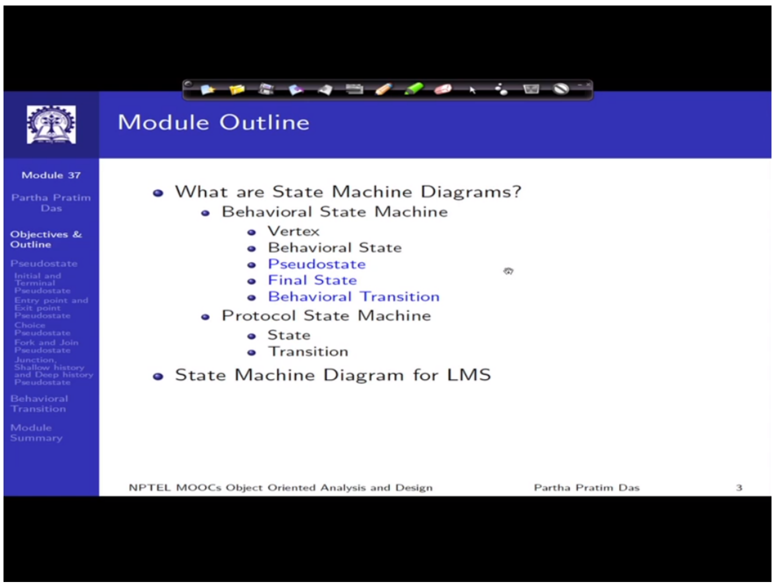
Object-Oriented Analysis and Design
Prof. Partha Pratim Das
Department of Computer Science and Engineering
Indian Institute of Technology-Kharagpur

Lecture – 49
State Machine Diagrams: Part II

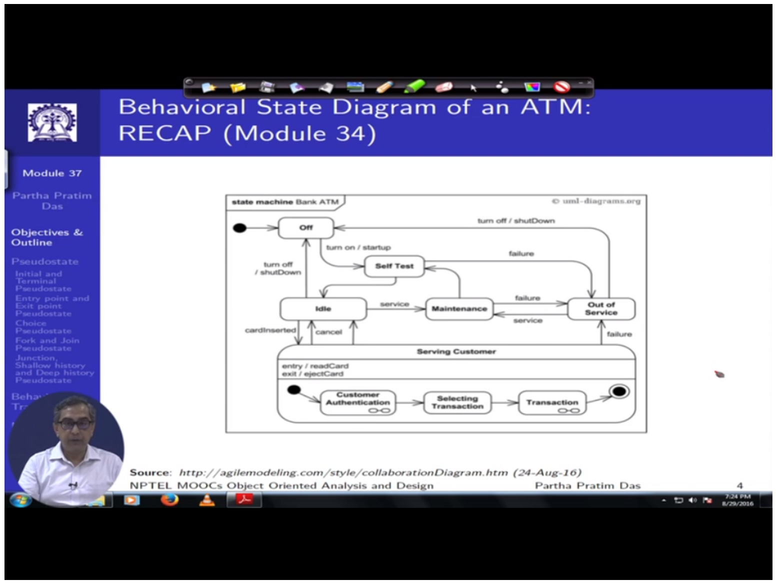
Welcome to module 37 of object-oriented analysis and design. From the last module we have been discussing about state machine diagrams. These are behavioral diagrams used to capture the discrete behavior of a complex system at an appropriate level.

(Refer Slide Time: 00:49)



And we have in this outline, we have already covered the vertex and behavioral state of a state machine diagram and where in we have seen that a state machine diagram could be of behavioral state nature or it could be of a protocol state machine nature. Protocol state machine we have not discussed as yet. In terms of the behavioral state machine we have seen that there could be different behavioral states: simple, composite which comprise lot of sub states in one region or in multiple regions, orthogonally where which happen concurrently or it could have sub machine states.

(Refer Slide Time: 01:44)

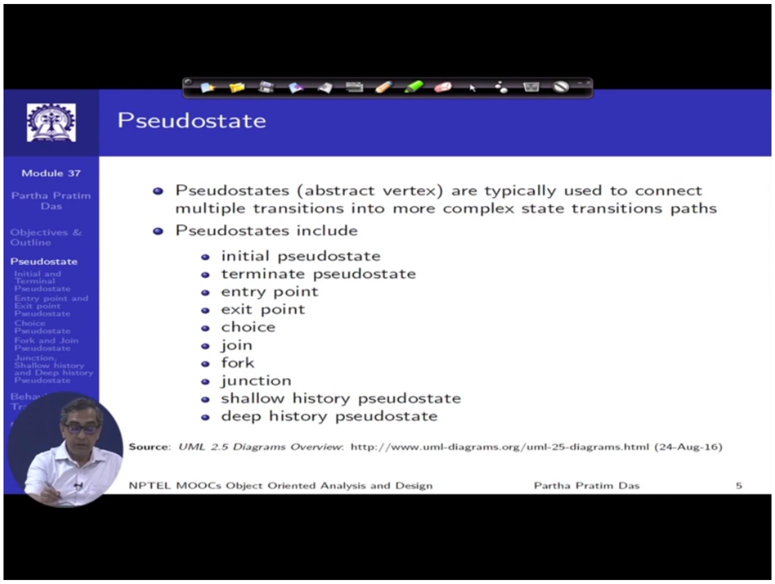


In this context we will next introduce the concept of pseudo states and talk about the behavioral transitions in this model. So umm this is the state machine diagram of an ATM that we closed the earlier module which and I expect that you have already tried to understand and work through this. So in this, what we can see that this is the start and this is the final end termination of the state machine diagram.

So this is when it is in a state off and you can turn it you can turn it on and get into a that gets into a self test mode and then if it is if its tested ok then it gets into the idle state where it will remain there till somebody inserts the card, where it gets into this serving customer state which is as you can see it is a composite state but it is a non orthogonal composite state where simple composite state because it has only one region and the operations here in are sequential in nature and here in you have other composite states.

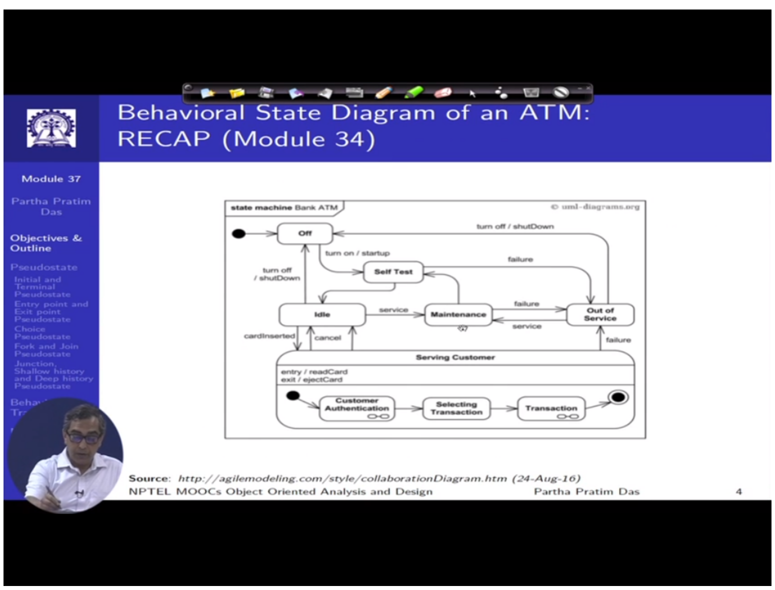
These symbols show you there are other composite state; you have further detailing of the customer authentication and the transaction kind of state behavior that is being talked of. This composite state has entry and exit as readCard and ejectCard, is on entry the action is readCard, on exit the action is ejectCard and the overall action of this activity of this composite state is what is shown through this. And there are several exits if you cancel this comes back to idle, if it fails it goes to out of service, there would be different transitions between maintenance, out of service, self test and so on. So this mode comprehends a bit class to define the state transitions of a of a net game system at a behavioral level.

(Refer Slide Time: 03:49)



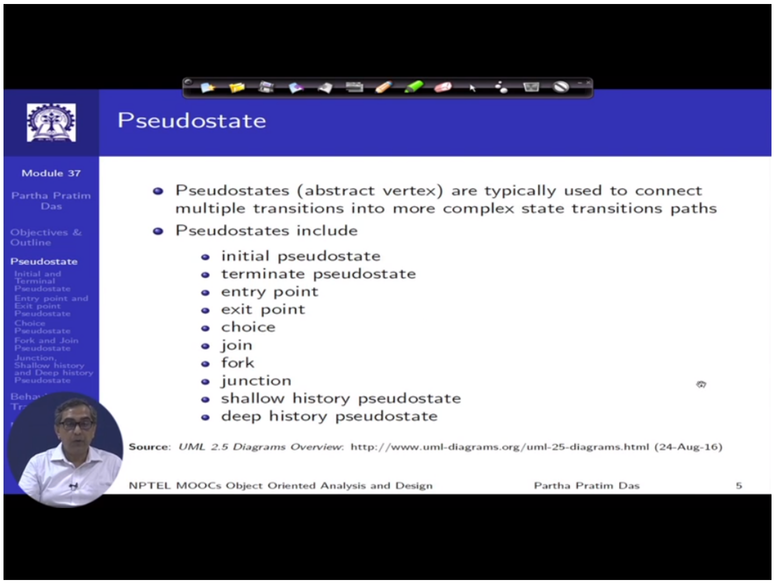
So moving on we are talking about pseudostates or abstract vertex. The idea is very simple; we are talking about state transition or state machine diagram. So these are these are typically how the states are drawn and you have transitions between them. The transition may have some event, some action with the transition to happen. So these we have seen as called the different types of behavioral type and so on. But if you look at the whole diagram as a graph then you will find that just by using this state symbols will not be able to define the whole diagram.

(Refer Slide Time: 04:42)



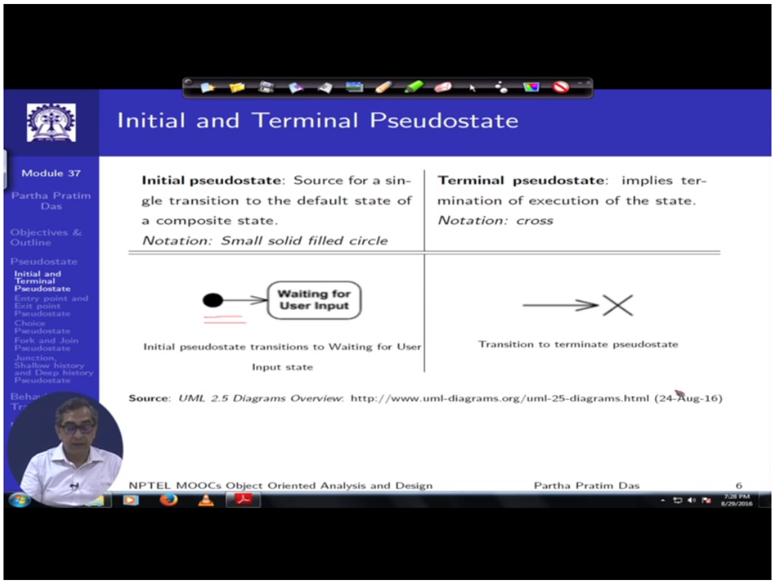
So for example if you just go back and take a look into this state machine diagram, these are these are all different behavioural states. But with their transition alone we cannot define the whole system because we need for example disc mode as well because whatever be your beginning there have to be something where there is no state then your whole system will start. There will have to be something where beyond which there is no further state and so on. So these kinds of states which need to be there as vertices in the state machine diagram but do not represent the behavioral state are typically called the psuedostates.

(Refer Slide Time: 05:30)



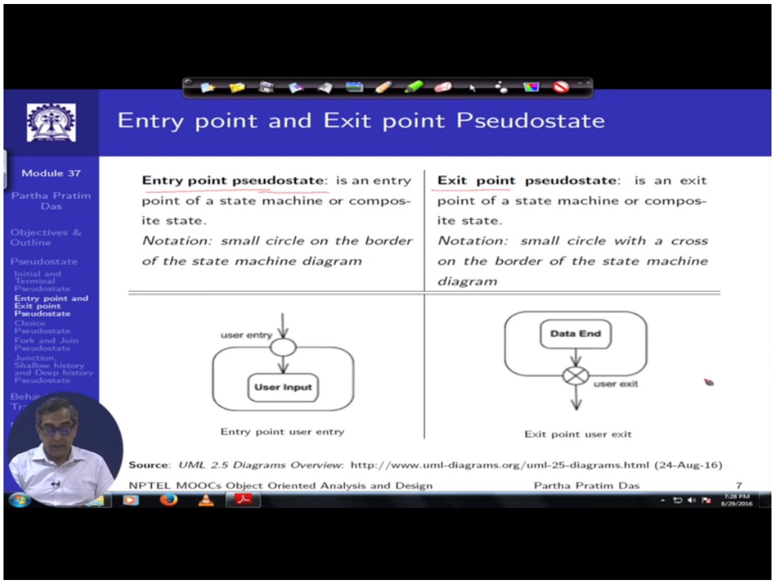
And we will say that psuedostates also carry a lot of information and there are as you can see the list, there are whole lots of pseudostates which not only define certain resting place for the transitions but they also define certain different ways that the transitions can actually get composed.

(Refer Slide Time: 05:48)



So initial pseudostate is what you have you have seen this is the where you get started and there this simply shown has a blog and a terminal pseudostate is beyond which you cannot proceed further. So implies the termination of the execution of these states and this is not seen as a final state of the state machine that is the different thing. This is just termination of the execution so which means that in terms of the transition there is no further transition possibly, the state machine is still continued executes and so on. So these are the first simplest kind of pseudostate.

(Refer Slide Time: 06:30)

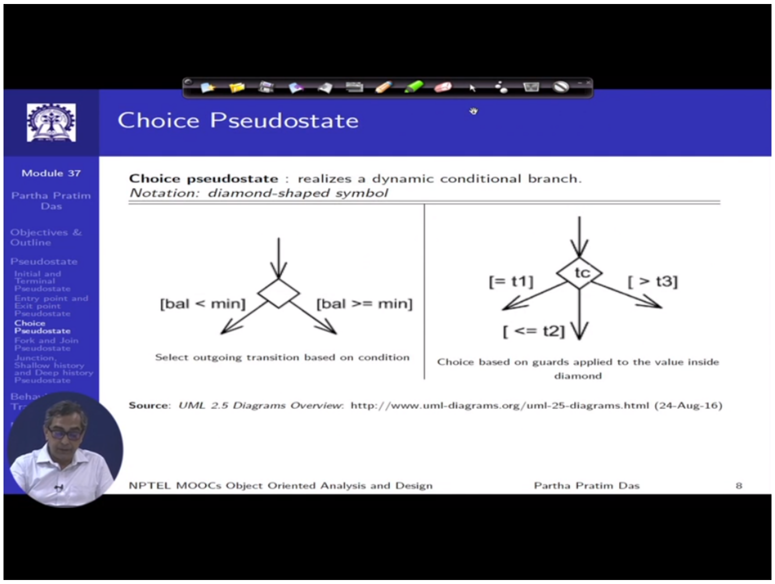


Then you have entry point and exit point pseudostates. So the entry point and the exit point pseudostates are this is a state and you say that this is a user entry, you talk about entry and exit of states. So this is the user entry that is happening here. So with this symbol you show that at this point for the state to actually make transition each to take some input, some entry from the user. Similarly there are these are this is another state and this symbol show that the user exit.

And this cross shows is an exit point of a state machine of the composite state and this is the point through which it will go out. So these are basically entry point and exit point pseudostates are basically more meaningful if you have submission state or a composite state where you show by this entry point where your entry composite state and or the submission state or where you leave go out of the submission state or composite state. Mind you these are not same as the initial.

And final state but the state machine which are basically true for the whole machine here you are talking about entry and exit from a composite state which is a part of this state.

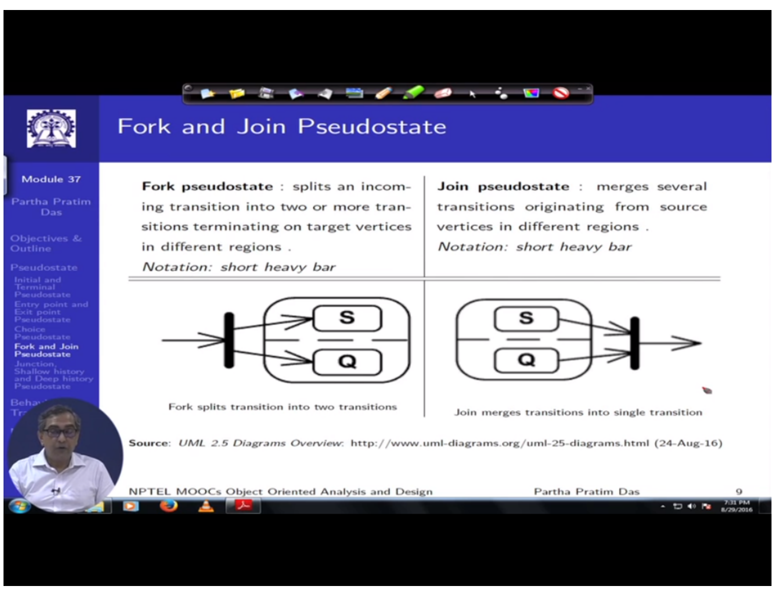
(Refer Slide Time: 07:53)



There are choice pseudostate which are basically like decision parts, so in terms of a transition could happen that based on certain condition which are the guard condition and this will look very similar to the kind of decision mode and kind of merge modes and all that leads to the activity diagram but there they mean some I mean they are semantically similar but they mean different things.

Here you are saying that these are pseudostate so that if a transition is suppose to happen on this part depending on this condition will either happen on this part or happen on this part. So that is the choice that you are making. The choice need not be binary; this could be multiple choices as well. And such vertices that happen in the state machine graph is known as choice pseudostates.

(Refer Slide Time: 08:57)

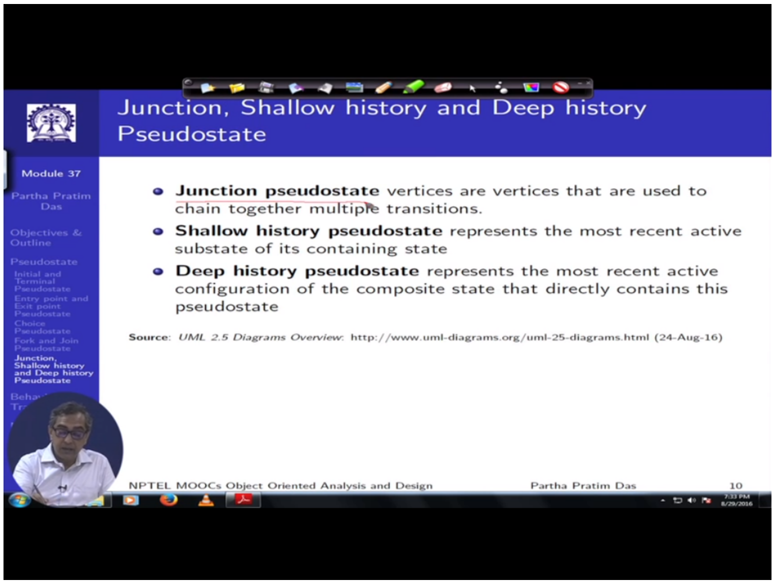


Then as we know that there could be concurrent access, there could be concurrent execution. So like the fork node and the join node we had in the activity diagram, we have fork and join pseudostates in the state machine diagram. So this fork in so this what this say that if this is a fork pseudostate so if splits a transition incoming transition into 2 or more transitions terminating at a target vertices in a different regions.

So what did we specify? we said that if we have composite state with more than one region say as we see here this is the this whole thing is a composite state, this is one region, this is another region shown by separated by this dotted separator and this is the whole composite state, these are the sub states, S and Q are the sub states of the composite state and they are into 2 different regions. So if we want to show that we in transitions happen into both of these regions together.

So it is like the transition is happening on this and happens on the sss sub state you have region 1 and sub state of region 2 where they are concurrent, then you say you have a forking transition or you have a fork pseudostate. Similarly if you have multiple regions we need to merge the different regions the transitions, common transitions. So this is region 1, this is region 2, they merge together, this is again we can merge node of a diagram. So they are merged together in the transitions into single transition, so beyond this point it becomes a single transition.

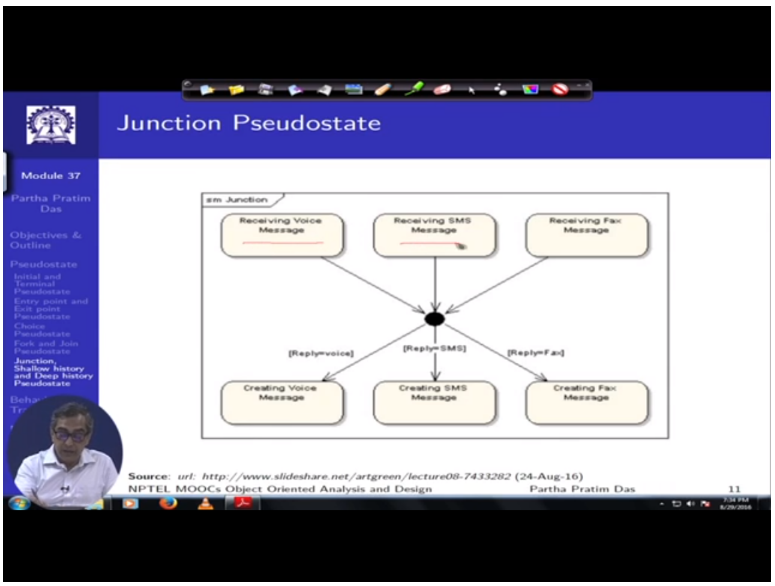
So these are also nodes in states in a state machine diagram but these are pseudostates required for expressing the concurrency in terms of the transition.
(Refer Slide Time: 11:05)



We have a couple of other pseudostates as well, one is called the junction pseudostate. The junction pseudostate is where you create a pseudostate, you do a confuluence of multiple different transitions at one point and so they merge and they diverge together and so on. And you have shallow history pseduostate and deep history pseudostate is basically give some composite states put on hold in terms of its execution.

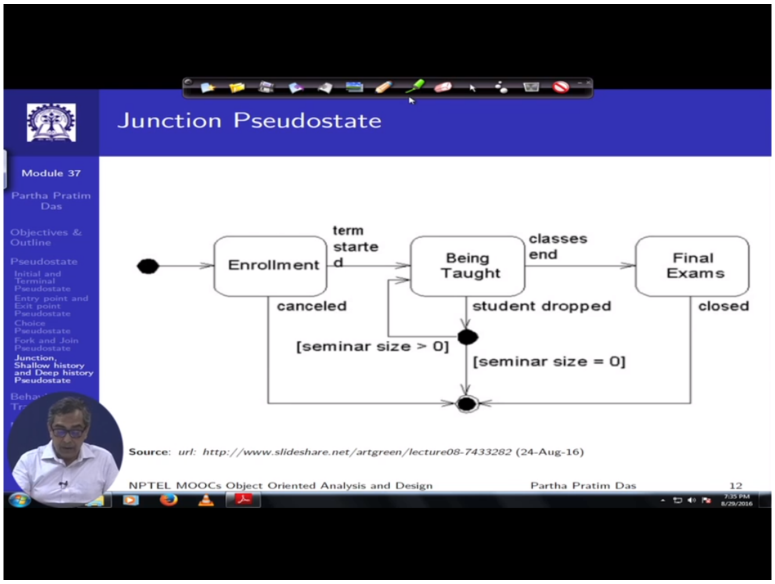
And then that region is resumed at a later point time to consume how, what whether it will resume from the most recent activity or between if a retrieve will resume from the most recent configuration depending on that you have 2 different kinds of this pseudostates and those are also part of state machine diagram.

(Refer Slide Time: 12:07)



So here they have a join junction pseudostate where this is receiving a voice message, receiving sms message, this is receiving a text message. So on this junction the transition is on junction pseudostate up to which you basically get into different transitions of depending on they are replying in voice, they replying on sms or replying as fax. So whenever you have such confluence message you say you have a junction pseudostate.

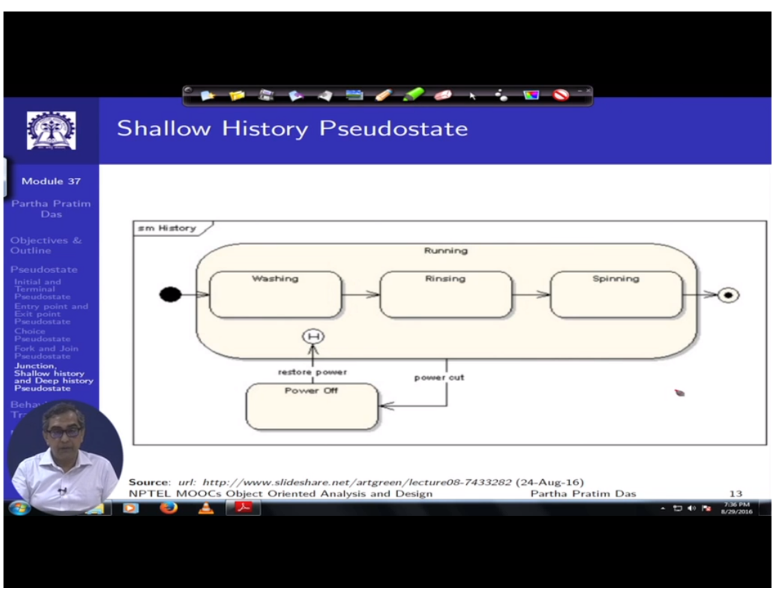
(Refer Slide Time: 12:45)



So this is another example. So here we are showing process of enrolment into a semester. So this is start, initial node and this is the sub state node that enrollment here, the term started. So then you make transition being taught and if the enrollment is cancelled then here and here this this process end you go for the final exam otherwise the student might drop out and then you can if the seminar has a zero that is the class size becomes zero each termination otherwise this greater than zero.

There are more students even after this student has dropped, there are more students when you come back and continue each state. So at this node, you can you can easily see that the junction is taking place, junction is taking place here as well also it merges to the final state final pseudostate of the state machine diagram. So junction is a nice representation for the pseudostates.

(Refer Slide Time: 14:00)

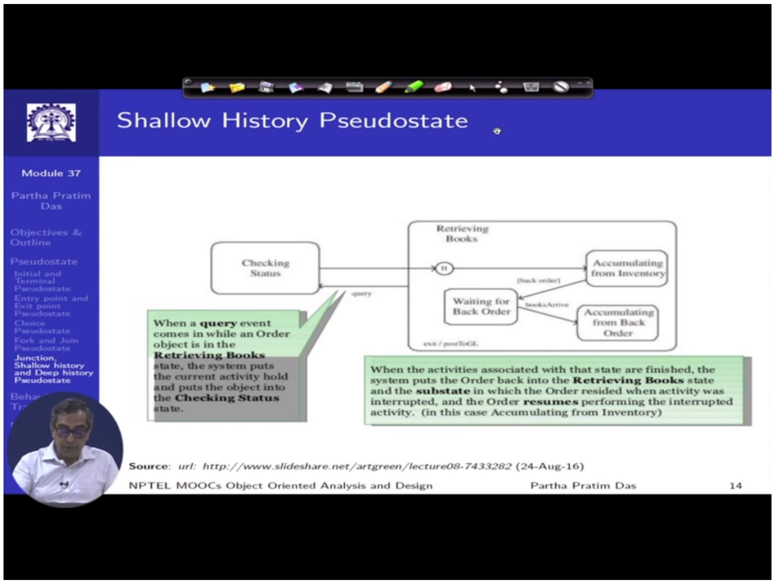


Here we are showing some shallow history pseudostate. I will not go through, there are couple of examples that they have put in. I am not going to detailing on the discussion of them I would expect you to really read through and understand them more. So here we are just showing that there is a history, there is a washing machine state transition state machine diagram for a washing machine. So there is in a washing state, there is a rinsing state where you put in more.

And more water and cleans your clothes and the finally spinning state finally take away water from the clothes and so this is the initial state, this is the final pseudostate. Now suppose if there is a power cut there is a power is put off then it goes to the power off state and then when you restore power it what you are showing when you restore power you restore power to get back to a shallow history pseudostate which means that just think about the practical situation, if the power have got disconnected when you are in washing state.

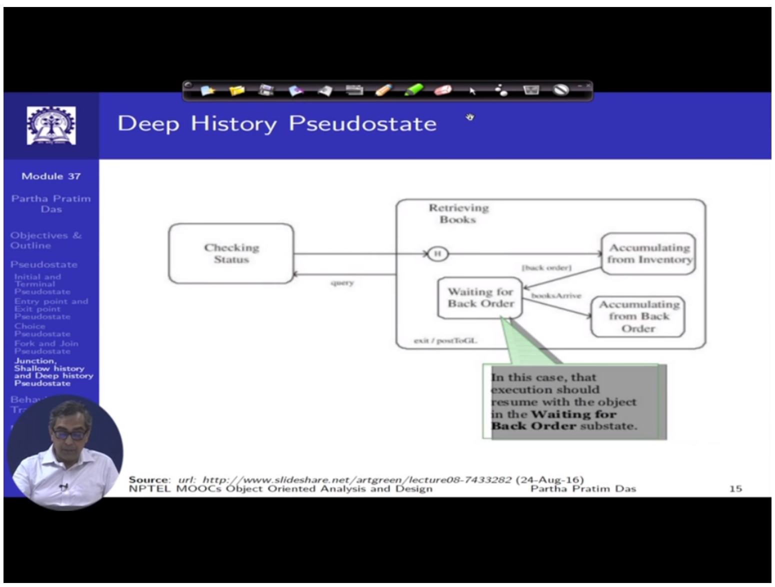
If that happens and when the power comes back you want the machine to restart in the washing state itself. You do not want the machine to be spinning because the rinsing has not happened. Whereas when the power went off if you are in the rinsing state you would like the machine to restart in the rinsing state otherwise if you are in the spinning then you would like to be started in spinning state. So this is the significance of this pseudo. So that is what you mean by history. So you remember try to remember the shallow history.

The shallow history is about the last active sub state that you were in and so when you had to be suspended because the power has back up and when you resume after the power has been restored, you restart in the last activity that you had been in, that is the shallow history that you have. In contrast you could have a deep history pseduostate which will not only depend on the last sub state but it might depend on the last sequence of sub states that you have transited. So I have put in some examples I will please go through them.
(Refer Slide Time: 16:32)



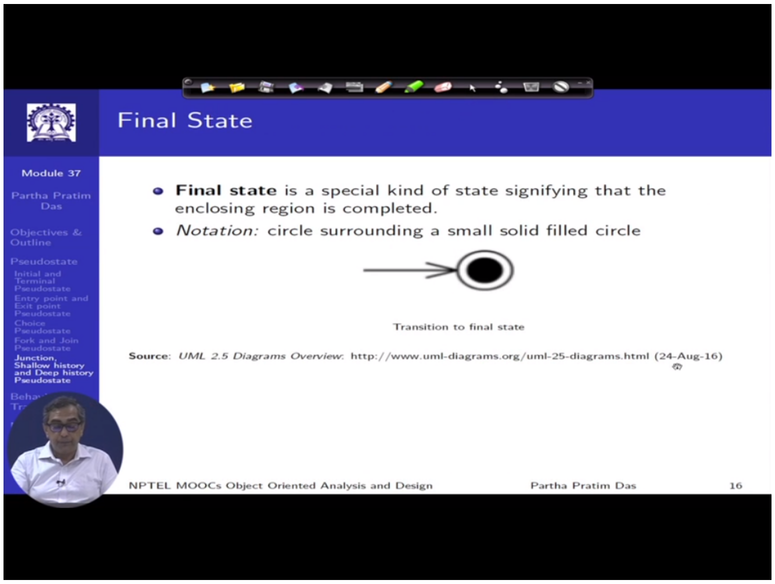
This is another example of shallow history pseudostate.

(Refer Slide Time: 16:38)



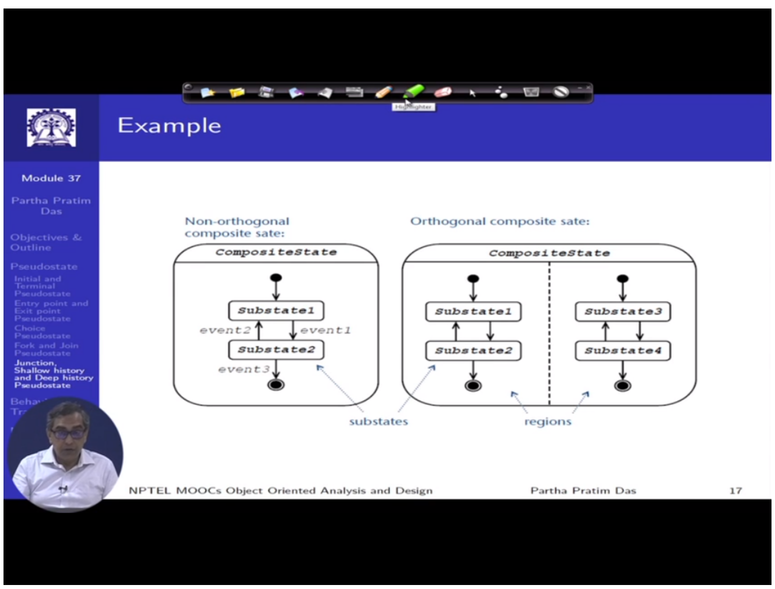
This is an example of deep history pseudostate. So from the presentation please go through them and try to understand them better.

(Refer Slide Time: 16:45)



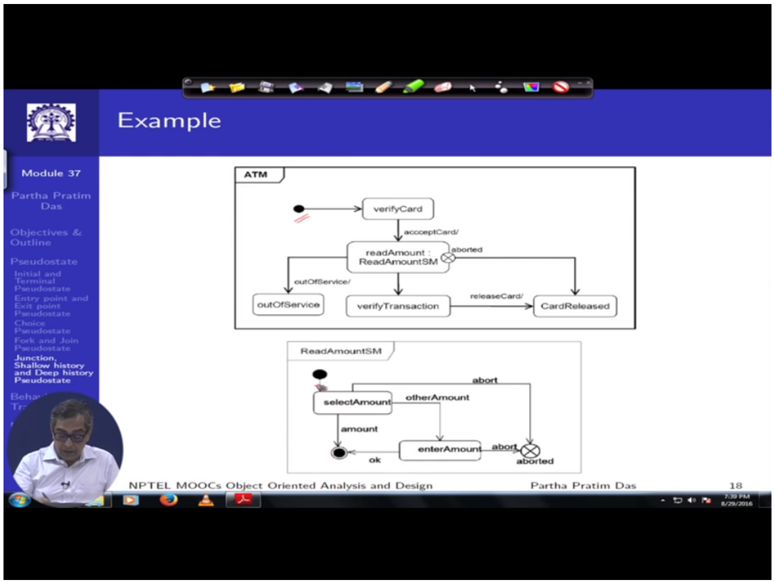
Now finally the there is a pseduostate called the final state which you have already seen in a number of places and once you reach this state then you know that the state machine diagram has actually come to a conclusion and there is no further state transitions to happen for this state machine diagram.

(Refer Slide Time: 17:08)



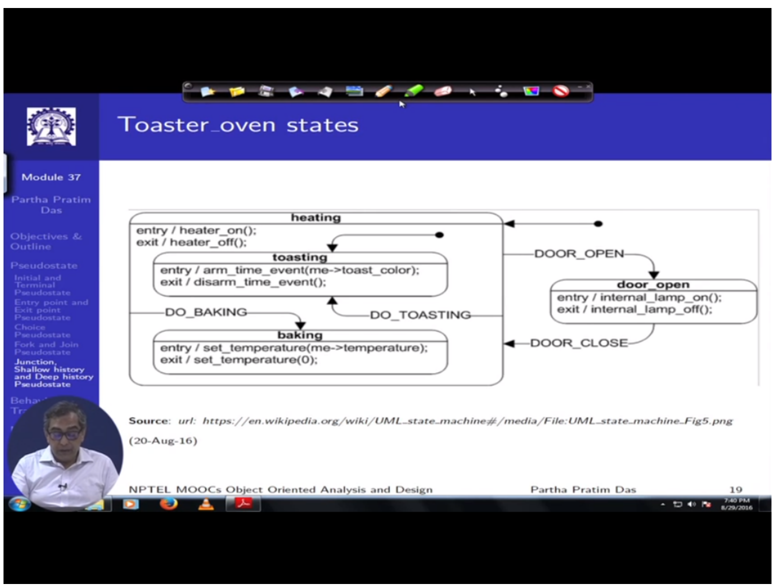
So threes are some examples showing you some pseudostate. So this is this is the initial pseudostate, this is the final pseudo state which we have been seen but regularly.

(Refer Slide Time: 17:23)



Then if we see here this is again the initial, the initial, the final. This is this you can say is an exit pseudostate. So this is the exit pseudostate which takes you out of the submission diagram.

(Refer Slide Time: 17:44)



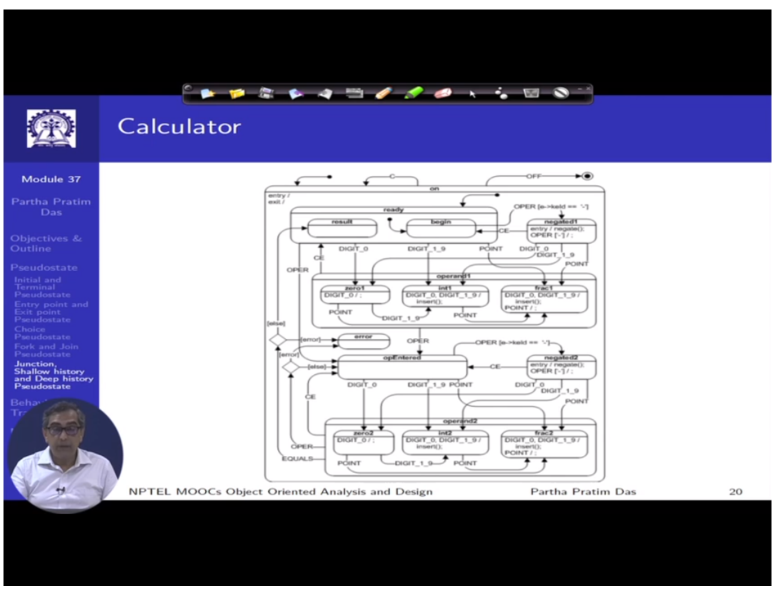
This is another toaster oven example. So you can see that, so what are the, if we identify first simple behavioral states, then this is the simple behavioral state, this is the simple behavioral state, this is the simple behavioral state. These states do not have any other do not need other states to work. Each one of them some have some entry and exit like toasting is entry is arm time event that you said that this is the kind of toast color they want.

And exit is some kind of a, that is the toast has happened for the specified period of time. So the timing timer is disengaged then the toasting actually stops. Then you have the entry for door open which is internal lamp being on and the exit is internal lamp being off. That is if you o open the toaster then the internal lamp will get on so that you can look inside and when you put the door closed then this will become off. So these are the entry exit points.

For the baking part, you will have a set temperature for entry and you have set temperature to default possibly you non baking temperature on exit. So these are your simple states and then you can see that these simple states have combined into a composite state of heating which has the entry heating on and heating off and for this composite state this is the start, of the of the whole state transition process which starts with toasting starts am sorry this is the start which brings it here.

So in this if you have the instruction to perform toasting then this is where toasting do toasting and start on this. If you have instructed to do baking the you start with this simple state machine and but when you get a door open event you go to go outside of this composite state and do the door open and once the door is closed you come back to the composite state. So this show using the composite states and the pseudostates I will describe between state machines involves here.

(Refer Slide Time: 20:21)



This is the little bit more you know detailed example but it is a very simple example I find this to be a interesting one; this is basically trying to de3scribe the calculator that we use. So this is the calculator so if you look, so what is a calculator is, where you can put in so I want to 35 plus 63 equal to 98. So what we do, I would first enter these digits which is basically operand 1,then I enter the operator, then I enter these digits which is operand 2,then I enter equal to, then I enter sum.

Then I can again enter another operator, enter an operand, get the result and so on. So this behavior we are trying to describe in terms of a finite machine diagram. So if you if you look into that then certainly you need sub machines or composite states which allow you to enter operands. You will find operand 1 composite state here. So which can either start and that has different simple states, 0, intt and fract, 0 is just when you enter a 0.

And intt is when you enter with starting with something like 1 to 9 and then you enter any of 0 or 1 to 9 and fraction is a decimal number. So if you enter a point you get it. So if you get the 0, you are in this simple state, if you get 1 to 9, you are in this simple state, if you get point you are in this simple state. In between if you get you are in this simple state you get a digit 1 to 9, you come to this and you continue to remain in that and you get a point you get to this state.

So this if you look into this operand 1 composite state that basically models how you enter the different digits of an operand. So you will have an operand 2 also which models how you enter different digits of the second operand. So if we so having said that if we rewrite this we get, so the overall action needs that I will enter the operand 1 which is basically with this and then I will enter the operator.

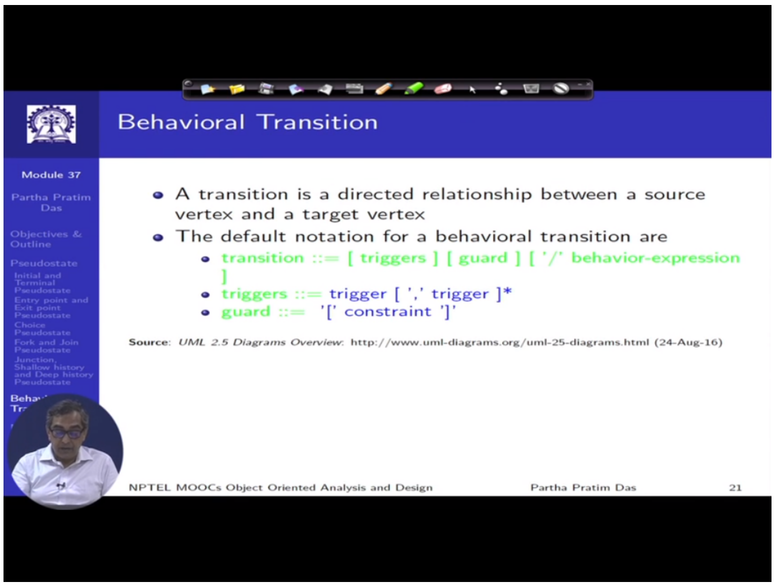
So this composite state has a transition on the operator which is basically another simple state which is operator entered and from that if this is entered and from that you have again transitions which takes you to the second composite state of taking in the next second operand. And once you press equal to key which is basically saying that you evaluate and on equal to goes and it checks we are if there is an error, it gives you an error.

For example if you have given say suppose you have given this then this gives error, then this will go to this simple error state. Otherwise it goes to the result state where it shows the result. and so shows you 98 here as a result and on this again you could you could actually start all over and enter a new expression or you could give an operator say plus 2, if you give an operator it gives you the operator simple state and then it enters, then you can enter the second operand again, the first operand by default with the results.

And then this way you can, there are there are further details and there are details in terms of what happens into ce, clear entry you could go back to the ready state of the calculator if you clear the whole thing you reset the whole operations back here and you can see that a operations of a calculator can be very nicely described in terms of a state machine diagram in this manner using simple states and composite states and transitions and different pseudostates you have this pseudostates, you have these pseudostates, you have basically the choice states which tell you which will go and so on.

So this is just , am just am just trying to give you different flavors of examples to understand that the state machine diagrams are really powerful to capture the dynamic behavior, discrete behavior of the system in a very simple manner.

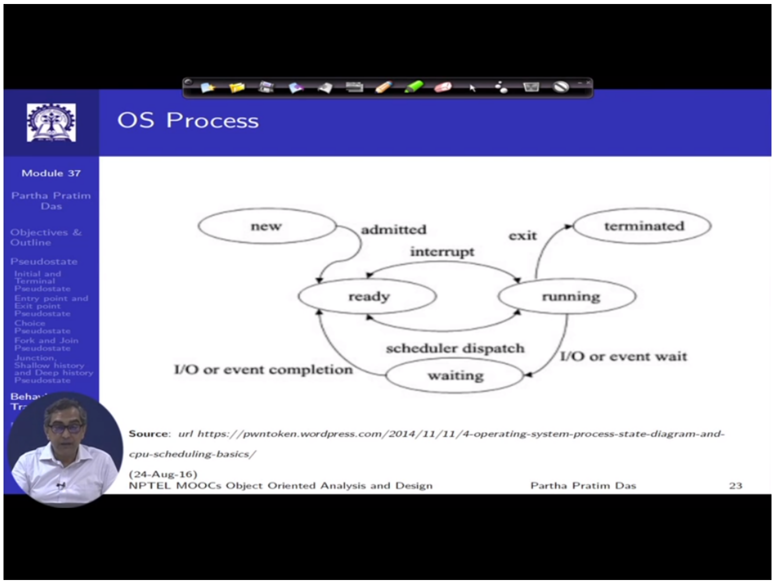
(Refer Slide Time: 25:37)



Besides the pseudostates you have the behavioral transition. The transition is a directed relationship between a source vertex and a target vertex which we have already seen. So these are the different transitions. So formally these are called the behavioral transitions where this is the source vertex and this is the target vertex, for this transition this is the source vertex, this is the target vertex and the transitions may have certain conditions, start conditions based on which the transition takes place.

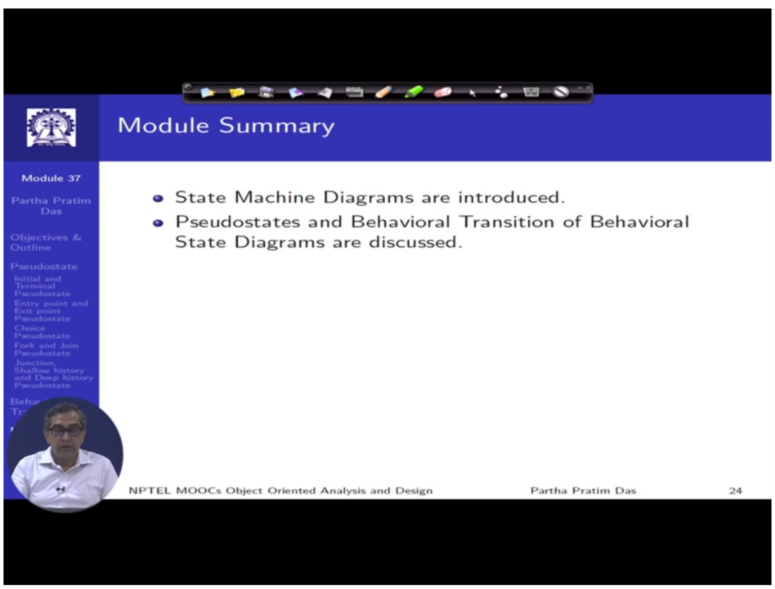
And further you can say that if I mean am in state maintenance and a failure has happen, this event has happen then I take this behavioral transition into an other behavioral state, out of service. So by this we can conclude the state machine diagram as on whole.

(Refer Slide Time: 26:33)



Just as an example I just put up a state machine which is which otherwise should be very known to all of you having computer science background, this is typical process state machine of an operating system. So the same representation will also work for the here.

(Refer Slide Time: 26:53)



So to summarize we have been talking continuing to discuss about the state machine diagrams and then in this module beyond the behavioral states, we have talked about several kinds of pseudostates and explained different examples and also talked about the behavioral transitions and we will continue in the next module talk about the whole state machines.