**Practical No. 7  
Modeling Sequence Diagrams**

**Aim:  
To model a sequence diagram for a Garage Management System (GMS) to represent the interaction between objects over time.**

Sequence diagram

It represents the behavioral aspects of a system. Sequence diagram shows the interactions between the objects by means of passing messages from one object to another with respect to time in a system.

Elements in sequence diagram

Sequence diagram contains the objects of a system and their life-line bar and the messages passing between them.

Object

Objects appear at the top portion of sequence diagram. Object is shown in a rectangle box. Name of object precedes a colon ‘:’ and the class name, from which the object is instantiated. The whole string is underlined and appears in a rectangle box. Also, we may use only class name or only instance name.

Objects which are created at the time of execution of use case and are involved in message passing , are appear in diagram, at the point of their creation.

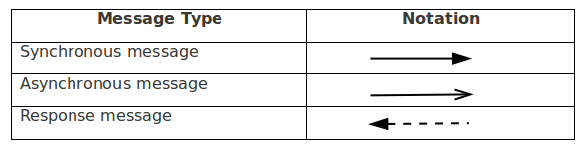
Life-line bar

A down-ward vertical line from object-box is shown as the life-line of the object. A rectangle bar on life-line indicates that it is active at that point of time.

Messages

Messages are shown as an arrow from the life-line of sender object to the life-line of receiver object and labeled with the message name. Chronological order of the messages passing throughout the objects’ life-line show the sequence in which they occur. There may exist some different types of messages :

* Synchronous messages: Receiver start processing the message after receiving it and sender needs to wait until it is made. A straight arrow with close and fill arrow-head from sender life-line bar to receiver end, represent a synchronous message.
* Asynchronous messages: For asynchronous message sender needs not to wait for the receiver to process the message. A function call that creates thread can be represented as an asynchronous message in sequence diagram. A straight arrow with open arrow-head from sender life-line bar to receiver end, represent an asynchronous message.
* Return message: For a function call when we need to return a value to the object, from which it was called, then we use return message. But, it is optional, and we are using it when we are going to model our system in much detail. A dashed arrow with open arrow-head from sender life-line bar to receiver end, represent that message.
* Response message: One object can send a message to self. We use this message when we need to show the interaction between the same object.

****

Messages

Messages are the communication between objects. They can be:

Synchronous Messages

* bookService() – Customer to Service
* assignMechanic() – Admin to Mechanic
* generateInvoice() – Service to Invoice
* makePayment() – Customer to Payment

Asynchronous Messages

* updateInventory() – Service to Inventory

**Sequence Flow Example (for Booking a Service)**

1. **Customer** sends bookService() to **Service**.
2. **Service** sends assignMechanic() to **Admin**.
3. **Admin** selects and sends assignTo() to **Mechanic**.
4. **Service** sends generateInvoice() to **Invoice**.
5. **Customer** sends makePayment() to **Payment**.
6. **Service** sends updateInventory() to **Inventory**.

