Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Academic Year: 2023_24

C12
Jonathan Dsouza - 60004220118
Sakshi Gangwani - 60004220116
Fatema Dolaria - 60004210081

Experiment No. 5

Aim: Develop Sequence, Collaboration and State diagram for the project.

Theory:

Sequence Diagram:

A sequence diagram is used to show the dynamic communications between objects during execution of a task. It shows the temporal order in which messages are sent between the objects to accomplish that task. One might use a sequence diagram to show the interactions in one use case or in one scenario of a software system.

A sequence diagram shows method calls using horizontal arrows from the caller to the callee, labeled with the method name and optionally including its parameters, their types, and the return type.

Academic Year: 2023_24

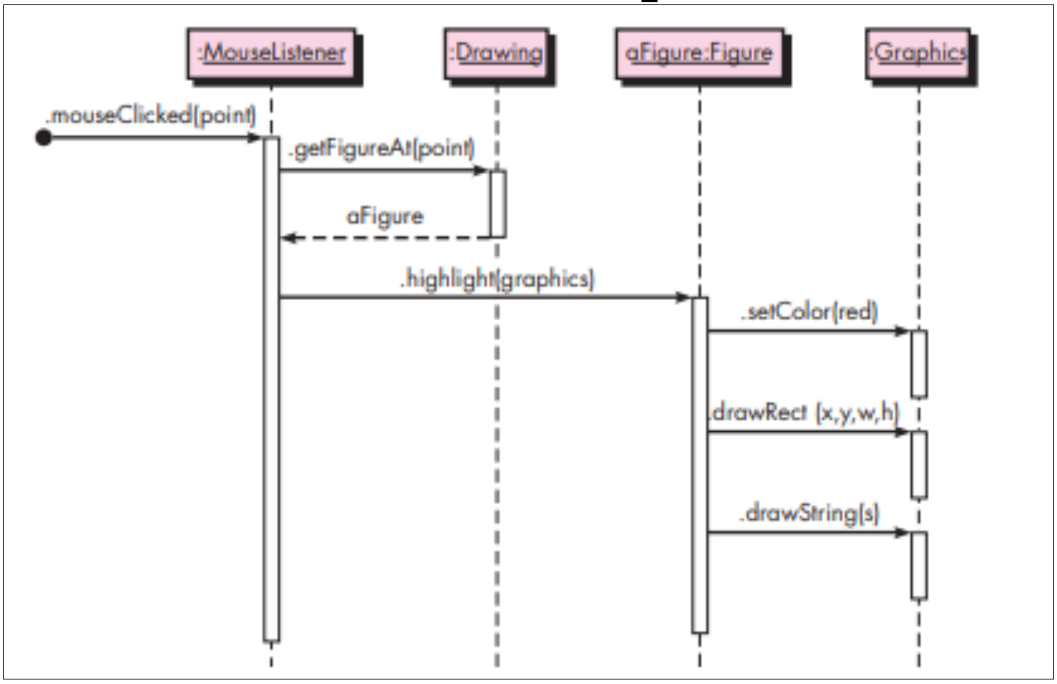


Fig 1. Sequence Diagram

The figure shows a sequence diagram for a drawing program. The diagram shows the steps involved in highlighting a figure in a drawing when it has been clicked. Each box in the row at the top of the diagram usually corresponds to an object, although it is possible to have the boxes model other things, such as classes. If the box represents an object (as is the case in all our examples), then inside the box you can optionally state the type of the object preceded by the colon. You can also precede the colon and type by a name for the object, as shown in the third box in Figure. Below each box there is a dashed line called the lifeline of the object. The vertical axis in the sequence diagram corresponds to time, with time increasing as you move downward. The diagram in Figure is very straightforward and contains no conditionals or loops. If logical control structures are required, it is probably best to draw a separate sequence diagram for each case. That is, if the message flow can take two different paths depending on a condition, then draw two separate sequence diagrams, one for each possibility

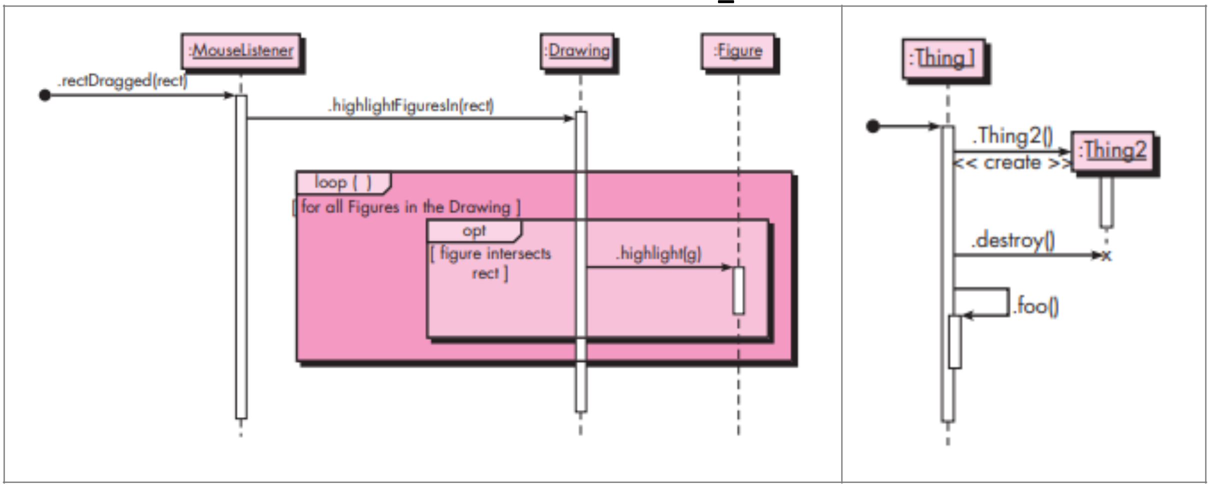


Fig 2. Sequence Diagram

Collaboration (communication) Diagrams:

A collaboration diagram, illustrated in Figure below, displays the same actions shown in the sequence diagram in Figure.

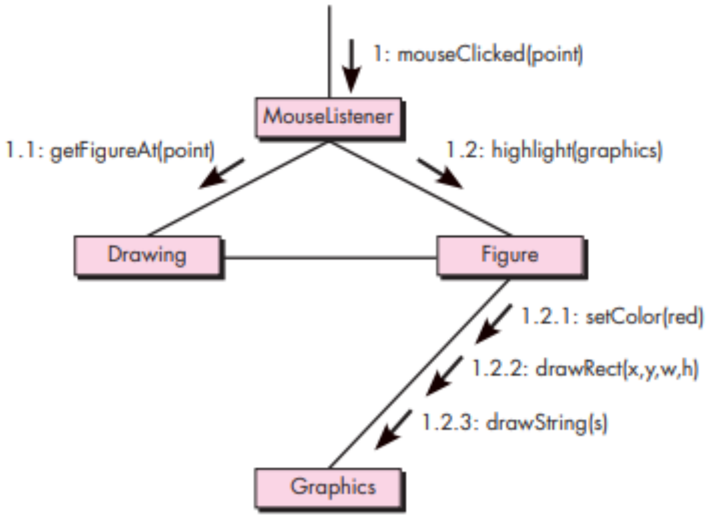


Fig 3. Collaboration Diagram

In a collaboration diagram the interacting objects are represented by rectangles. Associations between objects are represented by lines connecting the rectangles. There is typically an incoming arrow to one object in the diagram that starts the sequence of message passing. That arrow is labeled with a number and a message name. If the incoming message is labeled with the number 1 and if it causes the receiving object to invoke other messages on other objects, then those messages are represented by arrows from the sender to the receiver along an association line and are given numbers 1.1, 1.2, and so forth, in the order they are called. If those messages in turn invoke other messages, another decimal point and number are added to the number labeling these messages, to indicate further nesting of the message passing. In Figure, you see that the mouseClicked message invokes the methods getFigureAt() and then highlight(). The highlight() message invokes three other messages: setColor(), drawRect(), and drawstring(). The numbering in each label shows the nesting as well as the sequential nature of each message.

State Diagrams:

In the context of behavioral modeling, two different characterizations of states must be considered: (1) the state of each class as the system performs its function and (2) the state of the system as observed from the outside as the system performs its function. The state of a class takes on both passive and active characteristics. A passive state is simply the current status of all of an object's attributes. For example, the passive state of the class Player (in the video game application discussed in Chapter 6) would include the current position and orientation attributes of Player as well as other features of Player that are relevant to the game (e.g., an attribute that indicates magic wishes remaining). The active state of an object indicates the current status of the object as it undergoes a continuing transformation or processing. The class Player might have the following active states: moving, at rest, injured, being cured; trapped, lost, and so forth. An event (sometimes called a trigger) must occur to force an object to make a transition from one active state to another.

Each arrow shown in the following figure represents a transition from one active state of an object to another. The labels shown for each arrow represent the event that triggers the transition. Although the active state model provides useful insight into the "life history" of an object, it is possible to specify additional information to provide more depth in understanding the behavior of an object. In addition to specifying the event that causes the transition to occur, you can specify a guard and an action. A guard is a Boolean condition that must be satisfied in order for the transition to occur. For example, the guard for the transition from the "reading" state to the "comparing" state in Figure 4 can be determined by examining the use case:

if (password input = 4 digits) then compare to stored password

In general, the guard for a transition usually depends upon the value of one or more attributes of an object. In other words, the guard depends on the passive state of the object. An action occurs concurrently with the state transition or as a consequence of it and generally involves one or more operations (responsibilities) of the object. For example, the action connected to the password entered event (Figure 4) is an operation named validatePassword() that accesses a password object and performs a digit-by-digit comparison to validate the entered password.

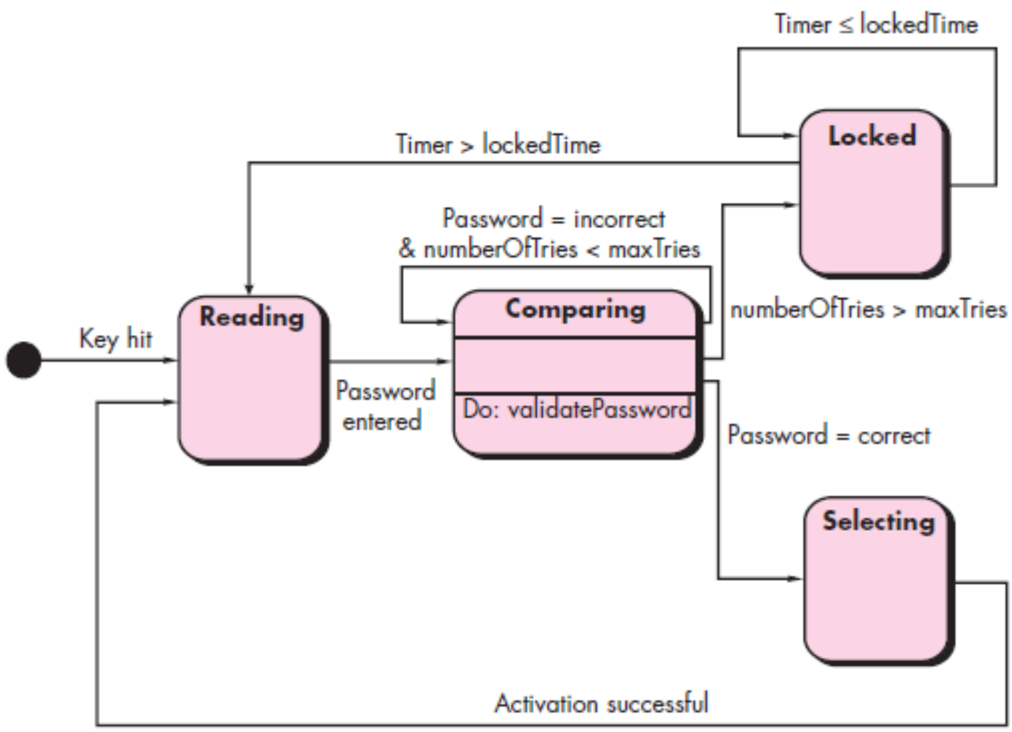


Fig 4. State Diagram

Academic Year: 2023_24

Practical:

For Sequence diagram

1. Identify any functionality involving more than two objects of your case study from Expt 3.
2. Identify the communication between the actors. Represent the same in the form of messages in the sequence diagram.

For Collaboration diagram

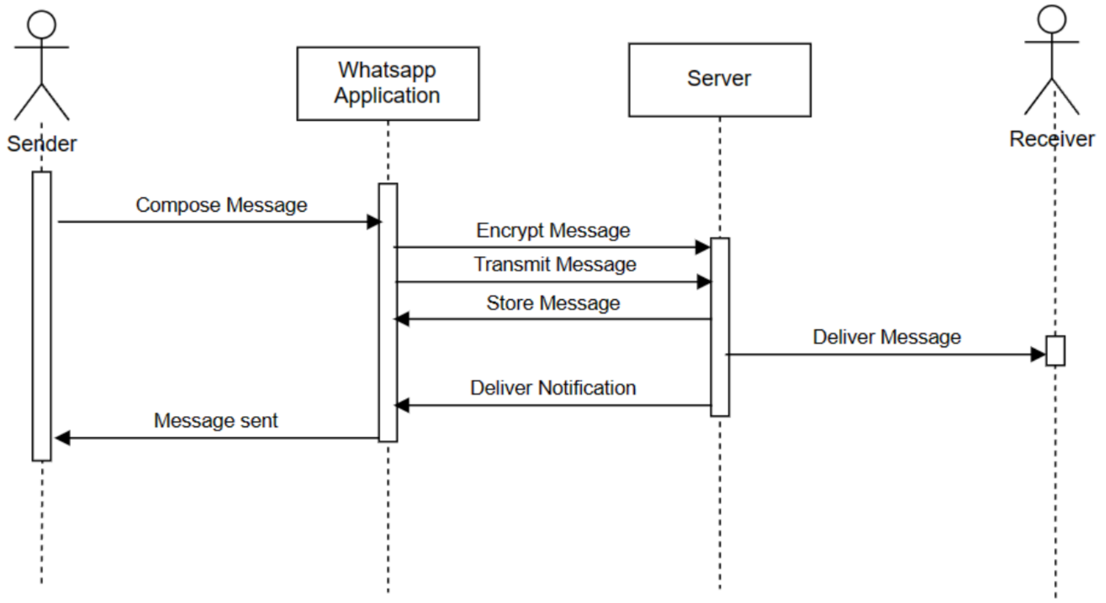
1. Convert the Sequence diagram for the same functionality into Collaboration diagram using respective notations

For State diagram

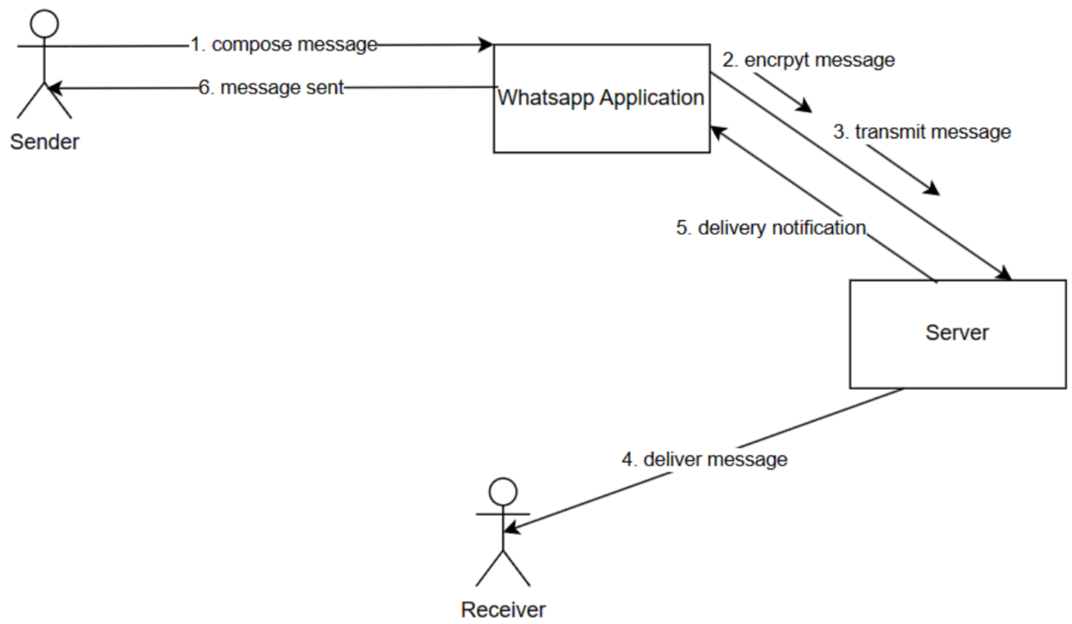
1. Identify active state for a class designed in the class diagram.
2. Identify events that can trigger the change in the state of the class.
3. Draw a detailed state diagram for this class depicting all the conditions and events for the transition

Academic Year: 2023_24

Sequence Diagram:

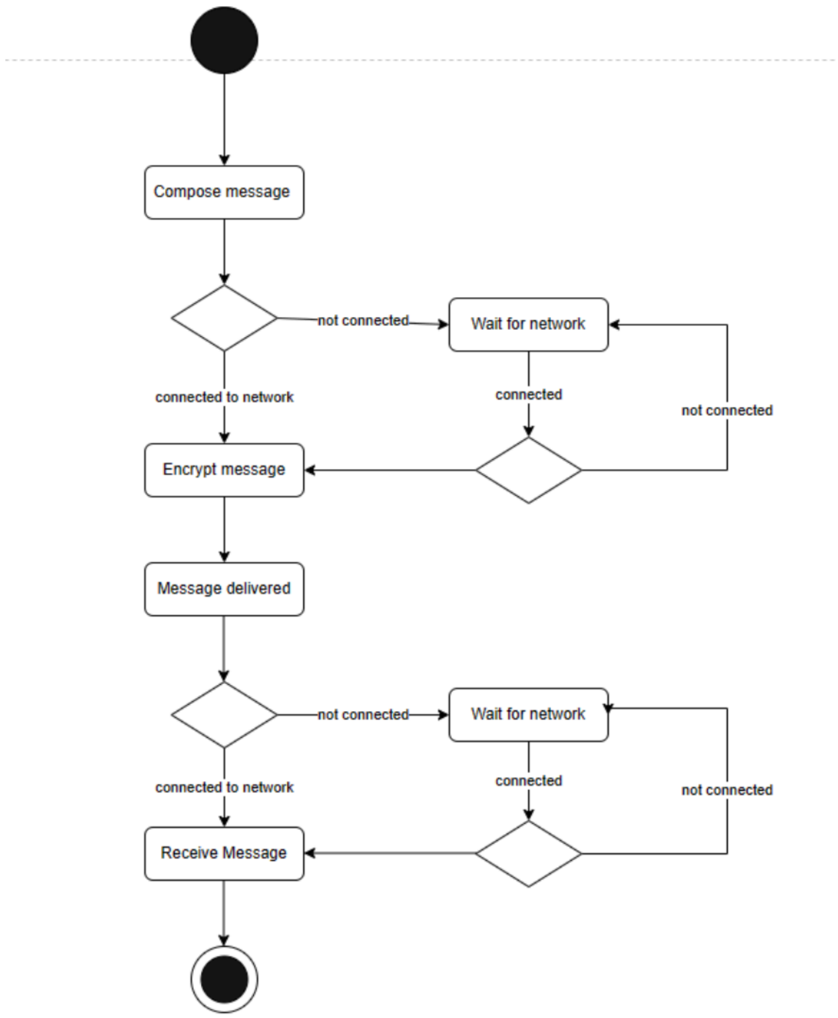


Collaboration Diagram:



Academic Year: 2023_24

State Diagram:



Conclusion: Thus, we are able to draw Sequence, Collaboration and State diagram for a functionality of our case study.