Nadar Saraswathi College of Engineering & Technology

NSCET E-LEARNING PRESENTATION

LISTEN … LEARN… LEAD…

Nadar Saraswathi College of Engineering & Technology

COMPUTER SCIENCE AND ENGINEERING

III YEAR / V SEMESTER

CS8592 OBJECT ORIENTED ANALYSIS AND DESIGN

A.DURAIMURUGAN , M.E, (Ph.d)

ASSISTANT PROFESSOR

Nadar Saraswathi College of Engineering & Technology,

Vadapudupatti, Annanji (po), Theni – 625531.

Nadar Saraswathi College of Engineering & Technology

UNIT III

DYNAMIC MODELLING

Nadar Saraswathi College of
Engineering & Technology

UML dynamic Modeling (Behavior Diagram)

- Objects are created and destroyed, objects send messages to one another in an orderly fashion, and in some system, external events trigger operations on certain objects.
- Objects have states.
- The state of an object would be difficult to capture in a static model.
- In OOD dynamic modeling can be represented by following diagrams
- Behavior Diagram
- Sequence diagrams
- Collaboration diagrams
- Statechart Diagram
- Activity Diagram

UML Interaction Diagram

- It is a diagram that describes how groups of objects collaborate to get the job done.
- Interaction diagram capture the behavior of a single use case, showing the pattern of interaction among objects.
- The diagram shows a number of example objects and the messages passed between those objects within the use case.
- There are two kinds of interaction models: sequence diagrams and collaboration diagrams.

UML Sequence Diagram

- The behavior of a system by interaction between the system and viewing the environment
- It shows the objects participating in the interaction by their life lines and the messages they exchange, arranged in a time sequence.
- A sequence diagram has two dimensions: the vertical dimension represents time, the horizontal represents different objects.

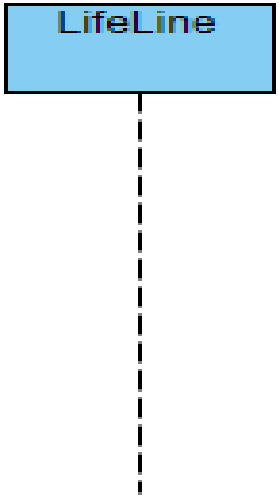
Sequence Diagram Notations

1) Object Symbol

- Represents a class or object in UML. The object symbol demonstrates how an object will behave in the context of the system. Class attributes should not be listed in this shape.

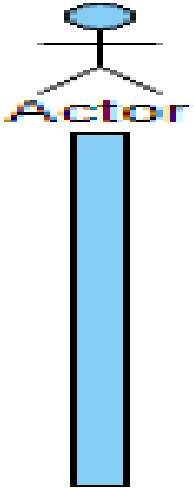
2) Lifeline

- A lifeline represents an individual participant in the Interaction.



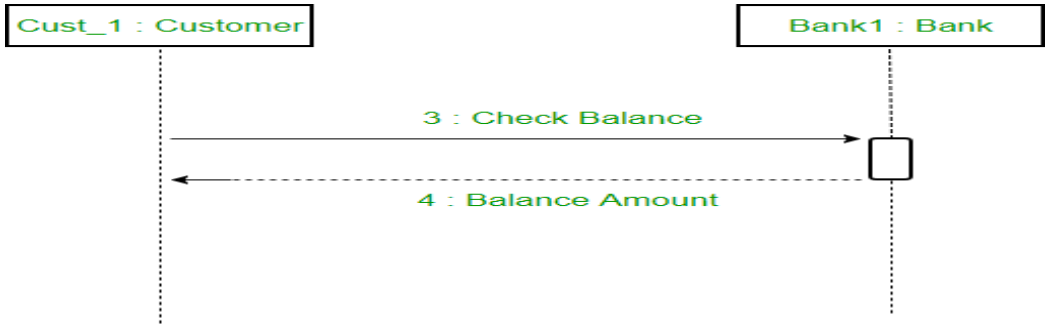
3) Actor

- Represent roles played by human users, external hardware, or other subjects.



Difference between a lifeline and an actor

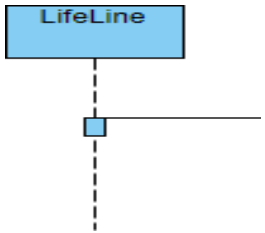
- A lifeline always portrays an object internal to the system whereas actors are used to depict objects external to the system. The following is an example of a sequence diagram:



4)Activation Box

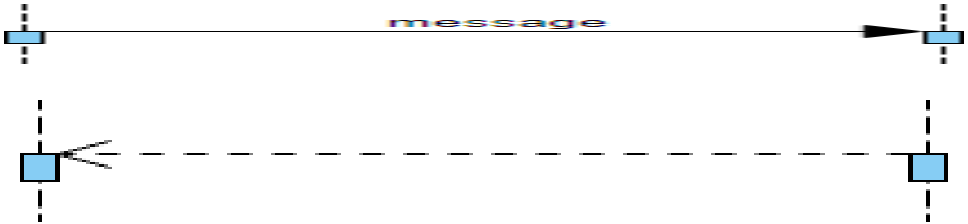
- Represents the time needed for an object to complete a task. The longer the task will take, the longer the activation box becomes.

- An activation is represented by a thin rectangle on a lifeline represents the period during which an element is performing an operation. The top and the bottom of the of the rectangle are aligned with the initiation and the completion time respectively

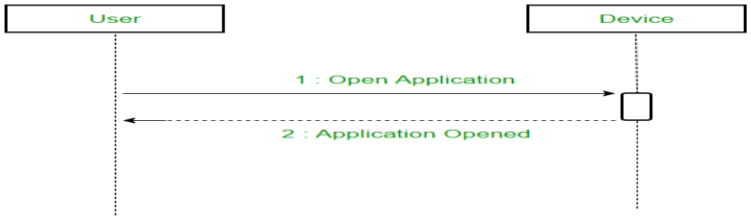


5) Messages

- Communication between objects is depicted using messages.
- The messages appear in a sequential order on the lifeline. We represent messages using arrows.
- Lifelines and messages form the core of a sequence diagram.



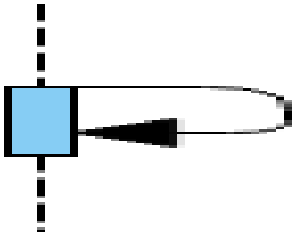
- Synchronous messages :-
- A synchronous message waits for a reply before the interaction can move forward.
- The caller continues only when it knows that the receiver has processed the previous message i.e. it receives a reply message.

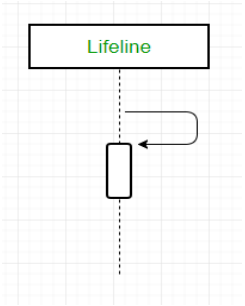


- Asynchronous Messages
- An asynchronous message does not wait for a reply from the receiver.
- The interaction moves forward irrespective of the receiver processing theprevious message or not. We use a lined arrow head to represent an asynchronous message.

Message

- Certain scenarios might arise where the object needs to send a message to itself.
- Such messages are called Self Messages and are represented with a U shaped arrow.



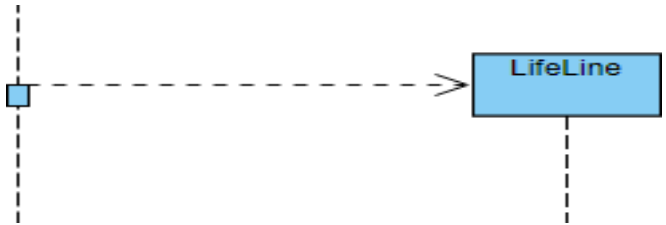


Self Message

- For example – Consider a scenario where the device wants to access its webcam. Such a scenario is represented using a self message.

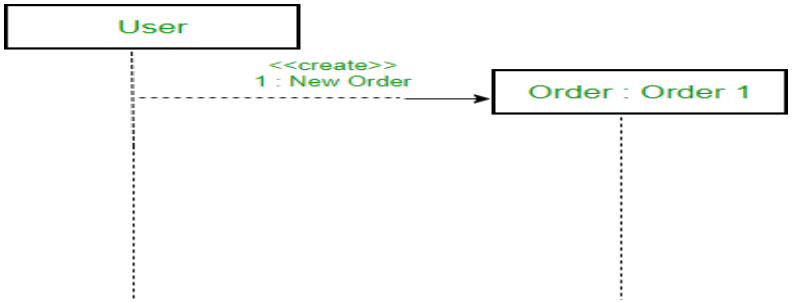
Create Message Representation

A create message defines a particular communication between lifelines of an interaction, which represents the instantiation of (target) lifeline.



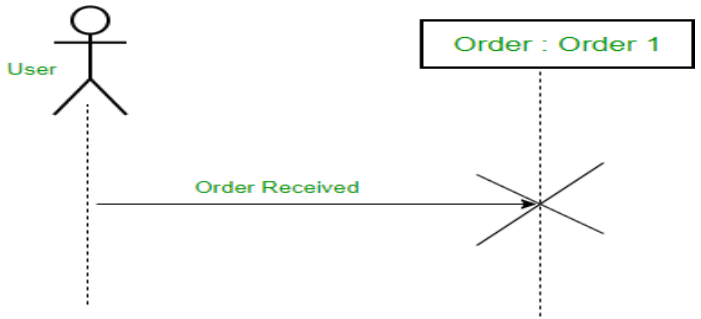
Create Message example

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

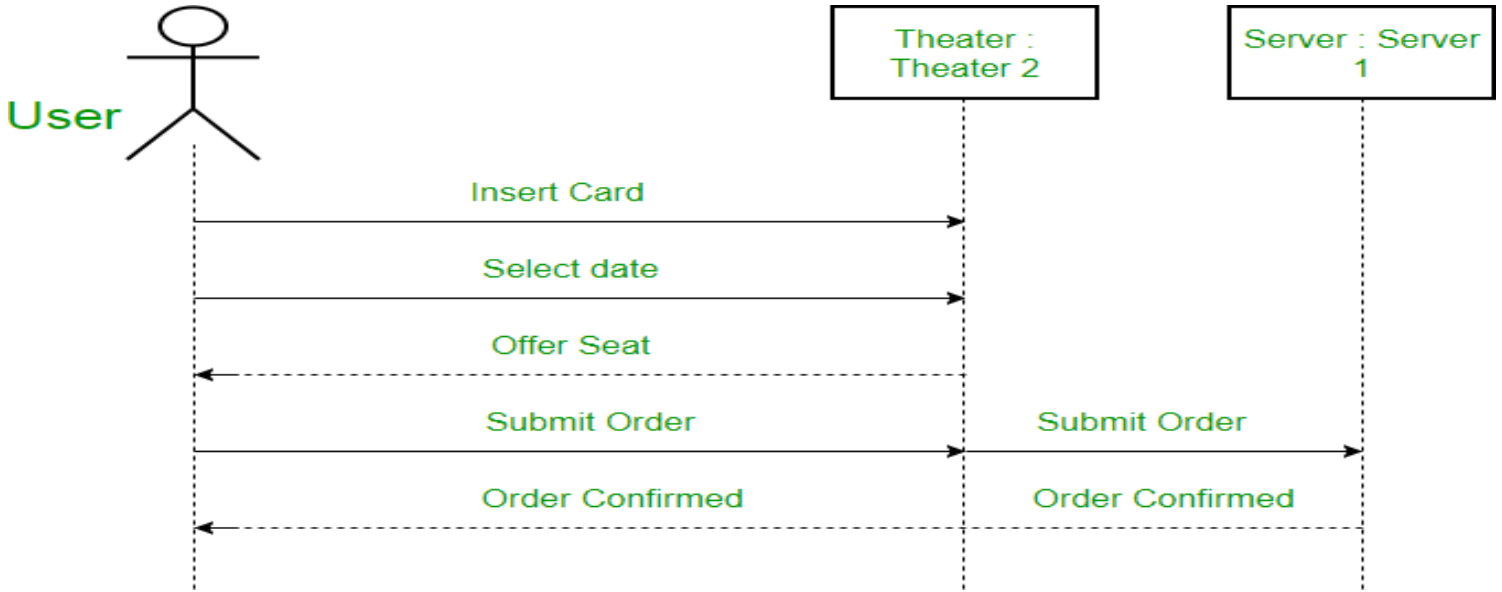


Destroy/ Delete Message example

- For example – In the scenario below when the order is received by the user, the object of order class can be destroyed.



Sequence Diagram Example-1



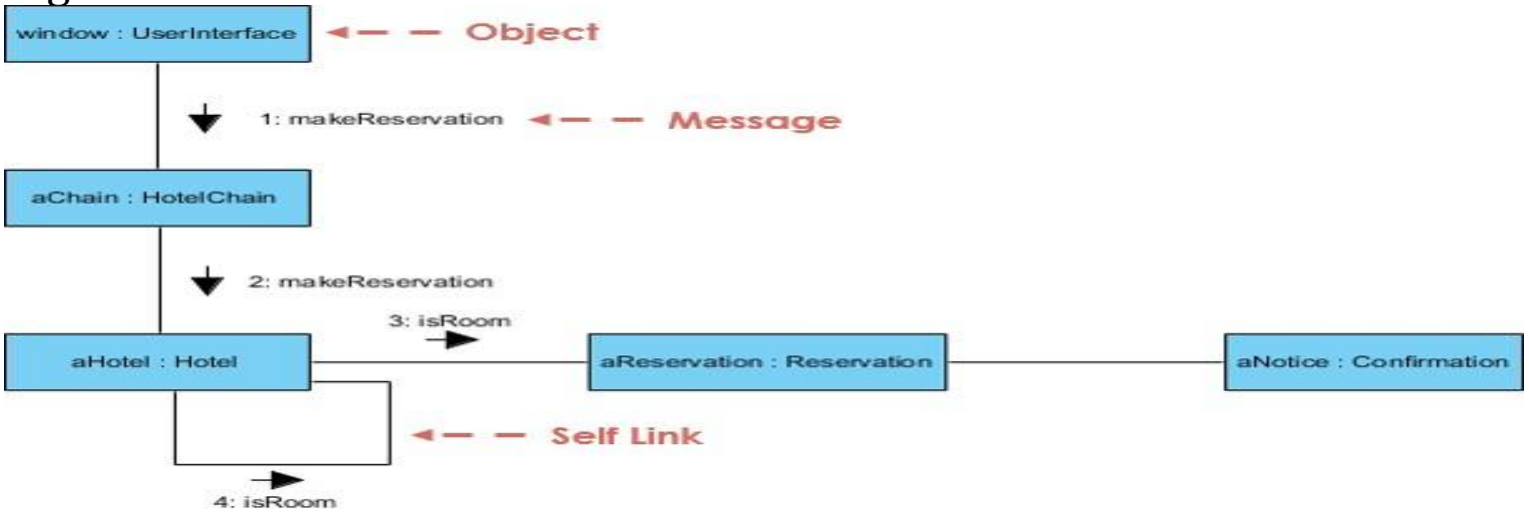
Why Collaboration Diagram?

- Unlike a sequence diagram, a collaboration diagram shows the relationships among the objects.
- Sequence diagrams and collaboration diagrams express similar information, but show it in different ways.
- The sequence diagram is used when time sequence is main focus.
- A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.
- In a collaboration the sequence is indicated by numbering the messages.

- Identify behavior whose realization and implementation is specified
- Identify the structural elements (class roles, objects, subsystems) necessary to carry out the functionality of the collaboration
  - Decide on the context of interaction: system, subsystem, use case and operation
- Model structural relationships between those elements to produce a diagram showing the context of the interaction
- Consider the alternative scenarios that may be required
  - Draw instance level collaboration diagrams, if required.
  - Optionally draw a specification level collaboration diagram to summarize the alternative scenarios in the instance level sequence diagrams

Collaboration Diagram?

The diagram



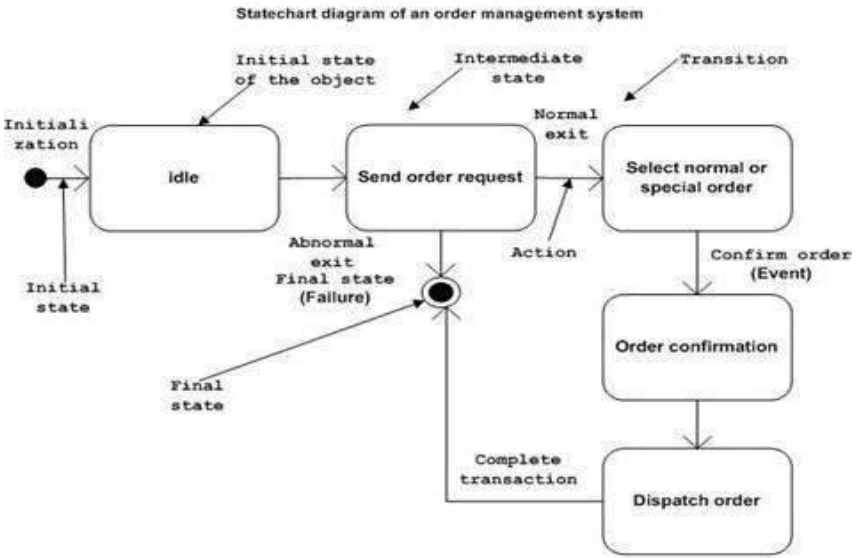
UML Statechart Diagram

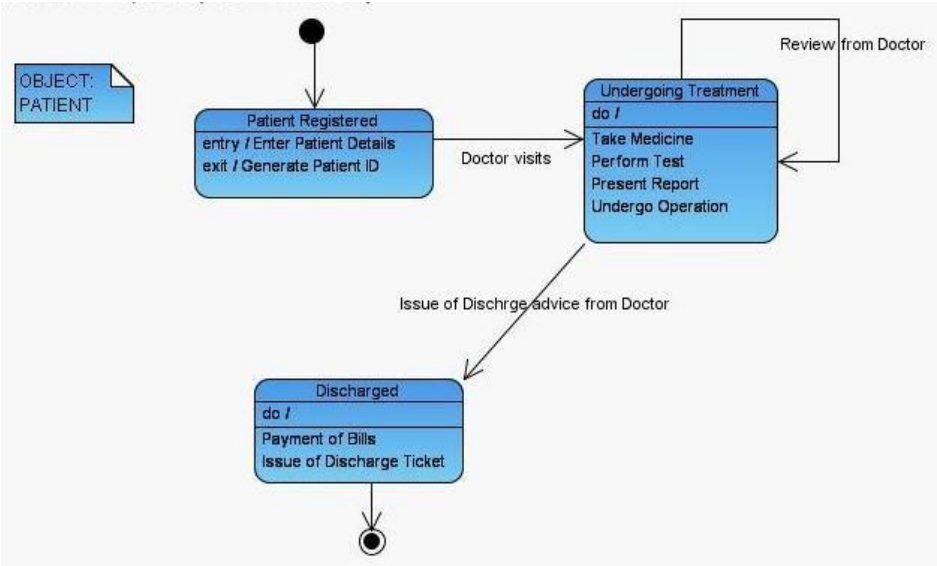
- A state diagram describes the behavior of a single object in response to a series of events in a system.

Following are the main purposes of using Statechart diagrams:

- To model dynamic aspect of a system.
- To model life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model states of an object.

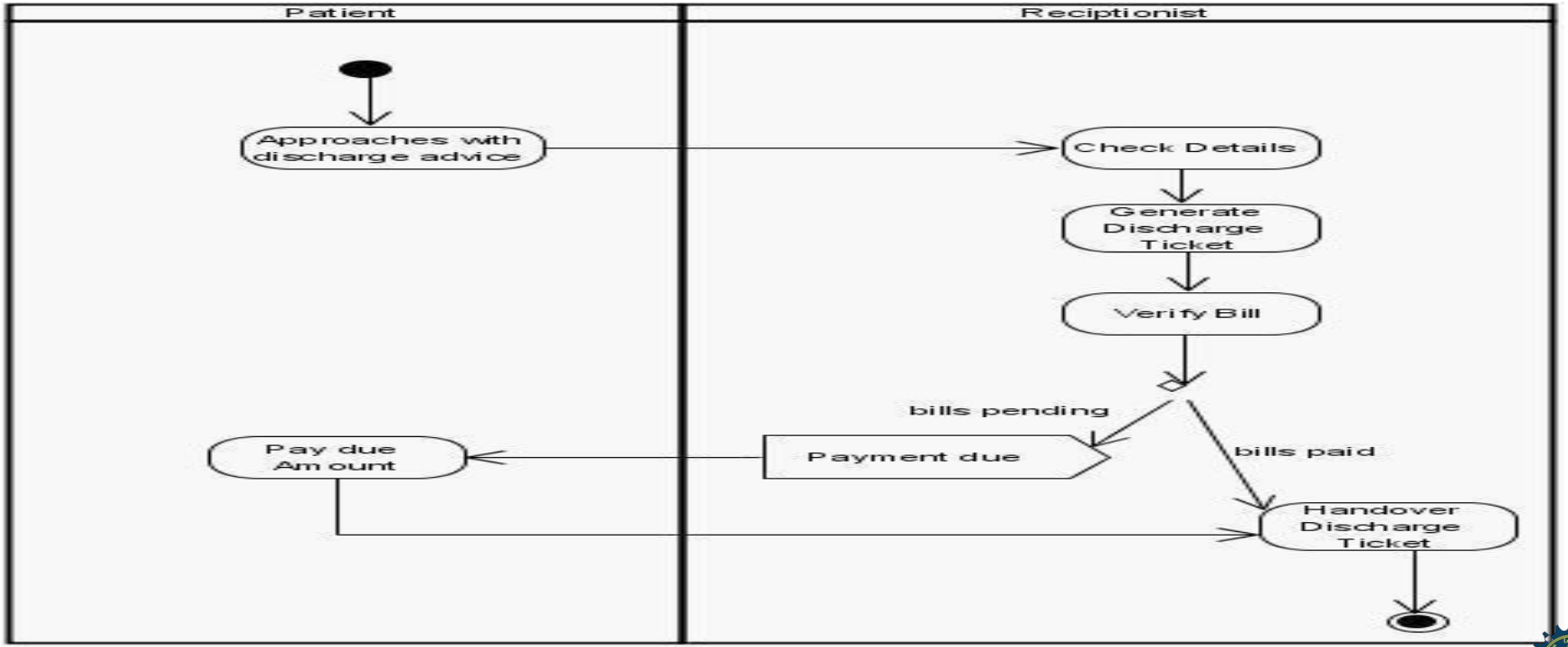
- State Chart Diagram for Patient





UML Activity Diagram

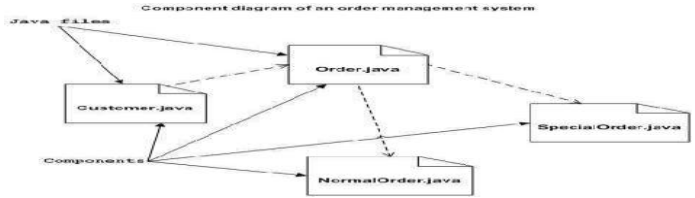
- An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram.
- Activity diagrams are often used in business process modeling.
- They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.
- In both cases an activity diagram will have a beginning (an initial state) and an end (a final state).
- It captures the dynamic behaviour of the system.
- activity diagram is used to show message flow from one activity to another



Implementation Diagram

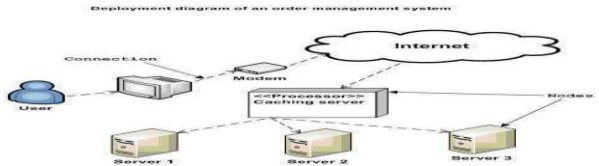
- It shows the implementation phase of systems development, such as the source code structure and the run-time implementation structure.
- There are two types of implementation diagrams: component diagrams show the structure of the code itself, and deployment diagrams show the structure of the runtime system
- a) Component diagram
- Component diagrams are used to model the physical aspects of a system.
- Component diagrams are used to visualize the organization and relationships among components in a system.

- Model the components of a system.
- Model the database schema.
- Model the executables of an application.
- Model the system's source code.



Deployment Diagram

- Deployment diagram represents or describe the static deployment view of a system.
- It specifies the physical hardware on which the software system will execute.
- It also determines how the software is deployed on the underlying hardware.
- It maps software pieces of a system to the device that are going to execute it.



Package Diagram

- A package is a grouping of model elements.
- Packages themselves may contain other packages.
- Package may contain both subordinate packages model elements.
- A package is represented as a folder, shown as a large rectangle with a tab attached to its upper left corner.
- If contents of the package are shown, then the name of the package may be placed on the tab.
- The contents of the package are shown within large rectangle.

Package Diagram