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

Lecture – 13
Elements of Object Model (Major): Abstraction and Encapsulation (Contd.)

(Refer Slide Time: 00:33)

Welcome back to module 8 of object-oriented analysis and design. We are discussing on elements of object model, we have talked about one major element of abstraction, we have introduced the notion of encapsulation. There is a second major element. We will continue from there, we will little bit in this see how encapsulation and abstraction interplay between themselves. So in encapsulation is a, for an abstraction and encapsulation does 2 things.

It separates the contractual interface and the implementation. This is a very very key concept to understand and this is the first point from where you should start realizing that the different elements, major and later we will see as well as minor elements of object models are not independent aspects. They are not competitive, they are more cooperative. In that one every major element supports the other major elements in terms of the design so we have seen the abstraction of concepts, the key concepts, different hierarchy of that.

Now encapsulation actually helps abstraction by in terms of this. That is what I want is, I do not want when I implement that abstract concept, I want to kind of as I say, hide that implementation. A concept,

a term you may have heard quite frequently is encapsulation basically hides that. You have put it all together into the packaging of the hard drive, you know that there are rotating disks but it is hidden, you cannot see that but at the same time for a system to, if everything is hidden then the system has no use.

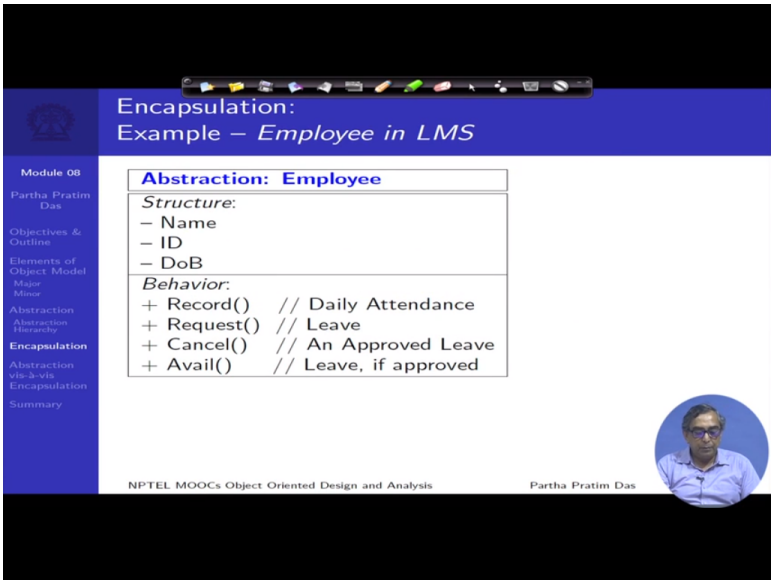
But for the system to be useful, we need to put an interface in which I write here as contractual interface. Why is it called contractual? It is called contractual because my actual implementation inside, my hard disk could be rotating disks, it could be magnetic disks, it could be optical disks, it could be solid state hard disk and so on. But my interface of being able to put an address and get the data out from that address or write a specific data to an address cannot change because then the hard disk as an abstraction fails to provide the services.

A stack may change in its implementation but the basic operations of push pop top and empty of a stack cannot change. So what we say is an interface that others can use to view the system, to model the system has to be invariant with respect to the implementation and that is kind of the concept kind of ease that in a client-server model interface is what your clients get to see, who wants to use your object, who wants to use that object.

And you have to make a commitment to the client objects that I am not going to change that giving rise to the notion that this kind of interfaces are contractual in nature. These are not contractual in the sense that you sign anything on the stamp paper but these are contractual in terms of commitments that you are making and then that process, the data encapsulation basically ri give rise to data hiding so the implementation is not exactly known to the outsiders but the contractual interface allows you to use that abstraction.

And so if we look, go from abstraction to encapsulation basically we are dealing with the same thing but in a different package. In abstraction, we have not at all talked about the implementation itself. But in encapsulation primarily talks about that implementation and wants to keep it away from the public view.

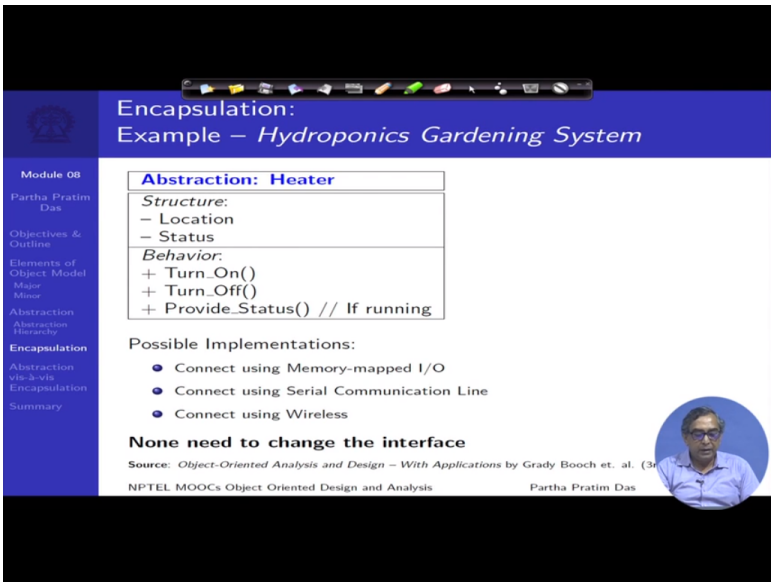
(Refer Slide Time: 04:33)



So there are different could be different example. For example here going back to the leaf management system, this is the employees, we saw it as an abstraction you can look at the same thing as encapsulated as I mentioned that these symbols will mean that this these are part of the implementation which are hidden, which cannot be seen by others. This show that this is public or this is contractual interface which other can use.

So given an employee, you may not exactly know how it stores the name, the id and date of birth and so on but you will always be able to use that object to record the attendance or request to leave and so on.

(Refer Slide Time: 05:26)



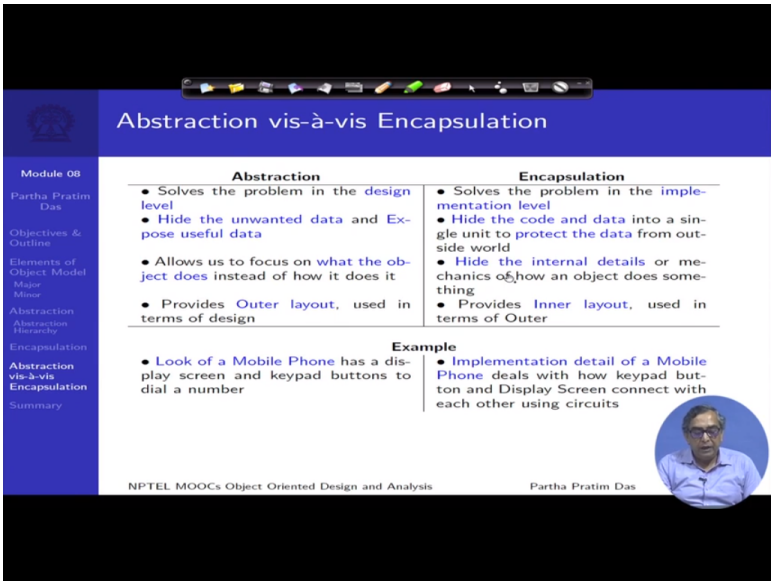
In the hydroponic gardening system, we can think about a heater. It is a abstraction is one of the other

abstractions and it has similarly a structure or implementation of location and status and the functionality or contractual interface of being turned on being turned off or just checking kind of a Boolean operation which checks whether the heater is running or it is off. Now it is quite possible that actually the implementation of this heater.

In terms of the system could vary in terms of the actual make of the heater even in terms of the way the computer system, your control system will control this heater or connect to that heater. You could connect using a memory mapped I/O. I mean if you have some idea about computer architecture or microprocessor will understand what is memory mapped I/O. It does not matter if you do not understand that buy it could also connect using a serial communication line.

Or it could be that you just want to connect it to a wireless communication stack so that there is no physical wires that you would like to turn it on and there could be various different possible implementations and the key point in the whole thing is that there is no need to change the interface that remains contractually fixed.

(Refer Slide Time: 06:50)



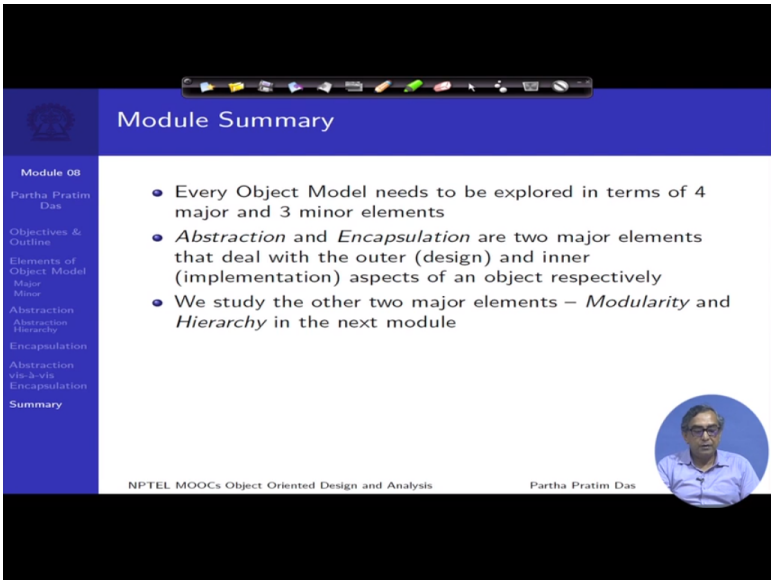
Finally if you look into abstraction and encapsulation as complementary elements of object model, we will see certain characteristics that abstraction is primarily at a design level whereas encapsulation is talking more in the implementation level. So if you are talking of abstraction you may be putting some hints about implementation but you are necessarily talking about the contractual interface because that is the design.

If you are talking of encapsulation, you will certainly have to honor the contractual interface, the abstraction, the concept that the abstraction has given you but you are certainly talking about how to implement that system, how to package it together. Abstraction on one side will hide unwanted data and focus of the on the useful data, we have seen that. In contrast encapsulation will hide the code and the data and protect that data. These are basic encapsulation basic purposes, data hiding.

Whereas abstraction will hide aspects which is not interesting for that concept. Encapsulation does not go back and rip that apart. Encapsulation starts with whatever abstraction has given it in with it works only within that paradigm but now in terms of the code and data it may choose to expose some and hide the rest. Abstraction will allow focusing on what the object does instead of how it does it. So what is again it is kind of the same thing being said in little bit different terms.

Encapsulation looks at the internal details and therefore abstraction is provides the outer layout whereas encapsulation provides the inner layout of any object model. For example if you look at a example of a mobile phone, it has a display screen, keyboard, buttons and you can use that to dial a number and make a phone call in terms of an abstract view these are the contractual interfaces but how that those are implemented? How that keypad is implemented?

How does the display function, are parts of the encapsulation which is hidden from our common view. So this is so you can see that encapsulation and abstraction are very strongly related and complementary concepts are complementary elements of the object model which we will deal with in terms of our object model design.
(Refer Slide Time: 09:30)



So in summary we had we have seen that every object model needs to be explored in terms of 4 major and 3 minor elements. And we have in this module explored on the 2 elements, abstraction and encapsulation one which deals with the outer or the design aspect of object models and the other encapsulation deals with the inner or the implantation aspect of the object model. In the next module we will study about the other 2 major elements of modularity and hierarchy.