

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

**Lecture – 21**  
**Relationships among objects (Contd.)**

Welcome back to module 12 of object oriented analysis and design. We are discussing relationships among object and in the earlier part, we have discussed the linked relationship between the object and talked about the links to for message flow. We have discussed about different rules that objects can have, controller, server and proxy. We have discussed about visibility and three kinds of mutual exclusion control for synchronization sequential control, guarded control and concurrent control.

(Refer Slide Time: 01:10)

**Aggregation**

Module 12  
Partha Pratim Das

Objectives & Outline  
Relationships / Associations  
Links  
Message Flow  
Rules  
Synchronization  
**Aggregation**  
Controller  
Server  
Proxy  
Summary

- Links denote peer-to-peer or client/server relationships
- In contrast, **Aggregation** denotes a whole/part hierarchy, with the ability to navigate from the whole (or aggregate) to its parts, that is, contained objects
- Aggregation may or may not denote physical containment
  - **HAS A (Strong Aggregation)**: Physical containment is necessary – Airplane & its Engine, Car & its Wheels, Book & its Chapters, Chapter & its Sections, etc.
  - **HAS (Weak Aggregation)**: Conceptual aggregation, Physical containment is not present – Shareholders & Shares, Library & Users, etc.

An Aggregation is a specialized form of Association (Link)

- **Aggregation** is sometimes better because it encapsulates parts as sections of a whole
- **Links** are sometimes better because they permit looser coupling and

NPTEL MOOCs Object Oriented Analysis and Design      Partha Pratim Das

Next we will talk about the relationship of aggregation. So aggregation denotes basically denotes a whole part hierarchy. Some kind of a whole/part hierarchy which gives us the ability to navigate from the whole or the aggregate to its parts that is the content objects. So the aggregation is a relationship where one object necessary in some way is a collection of other objects or one object is a some way a component of other objects and so on.

But the interesting point for in aggregation or the relationship with aggregation is that it may or may not be denote physical containment. What does it say that aggregation basically can be of 2 types? One is where the physical containment is necessary. I talk about an airplane object, I talk about its engine object naturally the airplane has a physical containment of the engine. Car has wheel object, book has chapters, chapters has sections, sections have paragraphs, paragraphs have sentences, sentences have

words so and so forth.

So these are all physical containment and the other could be conceptual aggregation of the kind that I talked about in the last lecture of this module is where the physical containment is not present but there is a conceptual containment, conceptual aggregation in the sense that the relationship between shareholders and the share. Both are objects, they are related but share is not a part of the shareholder right and vice versa. Library has users but users are not part of the library. They are not components of the library.

But when you talk about library and books then there is a physical containment. So both of these are known as aggregation and since physical containment is a stronger sense so an alternate name for physical aggregation is strong aggregation and the conceptual one is more commonly referred to as weak aggregation. You can realize that we have several times talked about has-a relationship. So strong aggregation necessarily has a relationship and many of us refer to the weak aggregation as simple has relationship.

So now we have aggregation and links so if you ask us to what should be the way you should model that is should you model in terms of the strong one or the weak one whether containment is better, physical containment is better, whether just associations are better and so on. The answer is both of them exist because there are different situations for their use. It is in some cases aggregation is better because it keeps the objects within itself as physically contained part.

So there by the principles of encapsulation they are encapsulated secrets of the object that is all of us have seen aircrafts but we have not seen their engine but links may be better in some other cases because links do not physically contain an encapsulate. So they provided more flexible loose coupling between objects. So this is the basic notion of aggregation to work with.

(Refer Slide Time: 04:52)

## A Temperature Controller System

**Module 12**

Partha Pratim Das

Objectives & Outline

Relationships / Associations

UML

Message Flow

Rules

Validation

Synchronization

Aggregation

Example

**Temperature Controller**

LMS

Summary

```

classDiagram
    class TemperatureController {
        -heater
        +schedule()
        +process()
    }
    class TemperatureRamp {
        +interpolate()
    }
    class Heater
    TemperatureController --> TemperatureRamp : regulates temperature using
    Heater o-- TemperatureController
    
```

- The temperature control system has two primary objects **TemperatureController** and **TemperatureRamp**
- The link between the two objects denote message flow
- The **Heater** object is a component of **TemperatureController**, hence **TemperatureController** is the whole and **Heater** is the sub part
- The whole component **TemperatureController** will have a link to aggregate **Heater**, to send or receive messages

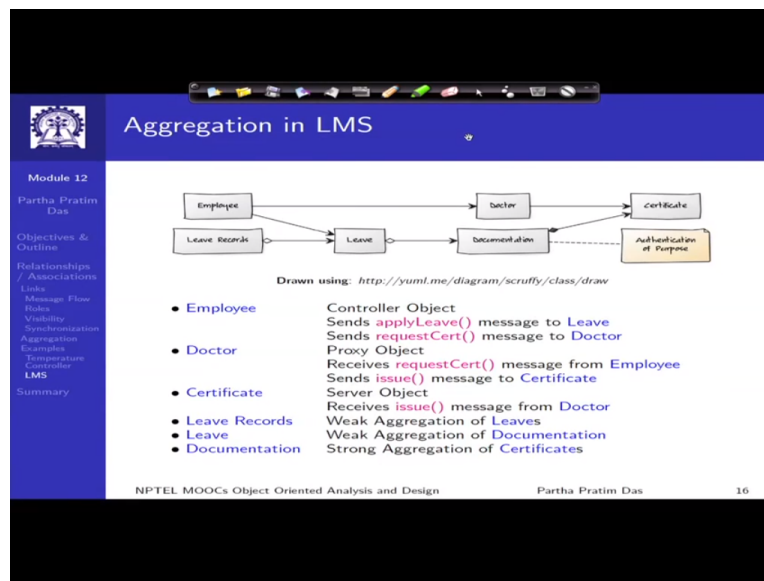
NPTEL MOOCs Object Oriented Analysis and Design
Partha Pratim Das

So we will just take a look into a couple of simple examples. Here is one, example of a temperature controller which has three objects certainly this one; the first one is the controller which decides to maintain a certain temperature profile for an industrial process. The temperature profile that needs to be maintained use an object itself which we say is a temperature ramp that is how the temperature should increase, how should it decrease and so on.

And certainly to actuate this whole process, we have a heater which can be turned high, turned low and so on. So if we look into these, then here we see a link where the temperature controller sends message to the ramp saying that this is the kind of profile that you need to schedule, this is a time when the temperature should start increasing, this much it should increase and so on so those information can be sent in terms of request from the temperature controller to the temperature ramp.

But if you look at the heater, heater necessarily is a part of the temperature controller; it is a part of the temperature controller. So there is an aggregation in terms of this heater which exists in the temperature controller system. So this is we just took we are just taking very simple examples to illustrate what different relationships could be in a real system.

(Refer Slide Time: 06:32)



Let us look at look back at our leave management system so this is a simple, a partial model of the leave management system so you are already aware of many of these objects like certainly I will have leave management system so I will have leave object. The leave is for the employees so we have employee object and am just depicting a situation where as you have studied the there are number of leaves like the sick leave and maternity leave and so on where we need separate documentation.

We need certificates from doctor, we need medical reports and so on for that leave to be approved or for that leave to be placed in the system. So if we look into this that part of the LMS system only a part of that which has some of the objects then we expect that there will be a there will be certainly an employee object. Now certainly the employee in this case will decide whether he or she wants to apply for leave or if the employee is an, is a lead or a manager then he or she would decide whether some leave should be approved or not.

Which means in essence it means that these employee object is kind of the driver in the system? So we will immediately say that this is a controller object. Now it sends message along this link, it sends messages along this link so the message along the employee to leave link could be apply leave message where an executive or a lead is applying for a leave. This link from employee to a doctor could be request certificate link that me employee has not been keeping good health.

So goes to the doctor and request for a certificate. If you look at the certificate, the doctor will issue the certificate in this link based on the status of the employee, the health status of the employee I mean. So if you look at the certificate object here then the certificate object receives a request form the doctor

that a certain type of certificate will have to be issued. So this will have to be printed or electronically signed and so on. So it just serves that request, so we say it is a server object which receives the issue message from the doctor and serves it.

So what is the doctor object? The doctor object is on one side, it receives the request certificate, the one that we say here from the employee which comes on this link and on the other side it will send a request for issue to the certificate object on the other link. So it is in both so we identify that the doctor object is a proxy object in this situation. Now if we look into the leaf so certainly an employee has applied for a leave that is not the only leave object in the system, surely there will be several other requests from the employees.

So we can conceive of a collection of leaves that exist in the system and we say this is a leave record. So the leave record will not contain will not encapsulate all the leaves, rather it will be a list with appropriate pointers or references to different leave objects much in the way that a shareholder talks about his share, I talk about my bank account in a similar manner. So you can say leave records as a weak aggregation of the leaves and you can see that in terms of this diagram,

We are showing this aggregation with a certain type of drawing where there is a diamond at the aggregate end which is empty and this emptiness represents weak aggregation. So now if we look at the leave itself then we know that the leave in this case, we need to have the documentation, the documentation of the doctors certificate, the documentation of the different medical reports, may be the x-ray report, the ECG report and so on so forth. So the leave has weak aggregation of documentation.

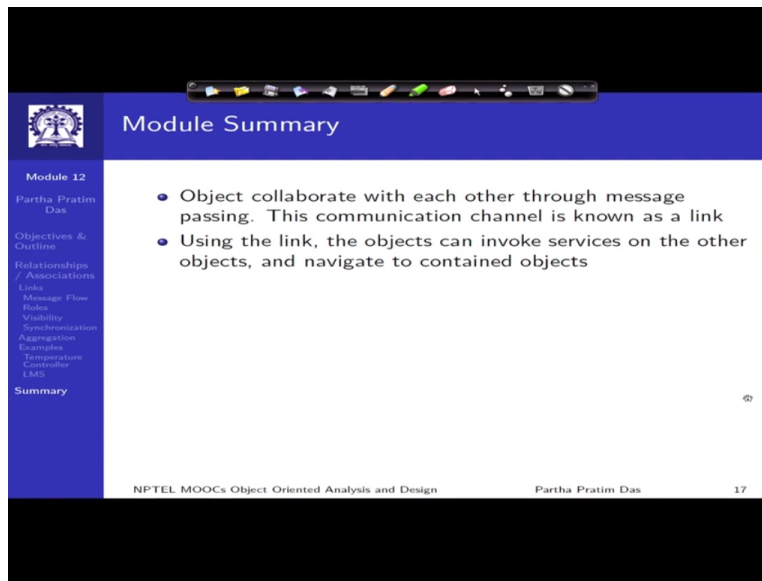
Now leave will not contain the documentation, it is not a part of that but leave will make references to the documentation so this is another situation of weak aggregation as depicted here. Now let us look at the documentation object itself. The documentation is the bundle of the certificate and the different reports. So the documentation is has certificate as a component object because your documentation file which in terms of if you think in terms of the physical situation then it is a documentation file which has the papers showing the doctors certificate, papers showing the medical report and so on.

So all of that the reports and all those are not showing here. I just shown the certificate becomes a strong aggregation or a strong or a component of the documentation therefore if you just look at the diamond here this diamond is a filled up one, unlike these diamonds which are empty ones depicting

weak aggregation here. It is a filled up diamond depicting strong aggregation. So this was just referring back to our leave management system to show that the concepts of relationship amongst objects

That we have just understood in this module can be found out found plenty in terms of a leave management system. We are just taken a look into one part of the system you can I would encourage you to look into the whole specification and listen to the ehh tutorial that we have put up on the leave management system and identify more and more object relationships in the leave management system.

(Refer Slide Time: 12:54)



The screenshot shows a presentation slide titled "Module Summary". At the top, there is a navigation bar with a logo on the left and a series of icons in the center. Below the navigation bar, the slide is divided into two main sections. On the left is a vertical sidebar with a blue background and white text. It lists the following items: "Module 12", "Partha Pratim Das", "Objectives & Outline", "Relationships / Associations", "Links", "Message Flow", "Roles", "Visibility", "Encapsulation", "Aggregation", "Examples", "Temperature Controller", "UML", and "Summary". The "Summary" item is highlighted. On the right is the main content area with a white background. It contains two bullet points: "• Object collaborate with each other through message passing. This communication channel is known as a link" and "• Using the link, the objects can invoke services on the other objects, and navigate to contained objects". At the bottom of the slide, there is a footer with the text "NPTEL MOOCs Object Oriented Analysis and Design" on the left, "Partha Pratim Das" in the center, and "17" on the right.

Module Summary

- Object collaborate with each other through message passing. This communication channel is known as a link
- Using the link, the objects can invoke services on the other objects, and navigate to contained objects

NPTEL MOOCs Object Oriented Analysis and Design Partha Pratim Das 17

So in this module, we have discussed to summarize, in this module we have discussed the object relationship is how does the objects relate between themselves, how do they collaborate with each other through messages and we have noted that such communication links between object such communication channels between objects is known as links and using the link the object can provide services or invoke services or the navigate from one object to another.