**Experiment No.05**

PART A

(PART A: TO BE REFFERED BY STUDENTS)

**Experiment 5**

**A.1 Aim:** Design State chart diagram using case study’s Control specifications. Make use of Star UML software for design

**A.2 Prerequisite:**

Process Model, DFD, Requirement Engineering

**A.3 Outcome:**

After completing this experiment you will be able to:

* Identify the distinct states a system have
* Represent the above information pictorially using simple states

**A.4 Theory:**

**State Chart Diagram:**

Capturing the dynamic view of a system is very important for a developer to develop the logic for a system. State chart diagrams are popular UML diagram to visualize the dynamic behavior of an information system.

In this experiment, we will learn about the different components state chart diagram and how these can be used to represent the dynamic nature of an information system.

In case of Object Oriented Analysis and Design, a system is often abstracted by one or more classes with some well defined behaviour and states. A statechart diagram is a pictorial representation of such a system, with all it's states, and different events that lead transition from one state to another.

To illustrate this, consider a computer. Some possible states that it could have are: running, shutdown, hibernate. A transition from running state to shutdown state occur when user presses the "Power off" switch, or clicks on the "Shut down" button as displayed by the OS. Here, clicking on the shutdown button, or pressing the power off switch act as external events causing the transition.

Statechart diagrams are normally drawn to model the behaviour of a complex system. For simple systems this is optional.

A state is any "distinct" stage that an object (system) passes through in it's lifetime. An object remains in a given state for finite time until "something" happens, which makes it to move to another state.  All such states can be broadly categorized into following three types:

* **Initial**: The state in which an object remain when created
* **Final**: The state from which an object do not move to any other state [optional]
* **Intermediate**: Any state, which is neither initial, nor final

As shown in figure, an initial state is represented by a circle filled with black. An intermediate state is depicted by a rectangle with rounded corners.

State

A final state is represented by a unfilled circle with an inner

black-filled circle. Figure Representation of initial, intermediate, and final states of a statechart diagram Intermediate states usually have two compartments, separated by a horizontal line, called the name compartment and internal transitions compartment. They are described below:

* **Name compartment**: Contains the name of the state, which is a short, simple, descriptive string
* **Internal transitions compartment**: Contains a list of internal activities performed as long as the system is in this state

The internal activities are indicated using the following syntax: action-label / action-expression. Action labels could be any condition indicator. There are, however, four special action labels:

* **Entry**: Indicates activity performed when the system enter this state
* **Exit**: Indicates activity performed when the system exits this state
* **Do**: indicate any activity that is performed while the system remain in this state or until the action expression results in a completed computation
* **Include**: Indicates invocation of a sub-machine

Any other action label identify the event (internal transition) as a result of which the corresponding action is triggered. Internal transition is almost similar to self transition, except that the former doesn't result in execution of entry and exit actions. That is, system doesn't exit or re-enter that state. Figure shows the syntax for representing a typical intermediate state.

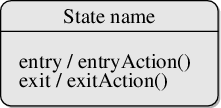


Figure: A typical state in a statechart diagram States could again be either simple or composite (a state congaing other states). Here, however, we will deal only with simple states.

**Transition:**

Transition is movement from one state to another state in response to an external stimulus (or any internal event). A transition is represented by a solid arrow from the current state to the next state. It is labeled by: event [guard-condition]/[action-expression], where

* **Event** is the what is causing the concerned transition (mandatory) -- Written in past tense
* **Guard-condition** is (are) precondition(s), which must be true for the transition to happen [optional]
* **Action-expression** indicate action(s) to be performed as a result of the transition [optional]

It may be noted that if a transition is triggered with one or more guard-condition(s), which evaluate to false, the system will continue to stay in the present state. Also, not all transitions do result in a state change. For example, if a queue is full, any further attempt to append will fail until the delete method is invoked at least once. Thus, state of the queue doesn't change in this duration.

**Action**

As mentioned in, actions represents behaviour of the system. While the system is performing any action for the current event, it doesn't accept or process any new event. The order in which different actions are executed, is given below:

1. Exit actions of the present state
2. Actions specified for the transition
3. Entry actions of the next state

Figure-03 shows a typical statechart diagram with all it's syntaxes.

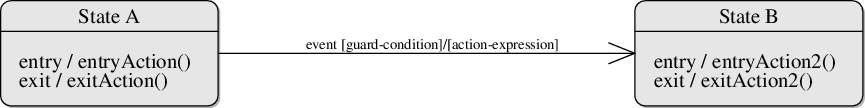


Figure: A statechart diagram showing transition from state A to B

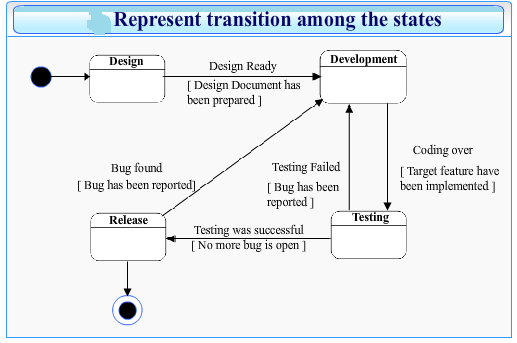
**Guidelines for drawing State chart Diagrams**

Following steps could be followed, as suggested to draw a statechart diagram:

* For the system to developed, identify the distinct states that it passes through
* Identify the events (and any precondition) that cause the state transitions. Often these would be the methods of a class as identified in a class diagram.
* Identify what activities are performed while the system remains in a given state
* List all sensors that are "read" by the software.
* List all interrupt conditions.
* List all "switches" that are actuated by the operator.
* List all data conditions.
* Recalling the noun-verb parse that was applied to the software statement of scope, review all "control items" as possible CSPEC inputs/outputs.
* Describe the behavior of a system by identifying its states; identify how each state is reach and defines the transitions between states.

Drawing statechart diagram from a problem statement

Consider the development of a medium-scale software in, say, C++. In a software development life cycle typically four phased could be observed. It begins with planning and design, after which the development work starts. Once the required features have been implemented, the software is tested. If testing is successful, the software is released. If there is even a single failure, that is to be fixed, and tested again. However, certain bugs might become apparent only after someone uses the software in real scenario. As such, the software again enters into development phase, followed by testing, and then another release.

****

**Task to be completed:**

1. List all the states existing in your case study
2. Identify the transitions and actions
3. Design the state chart diagram for your project

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**PART B**

(PART B: TO BE COMPLETED BY STUDENTS)

**(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)**

|  |  |
| --- | --- |
| Roll No. B228 | Name: Pranav Kolhe |
| Program: Btech | Division: CS |
| Batch: A | Date of Experiment: |
| Date of Submission: | Grade : |

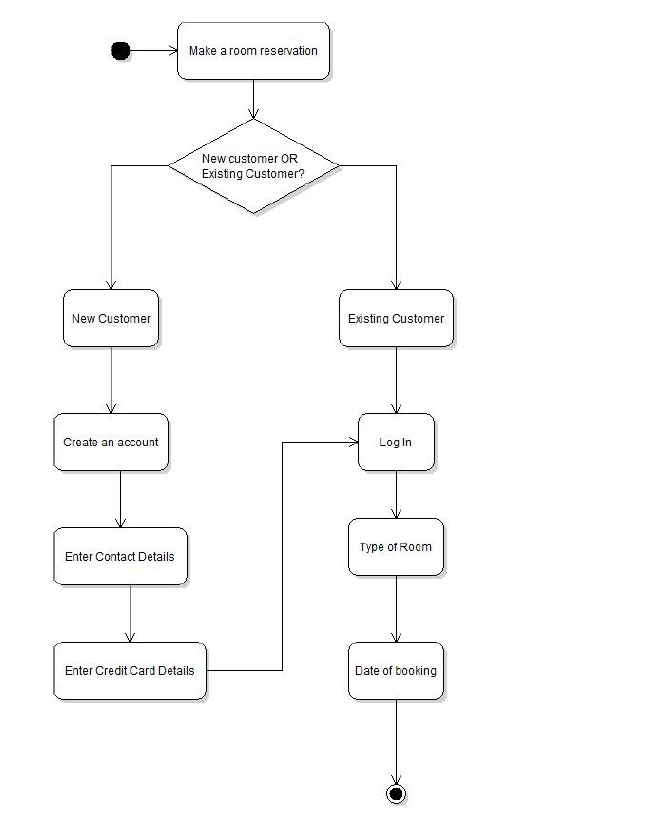
**B.1 Tasks given in PART A to be completed here**

*(****Students must write the answers of the task(s) given in the PART A)***

**B.2 Observations and Learning:**

*(****Students must write the observations and learning based on their understanding built about the subject matter and inferences drawn)***

State Chart Diagram was created for the project.



**B.3 Conclusion:**

*(****Students must write the conclusive statements as per the attainment of individual outcomes listed above and learning/observation noted in section B.2)***

The experiment was executed successfully.

**B.4 Question of curiosity:**

**1. Why it is necessary to design state chart diagram in software engineering?**

The most important purpose of State chart diagram is to model lifetime of an object from creation to termination. State chart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system. To model the dynamic aspect of a system.

**2. Differentiate between state chart and data flow diagram.**

State diagrams are used to model the behavior of one object over several use cases in the system. A state describes the condition of an object at a given point in its lifetime. A state diagram describes all possible states of an object, and the paths to get from one state to another when an external event is received.

A Data Flow Diagram (DFD) is a graphical representation of flow of information or data throughout the system.

**3. What does an entry action of a state indicate?**

**Action performed after the system moves into the given state http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**Action performed before system moves into the given state http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**An optional action performed when system moves into the given state http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**None of the above http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**4. What does the guard condition depicted over the transition between any two states indicate?**

**A condition that must be true for the transition to happen http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**A condition that must be false for the transition to occur http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**An indicator that this transition should not happen http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**An event that might happen as result of the transition http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**5. A state can contain one or more sub-state(s) within it**

**True http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

**False http://vlabs.iitkgp.ernet.in/se/isad_static/isad/images/transparent_2x2.png**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*