a found message. A lost message can occur when a message is sent to an element outside the scope of the UML diagram.

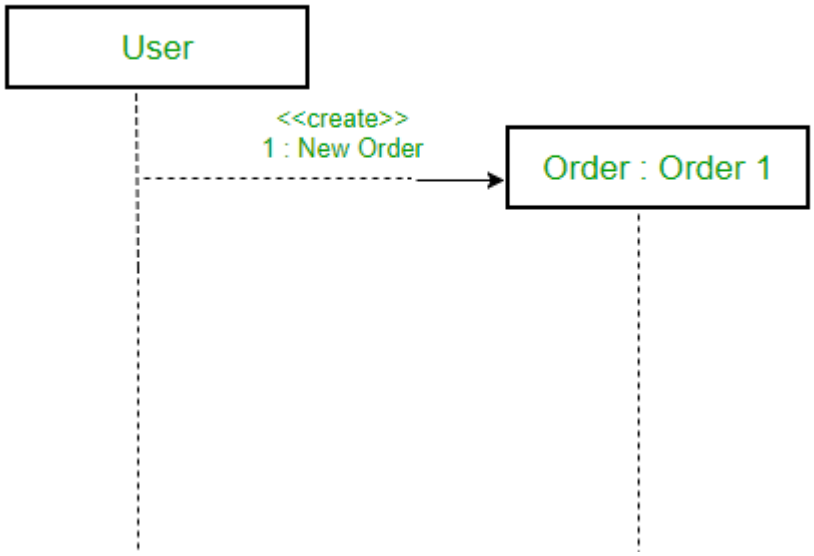
Example representations of the different messages are mentioned below:
Synchronous Message:



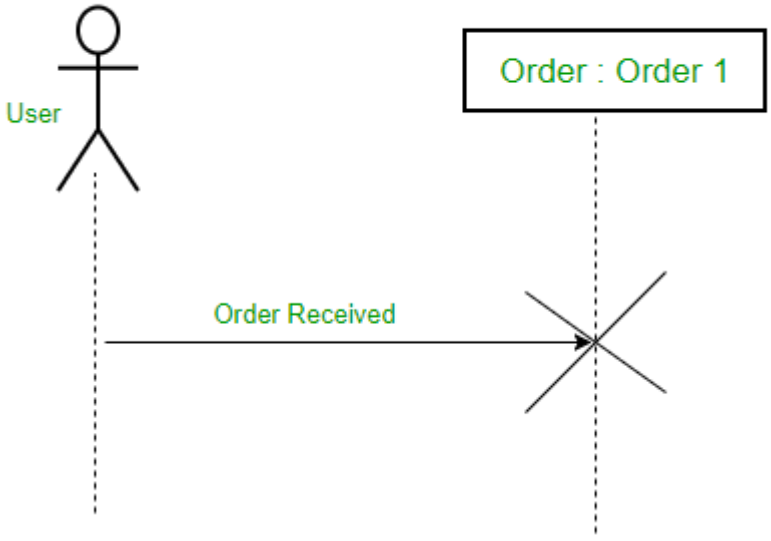
Asynchronous message:



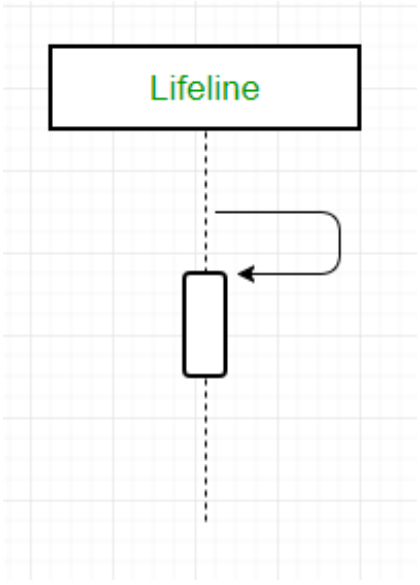
Create message:



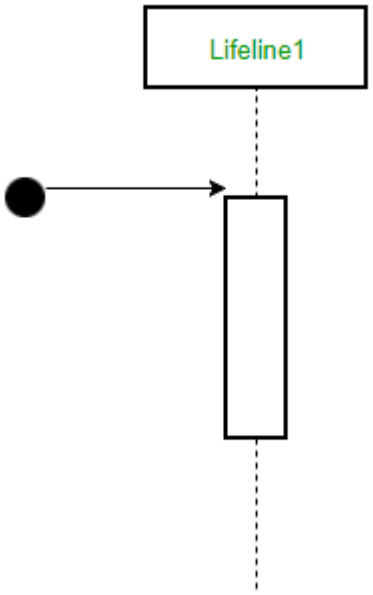
Delete Message:



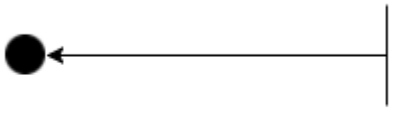
Self Message:



Found Message:

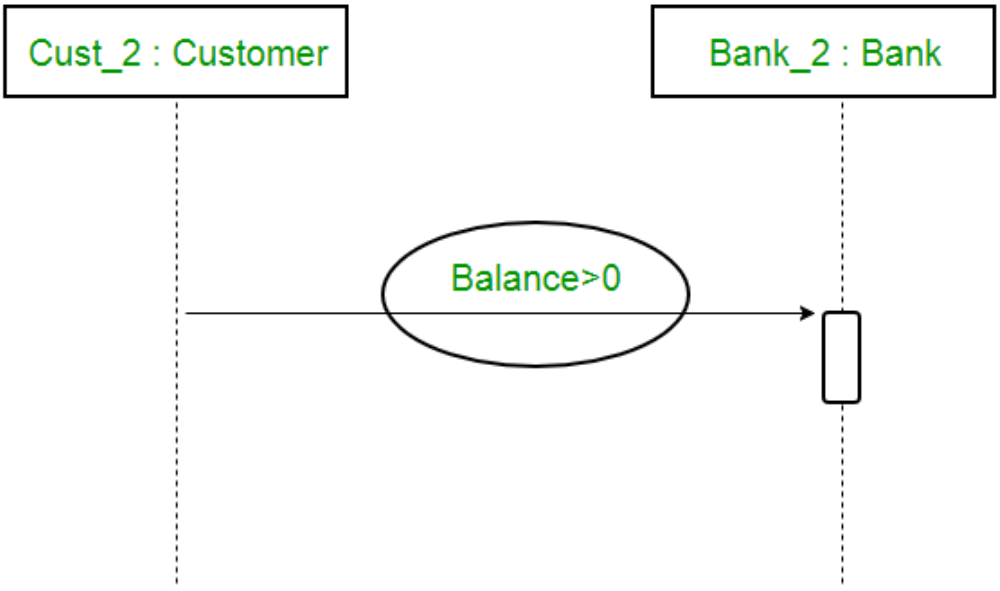


Lost Message:

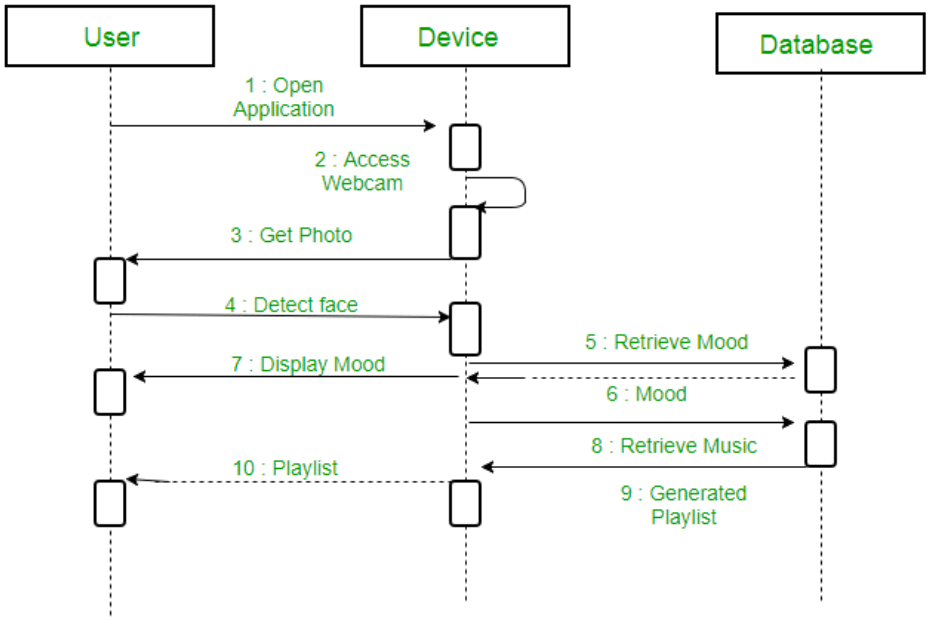


4. Guards:
   Guards are used to model conditions in UML. They are used when there is a need to restrict the flow of messages on the pretext of a condition being met. They play an important role to know the constraints attached to a system or a particular process.

   Example: Cash can be withdrawn from a bank only if the balance is greater than zero.



Sequence diagram for an emotion based music player:



References :

1. https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/
2. https://www.ibm.com/support/knowledgecenter/it/SS5JSH_9.1.0/com.ibm.xtools.sequence.doc/topics/cmsg_v.html