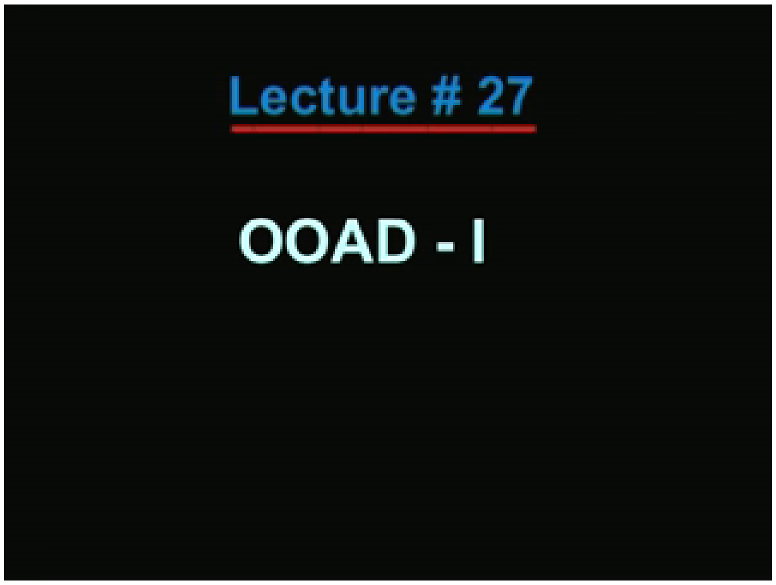
Management Information System

Prof. B. Mahanty
Department of Industrial Engineering & Management
Indian Institute of Technology, Kharagpur

Lecture - 27
OOAD - I

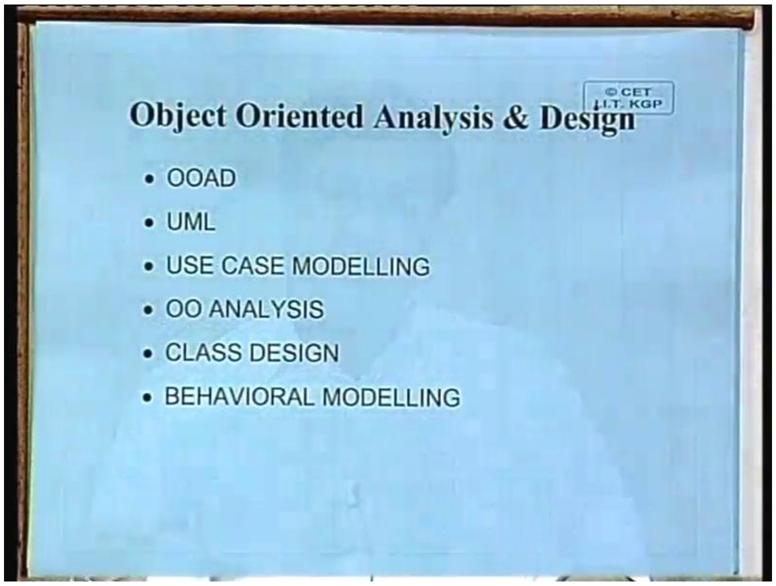
(Refer Slide Time: 00:45)



Lecture # 27

OOAD - I

(Refer Slide Time: 01:07)



Object Oriented Analysis & Design

- OOAD
- UML
- USE CASE MODELLING
- OO ANALYSIS
- CLASS DESIGN
- BEHAVIORAL MODELLING

Let us begin with a new concept - that is object oriented analysis and design; in short, OOAD Now, in this particular concept, I mean...we shall be covering the following topics. First is a general concept of what is the object oriented analysis and design and the object oriented analysis and design concepts are intertwined with the concept of UML. That is the unified modeling language. So, a rough idea of object oriented analysis and design along with the concept of UML.

Then we shall go directly to the modeling aspect of object oriented analysis and design. The very first one is the use case modeling, right? See, the idea of use case modeling is different from your traditional data flow based model, right? The basic idea of a use case is use case is like a business process. So, you have to see the idea of a use case purely from the view of a user, right? So essentially, the use case model, model depicts the use case and the actors. So, we will go into detail later on. Then, we certain other things about the object oriented analysis; how exactly from the use case you go ahead with the analysis part of the thing, how you basically designed the class, the attributes of the class and the operations that are associated with a class, then the design of the class, class design, right?

So, when you have a number of classes in a particular system, information system, how the classes are connected to one another and how they together create, let us say, a class diagram or a VOPC that is called a view of participating classes, right? VOPC.

So, this is the thing. Then on the other side, you have the behavioral modeling. Behavioral modeling means we define the classes and what are their attributes and their operations; but when it comes to operations, how the operation of one class is connected to another class, you see, that is where is the major departure from process based thinking. The basic idea of the object based thinking and the process based thinking is, in a process based thinking everything happens according to a process, logic; whereas in object based thinking, you assign objects with behavior, alright? So, it is like...there are two students let us say. Take a simple case of two students are fighting amongst one another or two children are fighting, basically. So in a process based logic, you have to give the entire logic. For example, if he kicks, if one boy kicks the other boy, then the other boy will react in two ways, alright?

So, what are the two ways? If he goes for this, what are the likely outcomes of the other boy? So, you have to think of the entire sequence of fighting and essentially put down that in the form of an ‘if-then-else’ logic whereas in object oriented thinking, you have to think that this is one boy, he has certain attributes, he has certain behavioral patterns, alright? So, if he does...he does this, he does this, he does this. On the other hand, the second boy is also another boy. He has an object. He is like an object and he also has certain operations. He does certain things.

Then you have to go for the behavioral modeling in the sense, how these two boys...when this boy does a...what the second boy does, alright? So, then you define all these; but the actual fighting between the two boys will come out of these behavior patterns. What exactly will happen, that will probably be dependent on perhaps the random number chosen by you. If you are simulating it as a discrete event simulation model, then that is the thing. So, the developer himself does not know what could be the outcome of this particular let us say, the fight sequence, alright? So, this is a trivial example.

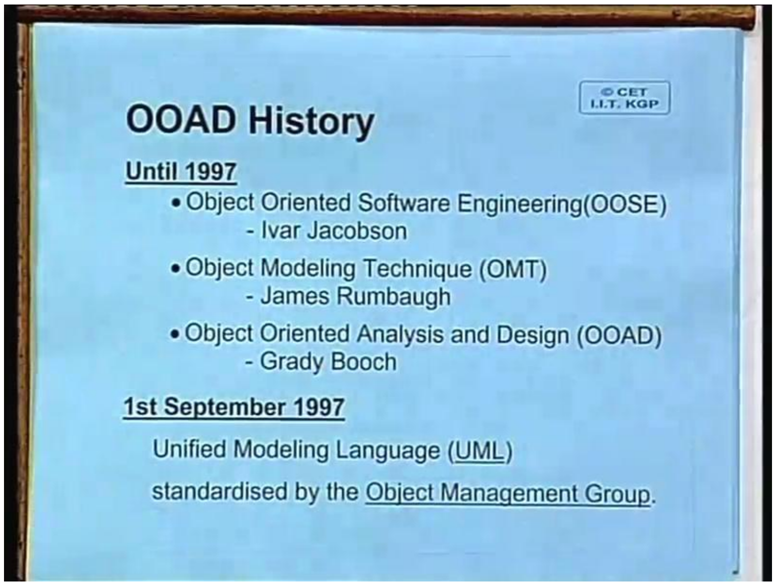
Let us take a slightly better example. Let us think about a runway, alright? So, a runway...sometimes like a Mumbai airport, you might have some idea that it is a very very busy airport, right? Therefore, the runway is also heavily utilized, heavily utilized. So sometimes, the planes have to wait, alright? Now, if you want to simulate this particular runway, what you can probably do? You can define the runway. What is a runway? What are its attributes and what are the conditions for a plane to land under these conditions?

If the runway is free, there are not more than...see, runway is a very large one and therefore you know, if there is only more than two planes in the runway, then the third plane cannot land. Suppose you define this that at a time there should be only two planes. If there are three planes on the runway then, another plane cannot land in the runway, alright? Similarly, for the plane...the plane...for plane to land, it has to fulfill certain conditionalities, right? It has to arrive at the destination, it has to reach certain height, it has to see the weather is clear and all these things. So now, exactly whether what will be the...see, you may have a some sort of behavior pattern. That, if it is raining and the runway is not free and something, then the plane has to wait.

So, you have all these conditionalities very clearly given, alright? Then, you just let it happen. You generate certain planes. You generate an initial condition of the runway and then you start simulating what exactly will come out of this. That is not known, right? So, there is no need to write a sequence of 'if-then-else' logics...logic, when you try to go for an object oriented design. In another sense, you can think of an object oriented design as like...as a huge menu system, like working in Microsoft Word.

What has happening, what you will be doing is your choice. You know, everything is there in the menu. You write something. You want to make it bold, you press the bold icon. So, bold icon is...as an object, what is a bold icon? It is nothing but an object, alright? So, this icon has got certain attributes. It looks like something...a 'B' will be written on that in capital, etcetera, etcetera; and it has got certain operations. What are the operations? That is, it makes the text highlighted, bold. So, these are the things that we shall be discussing.

(Refer Slide Time: 08:53)


© CET I.I.T. KGP

OOAD History

Until 1997

- Object Oriented Software Engineering(OOSE)
  - Ivar Jacobson
- Object Modeling Technique (OMT)
  - James Rumbaugh
- Object Oriented Analysis and Design (OOAD)
  - Grady Booch

1st September 1997

Unified Modeling Language (UML)

standardised by the Object Management Group.

To begin with, let us discuss some little bit of history about object oriented analysis and design. See obviously, before we come to the Jacobson Rumbaugh and Booch, the concept of object orientation is as old as humanity, alright? Basically...it is surprising that people did not think of object oriented design first rather than thought about process logic first, alright? This is because, see, when let us say, a child first learns anything...how do they learn? Basically, they learn to associate certain conceptual thing to the given objects, alright? Something like, let us say, a toy car. What is a toy car? The toy car to a child is something that looks like some car has got some color, attractive color, has got four wheels and it actually runs, alright? So, it...the child will associate the car with these things.

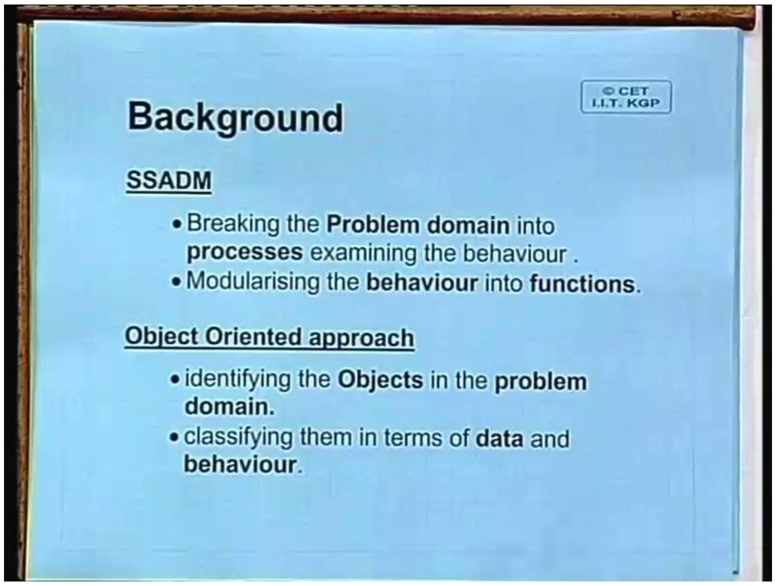
So these...you know, the basic learning process is basically trying to see a number of objects and try to assign certain attributes and operations to them. So, this is how the basic learning itself goes. So in that sense, object orientation is nothing very new. It was surprising that people did not start with it in the first place. Now although I have only put Jacobson Rumbaugh and Booch’s name, actually the development...there are many other people such as Codt. Codt has done a substantial amount of work towards subject orientation in the 80s, alright? Now during that time, the basic problem that arose...that there are so many methods. For example, we have the object oriented software engineering - OSE - by Ivar Jacobson, the object modeling technique - OMT - by James Rumbaugh, object oriented analysis and design by Grady Booch. There is a Booch method.

So you see, there are so many methods that people were confused. It is not that they are differing too much. There is definitely lot of commonalities amongst all these methods; but nevertheless, there was a lot of differences in notations, lot of differences in notations. So, what the object management group did, the OMG brought all the three people together right, the Jacobson, Rumbaugh and Booch, and together they started new concept called unified modeling language or the UML, alright?

Please understand - the UML is not really a concept. UML is a notational language, right? It is a notational language. Basic idea of UML is, it gives you a notational approach, a standardized notation to write, object oriented analysis and design diagrams, alright? So as a concept, we still call it object oriented analysis and design; but UML could be used to, you know, denote the diagrams. That is the basic idea, is it clear? So, since UML has become the de facto standard, see...Why de facto? Because OMG, it is not that everybody has formally or legally agreed that this is the formal standard.

It is not a formal standard. In that sense, it is a de facto or informal standard of object oriented; and this is a design as on today, alright? Very recently, UML2 has also been released in 2002 or 3, I think; and this UML2, I mean, differences are very little but nevertheless the latest standard if someone asks, we can say, it is UML 2 and not really just UML.

(Refer Slide Time: 13:17)



Background

© CET I.I.T. KGP

SSADM

- Breaking the **Problem domain** into **processes** examining the behaviour .
- Modularising the **behaviour** into **functions**.

Object Oriented approach

- identifying the **Objects** in the **problem domain.**
- classifying them in terms of **data** and **behaviour**.

Before going to the basic concepts, let us see little difference between your structure system analysis and design method - SSADM versus object oriented approach. Once again, basically in SSADM we break the problem domain into processes, examining the behavior and modularizing the behavior into functions; whereas in object oriented approach, we identify the objects in the problem domain and classifying them in terms of data and behavior.

So you see, what is happening in structured system? The behavior is modeled into functions. What about data? We do not talk about the data in the process based approach. The data is used separately in what is known as the data modeling. So, that is what I have explained in great detail. That structure system method has got two components: one is the data flow modeling, basically modeling - the processes and the data modeling which models the data; whereas the object oriented approach couples both the approaches: the data as well as data flow. So, it models data and behavior and put them together in objects. One can ask, why? What is the advantage and if possible are there any disadvantages? The advantage is, when you put the data and the behavior together into the forms of classes or objects, can you tell me what is the biggest advantage?

So, what...see, he has said encapsulation. Encapsulation - we will explain what is encapsulation means. We will go little later, but whether it is encapsulation or not, the basic idea of doing this is basically have the advantage of rework. See, reworking is easier, much much easier, alright? So reworking is much easier. Reuse is possible, right? So, that is the basic advantage, great advantage in reworking and reusing. See, moment you define an object or a class in a very generalized sense, this object or the class or this particular class - let us say, will I think...I must tell you the difference between class and object, right? A class consists of a number of objects. So, if I say an employee is a class, then an individual employee is an object, alright?

So, if S. Das, D. Sur, P. Reddy are 3 employees, then we can say, each of these employee are actually objects and together they constitute a employee class. Is it okay? So, that is the thing between individual objects. There will be differences. So, when you talk about class, we talk about attributes. Attribute is: name all objects; will have a name but the name could be different. One object may have name P. Reddy and another object may have another name, alright?

So, that is the difference between classes and objects; but when it comes to the conceptual thing, we do not use the class word very much. But when it comes to implementation, we must understand that there is a big difference between class and objects. So, identifying the objects in the problem, domain classifying them in terms of data and behavior - that is the basic idea of object oriented approach.

So, I was talking about the reuse. So, reuse is the basic advantage of object oriented analysis and why? Because, if you could have very specified class libraries defining the specific uses of a particular thing. For example, if you define a menu class, what is a menu class? The menu class...essentially pops of a menu of variable number of items and you can click on one of them. Think about it. This menu class - you can use almost in any application, alright?

You have been, you have taken care to see that menu, the individual, the color to the menu to some extent may be shape and size and also the number of menu items can also vary, you can vary. So, this generalization has been actually done. So, if this is a separate library, if you are having another application, you do not have to write this menu class again. You can simply reuse that menu class. See, this is the basic advantage that you can have - reuse.

In fact, reuse and component based design, reuse and component based design has become the of today's. What is component based design? If you have reusable components. For example, the menu, menu is just a single class but together with few such classes, you can design a component, right? For example, a component for window management, a component for a given calculation, may be the facility, location, calculations. So, all you have to do is put together all these components and then develop a little bit of your own to bring or develop a customized software, right?

So, what happens? Your development time reduces; your dependent on certain kind of developers, expert developers in a certain area reduces, alright? These are the basic advantages of object oriented analysis. On the flip side, What is the flip side of this? That you can do lot of things; but for that you require expert people, trained people, alright? People who really understand objective orientation, slowly, initial, after the initial euphoria is over, people are being increasingly aware that object oriented analysis is not a very simple thing; is not a very simple thing. It requires, you know, little bit of maturity. The people should be well trained; and if you allow someone who does not understand object orientation much and he wants to go for object oriented design, most likely he would come up with something which is as bad as the process based design, process based design.

The advantage is, it is going on for long time, has lot of expertise available with people; all the old systems are developed through the process based thinking, because you see, you cannot throw away all that has been developed so far, alright? There are, you know, thousands and why thousand, millions and millions of dollars have been invested into information system, into the hardware and important applications are running on them. Think about large company running their payroll of 50 thousand people in an application developed by using process based logic. Now, will you throw the application and say, because it is not object oriented, we would not use it. You cannot do that because it again requires lots and lots of investments. Why should people invest for something which is already there, alright? The payroll is working, everything is fine. People are getting their pay slip towards end of the day. That is more important than application, is it not? What is the purpose of the application? To see that the payroll process runs in a very smooth manner, is it clear?

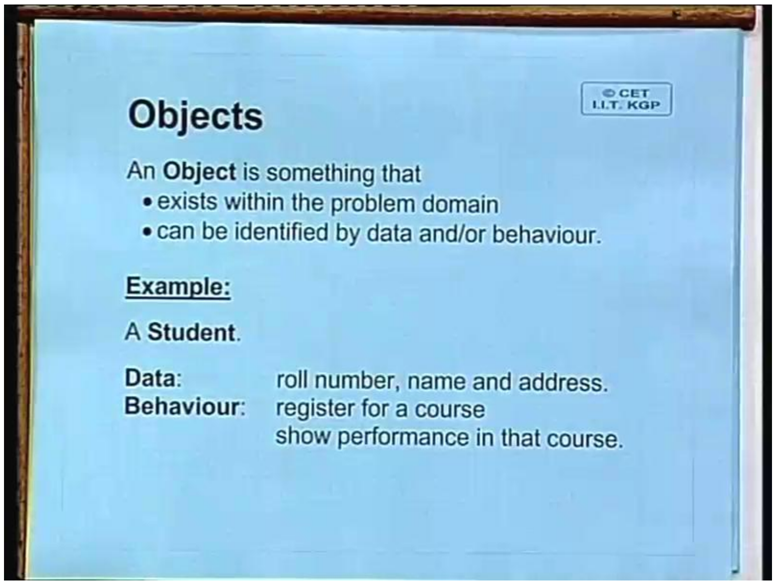
So, this is the great challenge all the more for India because if you really look at the Indian scenario, a large chunk of the software consultancy that India gets are essentially in the domain of your, you know, maintaining old systems, transfer of a system from an one old system to another new system. Basically no change in the application. Simply you are putting the system, from one system to another, alright?

These are the kind of jobs. I am not saying these are the only jobs India is getting these but this is a substantial part of the job that India is getting in the name of software consultancy, alright? So naturally speaking, you have to be aware of all the techniques and technologies which have been not only being used today but has also been used in the past, right? For example, Cobol - the Cobol programming - everybody thinks these is no future of Cobol but nevertheless who knows, you might be in a project...if you join a software company, where Cobol is being used, alright?

So, these are the pros and cons and one should be very aware that what is the advantage that you are getting, alright? Some systems are very process oriented where even if you do object orientation, probably not much to be gained, not much to be gained; whereas, something which basically you see, object oriented approach, basically is an offshoot of the data approach.

The data modeling approach - that is what I said while discussing the database management systems, right? So, anything that requires a high involvement of data, high involvement of a database and that is the major application; not the process logic alone, right? Naturally, object orientation analysis and design will be the appropriate approach. Now, when it comes to an object, an object is something that exists within the problem domain and can be identified by its data and or behavior. So example - a student, it is data is roll number, name and address, behavior register for a course, show performance in that course. So, that is the basic idea of an object.

(Refer Slide Time: 24:49)



© CET I.I.T. KGP

Objects

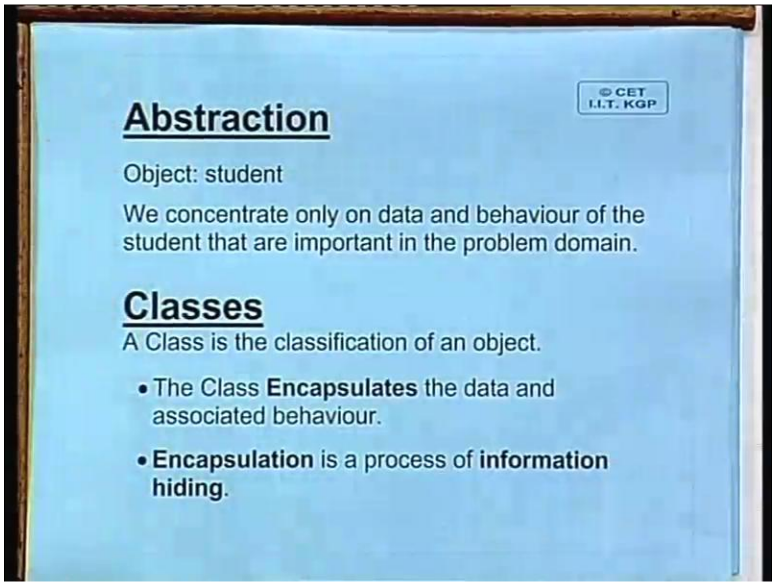
An **Object** is something that
- exists within the problem domain
- can be identified by data and/or behaviour.

Example:

A **Student**.

Data:	roll number, name and address.
Behaviour:	register for a course
	show performance in that course.

(Refer Slide Time: 25:02)



© CET I.I.T. KGP

Abstraction

Object: student

We concentrate only on data and behaviour of the student that are important in the problem domain.

Classes

A Class is the classification of an object.

- The Class **Encapsulates** the data and associated behaviour.
- **Encapsulation** is a process of **information hiding**.

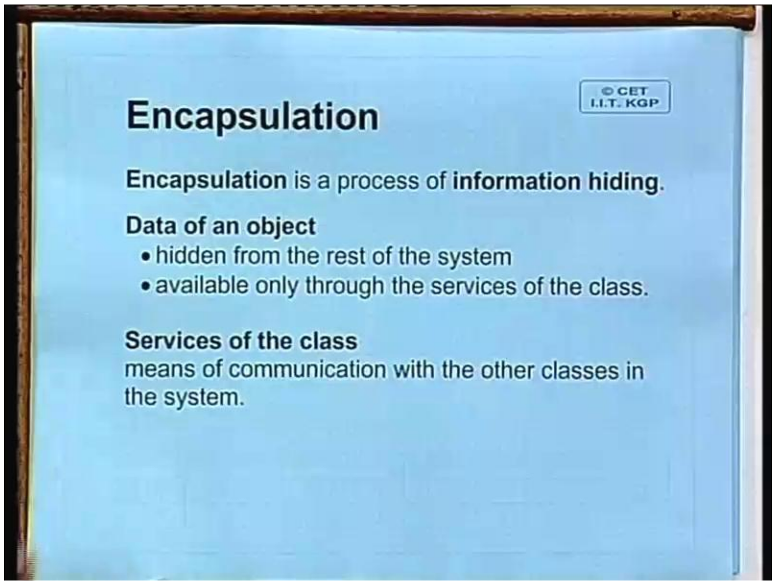
Then the very important concept of abstraction. See, abstraction essentially means that when it comes to a student, a student can have lots and lots of data and lots of lots of behavior; but are we going to use all of that? We are going to use only that aspect of student which are important to us in a given context. Is it all right in a given context? For example, if say for example, we say that a robot, a robot is a model of a person, is a model of a person.

Now, suppose a robot is just a gripper. All it does is it looks like a hand and grips certain object like this and it basically throws it, puts it somewhere else, right? It grips something, picks it up and puts it somewhere. So, {ho} ((00:26:14 min)) how does it look like? It looks more like a hand; and we are say...telling that this is the model of a person.

So, does the hand is a person? No, hand is not a person, right? So basically, that particular gripper is not a person; but an abstraction of a person is an abstraction of a person, alright? So, that is the basic idea of an abstraction. An abstraction is a component or a very important part of any object oriented analysis, right? So, that is what, the second important concept is that of classes. So, class is the classification of an object like I said. The student, when I say student is an object, I mean to say a particular student. A particular student may be, say, Sur. We have...suppose S, Sur...we have used that S Sur name many times. So, S Sur is a particular student, alright?

So, we concentrate only on data and behavior of the student that are important in the problem domain. That is the obstruction. So, when it comes to classes...but all students together, what it represents is, the...is a class...the class encapsulate, encapsulates the data and associated behavior. So you see, encapsulation word comes here. It is like a capsule know; you...if you open the capsule, two portions of the capsule, you find inside the things, the medicines, whatever is available. Similarly in this case, if you open the capsule of the class, what you shall see is the data and associated behavior for that particular student. Now, why it has to be put inside a capsule? So that it can be separated out from the rest of the objects and it can be reused later on. That is the idea. So, encapsulation is a process of information hiding, information hiding.

(Refer Slide Time: 28:26)



Encapsulation

© CET I.I.T. KGP

Encapsulation is a process of **information hiding**.

Data of an object
- hidden from the rest of the system
- available only through the services of the class.

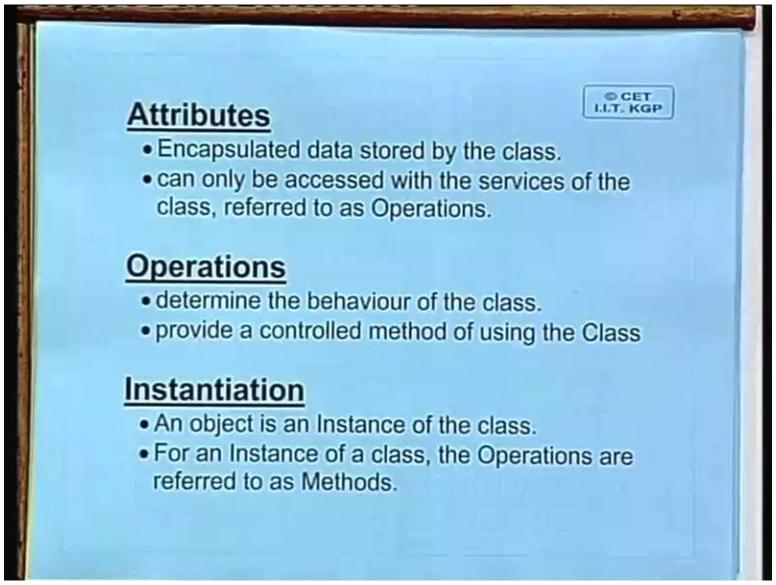
Services of the class
means of communication with the other classes in the system.

You are basically hiding the information. Encapsulation is a process of information hiding. The data of an object hidden from the rest of the system and available only through the services of the class, alright? The services of the class means, that is the means of communication with other classes in the system. So, how the classes will interact? The classes will interact only through the services of the class.

So, every class will have certain services and this we will discuss. How the services can actually be brought about and...otherwise, you cannot know anything of the class, right? So, it is like you are student. Unless you... unless I tell you what is your name, then you have your services, that you tell your name; only then I know your name. Otherwise, there is no way I can know your name, alright?

That is the basic idea. That is the...there is the concept of information hiding. The data is inside encapsulated and that data can be accessed only through the services of the class, alright? We will discuss more and more about this because encapsulation is a very key property of object orientation.

(Refer Slide Time: 29:46)


© CET I.I.T. KGP

Attributes
- Encapsulated data stored by the class.
- can only be accessed with the services of the class, referred to as Operations.

Operations
- determine the behaviour of the class.
- provide a controlled method of using the Class

Instantiation
- An object is an Instance of the class.
- For an Instance of a class, the Operations are referred to as Methods.

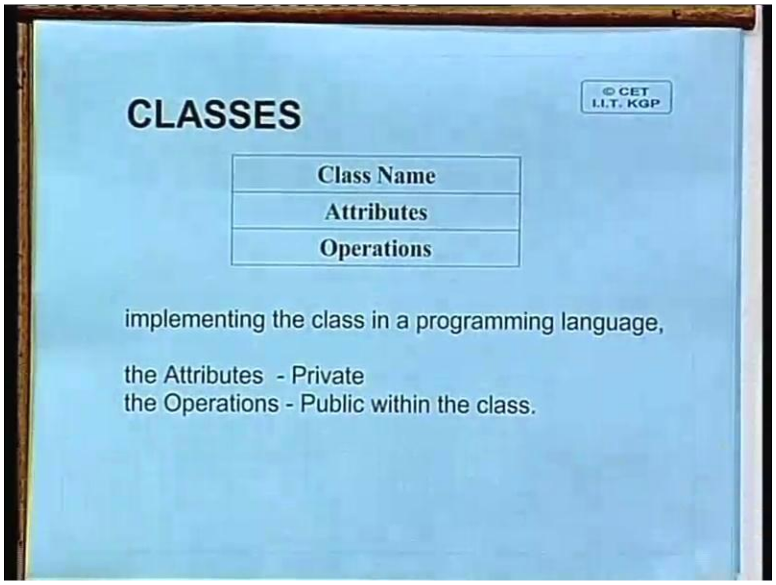
You see, we are using so many different words but we should be careful about exactly which words we are using. For example, we are using the word ‘data’. We have used the word ‘behavior’. Somebody can use the word ‘method’; but essentially we shall use attributes and operations for the data and behavior when we are talking about a class is all right the method the word method is usually to be used when we talk about an instance of the class for example an object.

So, these are the things that is explained here. So, encapsulated data stored by the class is an attribute. It can only be accessed with the services of the class referred to as operations. So, if you go back to our past slide just for a minute, see...data is hidden from the rest of the system; available only through the services of the class. And the services of the class is the means of communication with the other classes in the system; and these services of the class is nothing but the operations, alright? What is an operation that provides the services of the class? Through the operations, you can determine the behavior of the class and provide a control method of using the class, right?

So, we have the attributes and we have the operations. Then there should be a process by which we can instantiate a particular object of the class, because a class is like a common concept, like a student. Individual students are objects. So, how to instantiate? Anybody who has used Java programming...what is the command? How to instantiate a class in Java? The class... What is the command? Defining a class... Not defining a class...how to instantiate a class. New...

Yes, the new command, right? So, almost all of you have written Java programs. So, the new comment...the new command is essentially given to instantiate a class. That means create a new object, create another object of that particular class. So, then again for an instance of a class, the operations are referred to as methods. So you see, that is the difference between operations and methods. We talk about methods when we talk about objects; we talk about operations when we talk about the class. Then, let us go back. Go to the actual thing. That is the classes. The classes usually are represented in the form of a rectangle, in the form of a rectangle.

(Refer Slide Time: 32:36)



© CET I.I.T. KGP

CLASSES

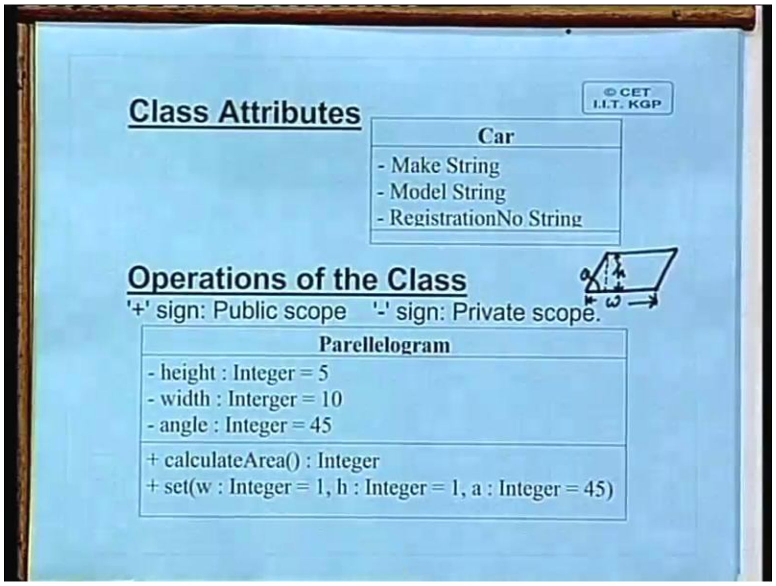
Class Name
Attributes
Operations

implementing the class in a programming language,

the Attributes - Private
the Operations - Public within the class.

The rectangle is to be divided into three parts: top portion we write the class name, middle portion, we write the attributes and the last portion rectangle, we write the operations. So implementing the class in a programming language, we require the attributes to be private and the operations to be public. Within the class, see, the private and public is important. The attributes - if you make them as private, they...what is the idea to make the attributes private? So that they are not accessible from outside. That means they are truly encapsulated. The operations however has to be public because anybody may like to use those. For example, you... I will give examples that will be easier to understand.

(Refer Slide Time: 33:38)


Class Attributes

Car
- Make String - Model String - RegistrationNo String

Operations of the Class

'+' sign: Public scope '-' sign: Private scope.

Parellelogram
- height : Integer = 5 - width : Interger = 10 - angle : Integer = 45
+ calculateArea() : Integer + set(w : Integer = 1, h : Integer = 1, a : Integer = 45)

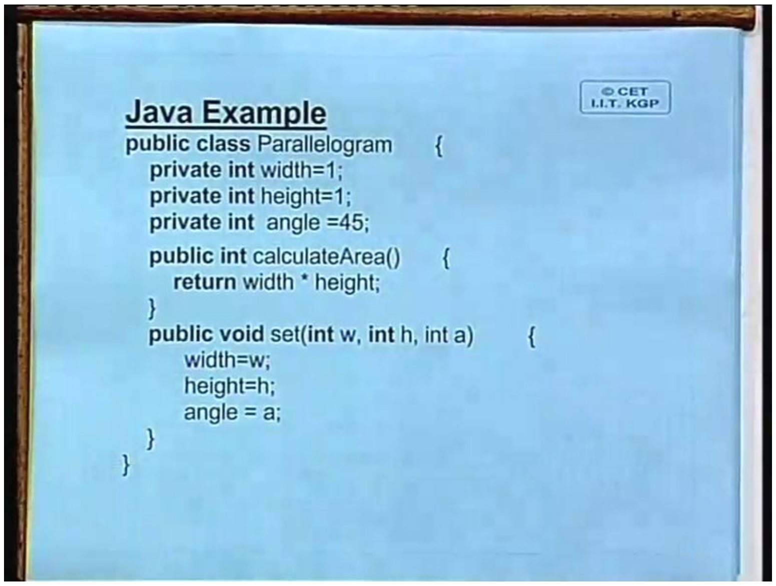
© CET
I.I.T. KGP
α
h
ω

So, let us take a car. The car has got, let us say, three attributes - its make, model and registration number, right? Make as a string, model as a string, registration number as a string, alright? So, this car is a class name. These are the attributes of the car. So, I have not shown any operations on the car now, operations of the class. So, this is another example. Let us say - a parallelogram. How to represent the parallelogram? Parallelogram has a height and width and an angle. So, like... suppose this is a parallelogram. So, this...in this parallelogram, we have to have a height. Say, this is its height. Then, it has its width and you have to define an angle. So, a parallelogram probably can be explained in terms of its height, its width and the angle. See, if we have these three things alright, you first draw the base, then you put the angle and you go up to the point where you reach that height. Then, rest of the things can be completed.

So, if these three things are there, you can construct the parallelogram. Then the parallelogram has got two more things. One is a...two operations - we have defined - one is the 'calculate area' and another is 'set its width height' and you are...basically, this is like a initialization, right? You can set w to 1, h to 1, as both are integer angle. Also, an integer, let us say, 45 degree, right? So, these are the two functions which are defined for the 'calculate area'. So, these are some of the...what are the operations of this class? Calculate area and set.

Now, this is a simple, very simple Java example. From Java programming point of view, how this parallelogram can be represented in a Java program? So, what we do? Public class parallelogram - the bracket starts, then private int width equal to 1, private int height equal to 1, private int angle equal to 45, then public int calculate area. See, we have try to put the operations as public.

(Refer Slide Time: 35:56)



Java Example

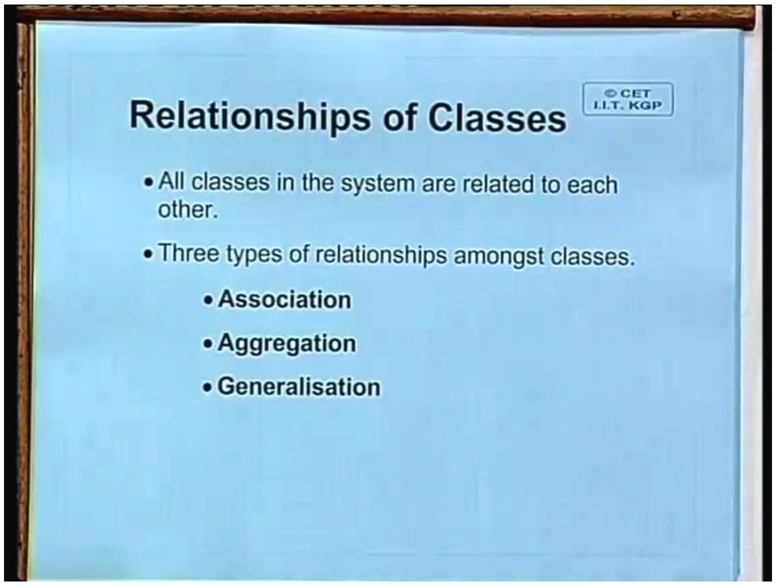
```
public class Parallelogram      {
    private int width=1;
    private int height=1;
    private int  angle =45;
    public int calculateArea()      {
        return width * height;
    }
    public void set(int w, int h, int a)        {
            width=w;
            height=h;
            angle = a;
    }
}
```

The...actually, when the operations are public, we show a plus sign here. This plus means these operations have a public scope. Then, public int calculate area, return width into height. See, it is a very surprising thing. We do not require the angle to calculate the area of a parallelogram; just width and height should be enough - return width into height. Then public void set int w, int h, int a, width equal to w, height equal to h, angle equal to a, right?

So, you can set by using the set command; using the set operation, you can set any... See, this one... 145 has been initialized, has been initialized. But we can use this way to set it to any value, but please understand - our purpose is not to teach you Java program here. I just took out this Java program with the basic idea, if it makes your concept little better, because it should also be clear, more clearer. If you...I mean, more clear. If you also, side by side, look at the implementation to some extent that is the basic idea of the classes and the objects.

Now let us see how the relationships between the classes are implemented, right? The relationship of the classes. So, the...all classes in the system are related to one another. The classes are actually related to one another and the relationships amongst classes are basically of 3 types, 3 types: The first one is called association, the second one is called aggregation and third one is called generalization.

(Refer Slide Time: 38:27)



Relationships of Classes

© CET I.I.T. KGP

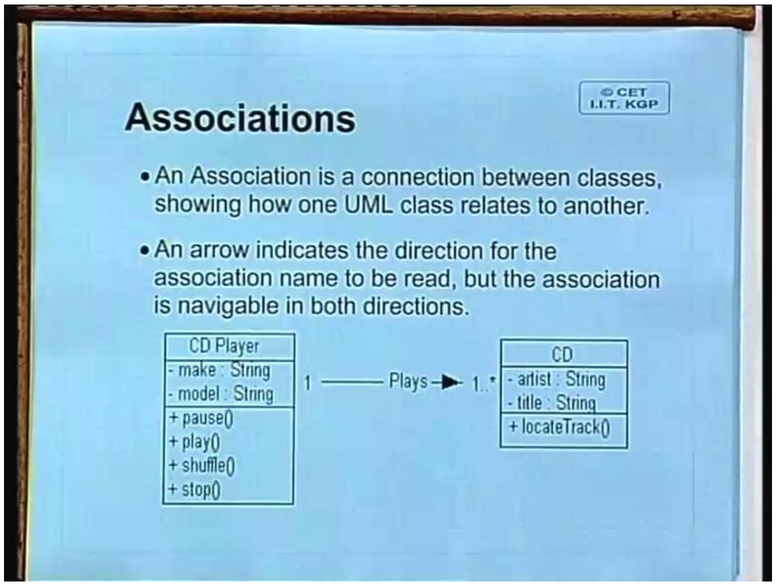
- All classes in the system are related to each other.
- Three types of relationships amongst classes.
  - **Association**
  - **Aggregation**
  - **Generalisation**

So association, aggregation and generalization. The association is basically when two classes are associated to one another, in the sense of an entity relationship diagram. The aggregation is essentially the idea of the whole part relationship, whole part, whole part. What is the whole part relationship? For example, the...let us say a particular assembly, alright? So in manufacturing assembly, the assembly will...how the assembly will be constituted? Probably it constitutes some sort of a plate with holes, then some bolts, some nuts, alright? So, we have the plate, we have the bolt, we have the nut - all are part of the assembly. So, we may say the plate-assembly constitutes of the plate - one plate, two bolts, nut. So, this is like a whole part relationship; the whole is a plate-assemble, parts are part 1, part 2, bolt, nut. Is it clear? That is the whole, a whole part. That is usually called aggregation.

On the other hand, the generalization - the basic idea is that...see, we can have different kinds of students, different kinds of students. For example, we can have research students, we can have B-tech students, we can have M-tech students - all are basically students. So, when I call a student, all are students. A Ph. D student is also a student, B-tech student is also a student, M-tech student is also a student, alright? But the basic idea of the student as such is a generalization of all these different kinds of students, alright? So, every student will have some common property - like they have a roll number, they will have a name, they will have an hostel attached room, alright?

So, these are the basic properties; but there will be certain attributes of a B-tech student will not be there in an M-tech student because B-tech students are here for longer time, right? They will be having certain specific things. Similarly, probably a Ph. D student will be having certain other attributes not usually found in B-tech or M-tech students. So, is it clear? So, that is the relationship of generalization. In fact, one very basic generalization relationship is a ISA relationship.

(Refer Slide Time: 41:53)



Associations

© CET I.I.T. KGP

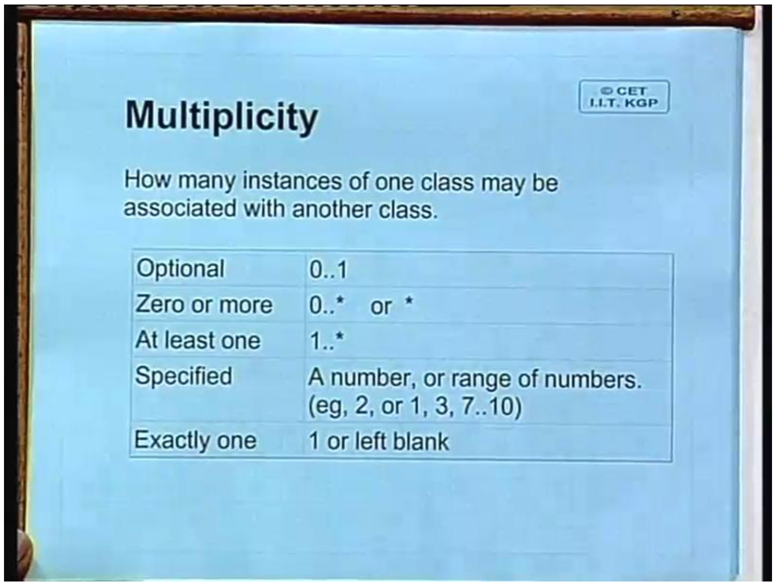
- An Association is a connection between classes, showing how one UML class relates to another.
- An arrow indicates the direction for the association name to be read, but the association is navigable in both directions.

CD Player
- make : String
- model : String
+ pause()
+ play()
+ shuffle()
+ stop()
1
Plays
1..*
CD
- artist : String
- title : String
+ locateTrack()

We have discussed while discussing your entity relationship diagrams. Say, we have the pilots and the crew, crew members. Both the pilots and the crew members, both are employees. All are employees. So, it is a kind of generalization. So, we shall see how these relationships are implemented in object oriented concepts.

Now first thing - that is the association. An association is a connection between classes showing how one UML class relates to another. See, we are using UML here because everything has been used are shown in terms of UML and the arrow; and see, when it comes to the arrow, arrow has got no specific meaning. This arrow is not the one. This to one, I told any entity relationship diagram that we draw, an arrow on the one side, it is not that way. This arrow simply means the name, how the name is to be read. See, the CD player plays the CD; it is not the CD does not play the CD player, alright? Since it is the CD player who plays the CD, we put the arrow this way. Suppose you could have written it in the other way around also.

(Refer Slide Time: 43:34)



Multiplicity

© CET I.I.T. KGP

How many instances of one class may be associated with another class.

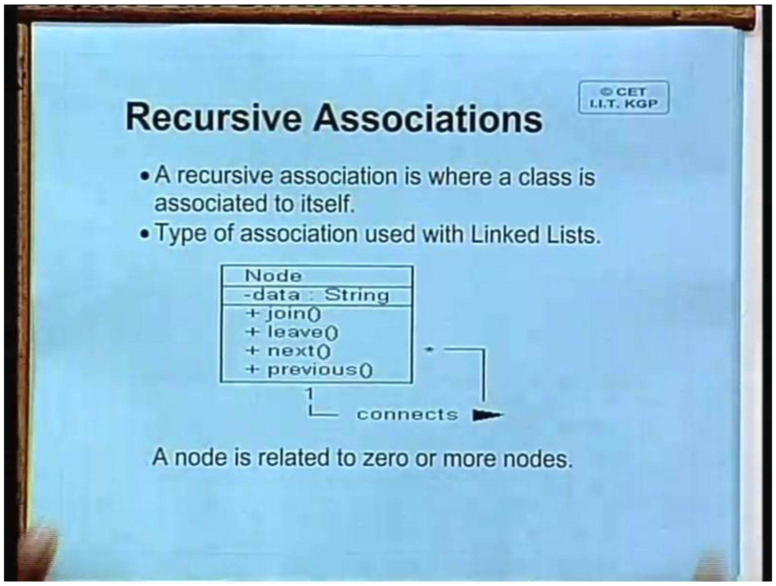
Optional	0..1
Zero or more	0..* or *
At least one	1..*
Specified	A number, or range of numbers. (eg, 2, or 1, 3, 7..10)
Exactly one	1 or left blank

So you may say, CD is being played by the CD player. So, if you name...its association name as ISA is played by, then arrow would have been the other way around, alright? So, do not put much weightage to this arrow. The arrow simply means how to read this association name; but what is more important is a multiplicity. See, I have told you, entity relationship diagram, we use 1 is to 1; 1 is to m and 1 is to n here. I will come back to this slide. The multiplicity is shown in a different way. These are the types or different types of multiplicity that are possible. The first one is an optional one. That means association is there, may not be there. So, it is either 0 or 1.

So it is shown as zero two dots and 1. It could be 0 or more. If it is 0 or more, then we say 0 dot dot star. Otherwise, simply star 0 dot dot star and simple star are one and the same, alright?

It could be at least 1. That is 1 dot dot star, at least one all right specified a number or range of numbers that is 2, 1, 3, 7, 10 suppose a given committee is constituted of 7 members. So whenever this committee will be formed steering committee the steering committee must have seven member suppose it is stipulated all right it is not any so steering committee to the member cannot be any number other than 7, right? The steering committee will have exactly 7 members, exactly 1. So, we can just write 1 or we can leave it blank, leave it blank. So, these are called, what is known as, multiplicity. So, when two objects... sorry, two classes are associated right. So, what is the multiplicity? So, let us come back to our previous slide between the CD player and the CD. The CD player plays the CD. One CD player can play number of CDs but at a time a CD can be played only on one CD, alright?

(Refer Slide Time: 46:17)


Recursive Associations

©CET I.I.T. KGP

- A recursive association is where a class is associated to itself.
- Type of association used with Linked Lists.

Node
-data : String
+ join()
+ leave()
+ next()
+ previous()
*
1
connects

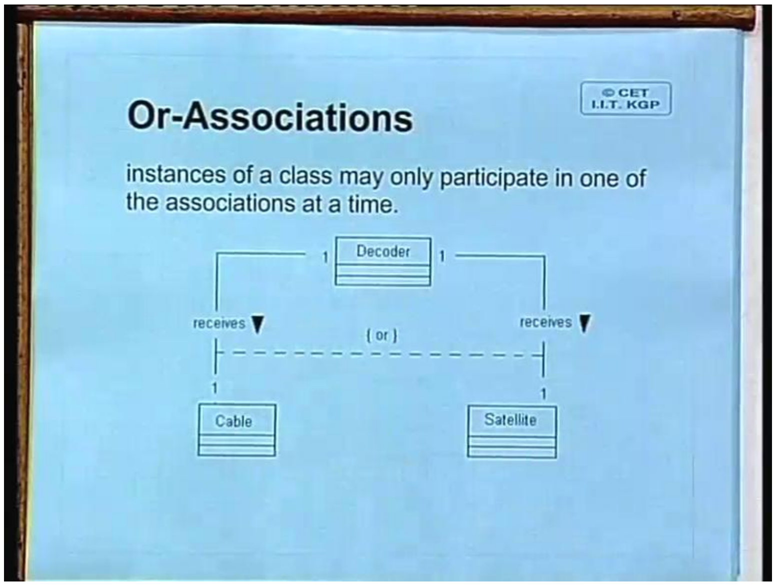
A node is related to zero or more nodes.

So, that is the idea. So, we have the CD player to number of CDs - 1 dot dot star... Multiple CDs can be played. Look at this - the CD player has got attributes as make and model and CD has artist and title; whereas, for the CD player we have pause, play, shuffle, stop and for the CD, locate, track, CD, locate, track, alright? So, these are the various operations that are actually possible between the CD player and the CD.

Then, we can have different kinds of associations, one such association is known as the recursive association, what is a recursive association where you connect to itself connect to itself. So recursive association is where a class is associated to itself. So it is a kind of self-association which is also allowed the type of association used with link list. So basically we have say a number of nodes.

So, all the nodes are available; but a given node can actually be connected to another node, is it not? Suppose you have 10 nodes. Suppose you define the ten pages of a book, alright? And say, HTML pages. So each HTML pages, you can define as one node as the different objects. Then, if you associate the objects to itself, what will happen? You can go from one page to the next page.

(Refer Slide Time: 48:09)



Or-Associations

© CET I.I.T. KGP

instances of a class may only participate in one of the associations at a time.

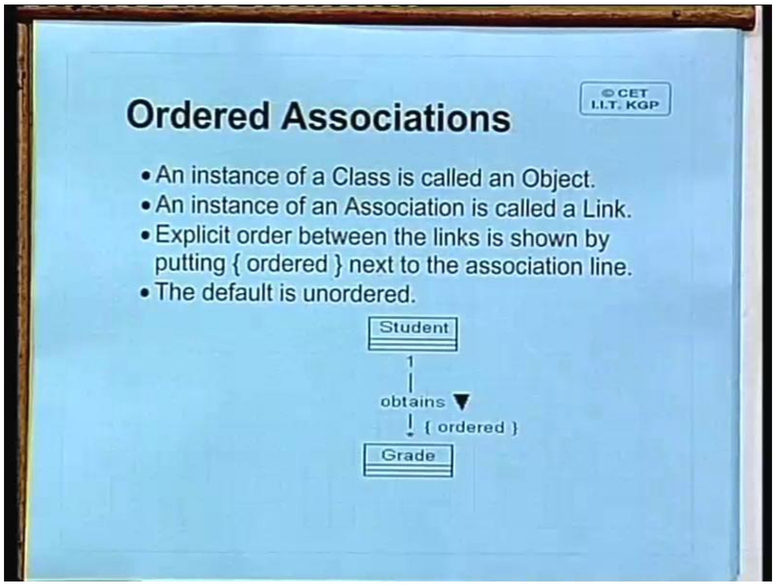
Decoder
1
1
receives ▼
receives ▼
{ or }
1
1
Cable
Satellite

Essentially the idea is, if you have instantiated this object, this particular node object, then what will happen? Suppose at a time some 10 objects are associated. So, you can...through a previous command, you go to the previous object. Through the next command, you go to the next page. So, may be a new operator could be given. So, this is called a recursive association, right? In a recursive association, a node is related to 0 or more nodes. Then we have what is known as the Or-Associations. We have the Or-Associations.

In an or-association, instances of a class may only participate in one of the associations at a time. In... so, say we have a decoder and we have cable and satellite. The decoder receives cable signal or it receives satellite signal, right? So, we have an or-association. So, or is written by a dotted line and or can be put here. Can you think of another or-association example?

Say for our internet, how does this internet come? Either from a landline or through a satellite line, alright? So, you can, you can define it as a satellite or the landline, right? The...basically cable; and you...what you are connecting the...we cannot call it the internet. We can call the signal or we can say the processing, alright? Internet facility, we can call internet facility; so, available through both these modes. Just one last slide for today. That is the ordered association.

(Refer Slide Time: 49:40)


Ordered Associations

© CET I.I.T. KGP

- An instance of a Class is called an Object.
- An instance of an Association is called a Link.
- Explicit order between the links is shown by putting { ordered } next to the association line.
- The default is unordered.

Student
1
obtains ▼
{ ordered }
Grade

So, an instance of a class is called an object; an instance of an association is called a link. So, you can also have an instance of an association. Explicit order between the links is shown by putting ‘ordered’ next to the associated line. The default is unordered. Say, a student obtains a grade, alright? A student obtains grade. So you see, you can have multiple students, you can have multiple students, definitely, and you can have multiple grades as well; but it is important to keep them ordered. Otherwise, you do not know which grade belongs to which student, because multiple grades are there to a student, multiple grades are there...the...suppose there are 4 subjects; the 4 subject grades are available, but which 4 grades are to a given student, alright?

So, if they have to be kept in an ordered manner, if you simply sort these grades, then everything will be everything will be haywire, alright? This is why the ordered associations are important, fine? So, we have discussed at least one type of relationship - that is, the associations. We still have other two to be discussed - aggregations and generalizations. These, we shall discuss in our next class.