

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

**Lecture – 06**  
**Object-Oriented Analysis and Design**

Welcome to module 4 of object-oriented analysis and design. We have so far taken a look into the difficulties of software development and what are the, look at some of the complexities, sources of some of the complexities of software development and then in order to form adequate tools, instruments to be able to handle this complexity. In the last module, module 3, we have studied the structure of couple of sample systems and extracted from the study of those sample systems.

We have extracted 5 basic attributes of complex systems. We have understood that the complex systems necessarily has to be in hierarchical nature there has to be a definition of primitives, there will be separation of concern between its components, there will be certain common patterns which we will find across systems and we will need to define some stable intermediate forms for the development.

(Refer Slide Time: 01:42)

The screenshot shows a video player interface. At the top, the video title is 'mod02lec06'. Below it, a blue header bar contains the text 'Module Objectives'. The main content area is white and displays the following:

- Module 04
- Partha Pratim Das
- Objectives & Outline
- Handling Complexity
- Canonical Form
- Decomposition
- Hierarchy
- Decomposition
- State and Object Structures
- Canonical Form
- Human Capacity
- Summary

The main objective listed is: 

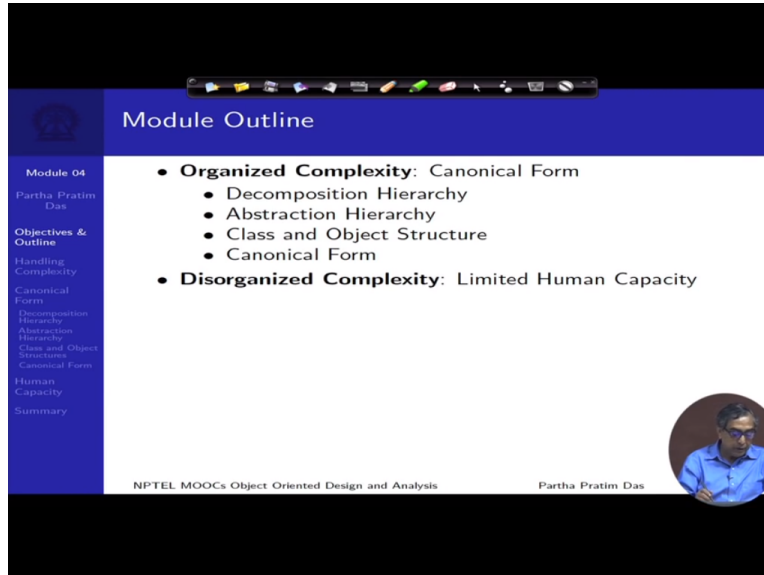
- Understand how to handle complexity

At the bottom of the slide, it says 'NPTEL MOOCs Object Oriented Design and Analysis' and 'Partha Pratim Das'.

The video player controls at the bottom show a progress bar at 1:41 / 23:21, a play button, a volume icon, and a settings icon.

Equipped with that, now we try to get closer to the design problem, the analysis and design problem and look into how we handle the complexity better. So this module will primarily be concerned with the approaches to handle complexity

(Refer Slide Time: 01:44)



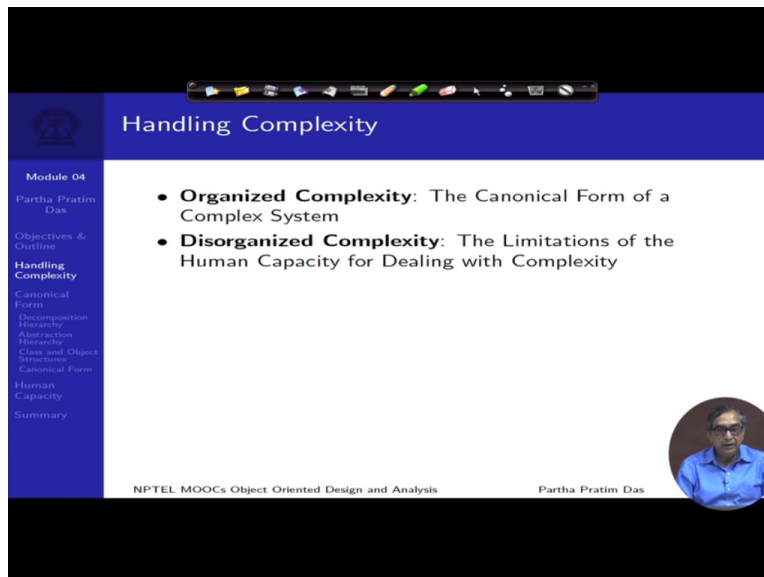
The slide is titled "Module Outline" and is part of "Module 04" by Partha Pratim Das. It lists the following topics in the left sidebar: Objectives & Outline, Handling Complexity, Canonical Form, Decomposition Hierarchy, Abstraction Hierarchy, Class and Object Structure, Canonical Form, Human Capacity, and Summary. The main content area lists two primary topics:

- **Organized Complexity:** Canonical Form
  - Decomposition Hierarchy
  - Abstraction Hierarchy
  - Class and Object Structure
  - Canonical Form
- **Disorganized Complexity:** Limited Human Capacity

The footer includes "NPTEL MOOCs Object Oriented Design and Analysis" and "Partha Pratim Das" with a small circular portrait of the speaker.

The module outline as it is in to the left of your slide.

(Refer Slide Time: 01:56)



The slide is titled "Handling Complexity" and is part of "Module 04" by Partha Pratim Das. It lists the following topics in the left sidebar: Objectives & Outline, Handling Complexity, Canonical Form, Decomposition Hierarchy, Abstraction Hierarchy, Class and Object Structure, Canonical Form, Human Capacity, and Summary. The main content area lists two primary topics:

- **Organized Complexity:** The Canonical Form of a Complex System
- **Disorganized Complexity:** The Limitations of the Human Capacity for Dealing with Complexity

The footer includes "NPTEL MOOCs Object Oriented Design and Analysis" and "Partha Pratim Das" with a small circular portrait of the speaker.

Now in terms of handling complexity we just divide the approach in 2 parts or define rather we discover that there are 2 parts to handling complexity. 1 is we can say is organized complexity rather this is kind of complexity where we can we are able to find structure, we are able to find

patterns, we are able to make device, tools and mechanisms by which we can approach the whole problem better.

So we will discuss those and leading to certain kind of a canonical form, standardized form for a complex system. So any complex system we want to design, we will try to express that in this canonical form and that is why it is called organized complexity. The other part of the complexity, unfortunately as it stands today is disorganized and that has to deal with us. A major or significant component or the most significant component in the whole process of analysis and design is a human game because it is a human process.

So unfortunately there is a severe limitation of the human capacity and that leads to a lot of complexity of the overall analysis and design problem. We cannot handle a lot of information together. So when we are thrown into a complex system for the first time, to put it in simple terms, we get lost. No individual can master and capture that. So we kind of struggle like a like a child thrown into the water and trying to learn to swim.

so this aspect of complexity is known as a or is characterized as a disorganized complexity because there is still a lot that needs to be developed in terms of techniques, tools and understanding before we can handle this more and this is one of the major source of issues that will continue to progress.

(Refer Slide Time: 03:50)

**Hierarchies of a Complex System**

- Hierarchies are of multiple types
  - Decomposition – **part-of** or **HAS-A**
  - Abstraction – **IS-A**

Module 04  
Partha Pratim Das  
Objectives & Outline  
Handling Complexity  
Canonical Form  
Decomposition Hierarchy  
Abstraction Hierarchy  
Class and Object Structures  
Canonical Form  
Human Capacity  
Summary

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

So the first let us take a look into the organized complexity we have seen that the complex systems are hierarchic now we go just one step further to say that hierarchy is not 1. there are multiple different hierarchies that are possible in a system and of this the 2 hierarchies which are very significantly impact are analysis and design process are hierarchies of decomposition which is commonly called the part-of or has-a hierarchy and hierarchies of extraction.

(Refer Slide Time: 04:24)

**Decomposition (HAS-A) Hierarchy**

- Personal Computer is composed of
  - CPU
  - Memory
  - Keyboard
  - HDD
  - ...
- This is **Decomposition Hierarchy**

Module 04  
Partha Pratim Das  
Objectives & Outline  
Handling Complexity  
Canonical Form  
Decomposition Hierarchy  
Abstraction Hierarchy  
Class and Object Structures  
Canonical Form  
Human Capacity  
Summary

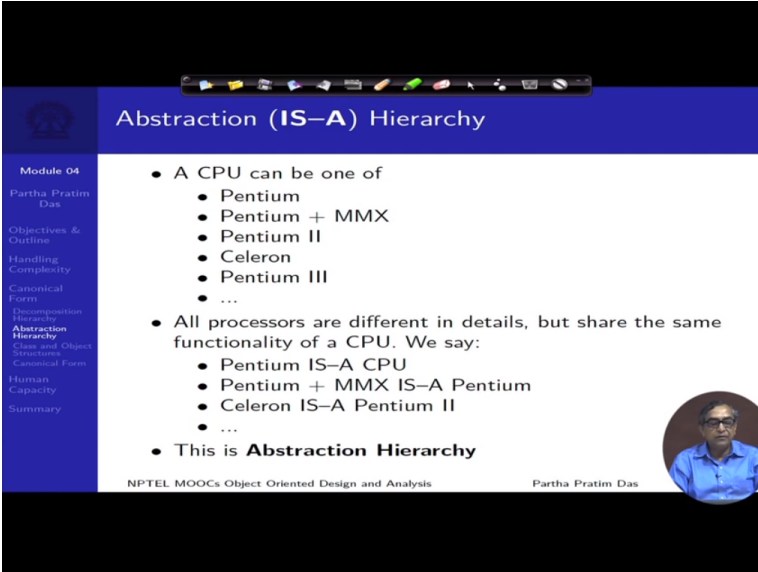
NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

Let us take a look. So we did analyze the structure of a personal computer. The personal computer is composed of CPU, memory, keyboard, hard disk and naturally we identify that this is the hierarchy. so this hierarchy is kind of a compositional hierarchy so if you look at from the

top, we call this as a decomposition hierarchy that is if I break down cpu I will get this component, am, if I break down a personal computer, I will get these components.

If I break down a CPU, ALU, I will get primary memory I will get parsers. If I break down an alu, I will get resistors and so on. So such kind of hierarchy is called as decomposition or has-a hierarchy. has-a in the sense that we can just in terms of simple English phrase I can say that personal computer has a cpu, personal computer has a memory, cpu has a alu so a short term form of decomposition hierarchy is has-a hierarchy.

(Refer Slide Time: 05:27)



The slide is titled "Abstraction (IS-A) Hierarchy". It features a blue header with a navigation bar at the top. On the left side, there is a vertical menu with the following items: "Module 04", "Partha Pratim Das", "Objectives & Outline", "Handling Complexity", "Canonical Form", "Decomposition Hierarchy", "Abstraction Hierarchy", "Class and Object", "Summary", "Human Capacity", and "Summary". The main content area is white and contains a bulleted list:

- A CPU can be one of
  - Pentium
  - Pentium + MMX
  - Pentium II
  - Celeron
  - Pentium III
  - ...
- All processors are different in details, but share the same functionality of a CPU. We say:
  - Pentium IS-A CPU
  - Pentium + MMX IS-A Pentium
  - Celeron IS-A Pentium II
  - ...
- This is **Abstraction Hierarchy**

At the bottom of the slide, there is a small circular portrait of Partha Pratim Das on the right, and the text "NPTEL MOOCs Object Oriented Design and Analysis" and "Partha Pratim Das" on the left.

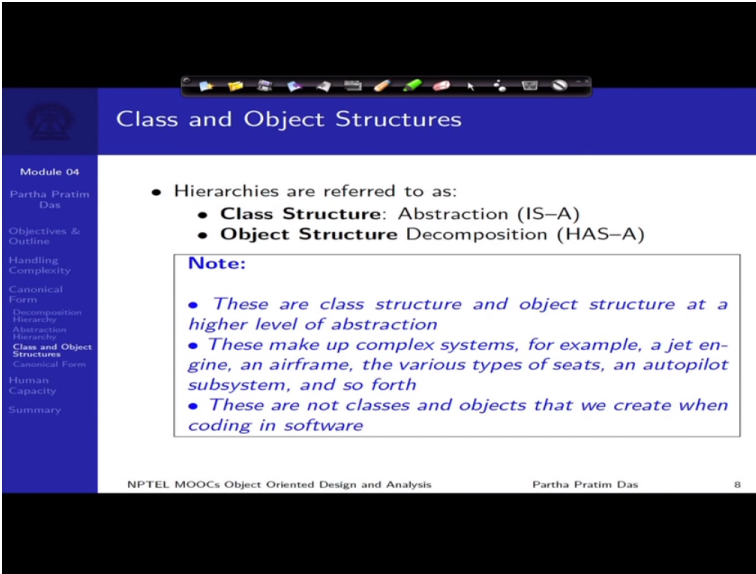
In contrast, let's say if I look at the CPU then I just say that CPU, I mean CPU has what is the functionality can execute programs. It will have some kind of an alu memory, bars and all that but we don't say what CPU we have. We all know that there are different variations of CPU so it could be a Pentium, it could be a Pentium MMX Pentium II, Celeron, Pentium III so and so forth it could be. So this all these different kinds of alpha CPU and all of these different ones.

Now when we have talked about a CPU, we did not say which specific one is this. So all these processors are different in their details but they share the same functionality, the functionality of that of being able to execute a program, access the memory, work through the buses and so on so we can easily say that the Pentium is a CPU that it is an instance of, it is a kind of the CPU. But then we have Pentium MMX which is Pentium with multimedia functionality.

So we will say that Pentium MMX is a Pentium but then it has something more. Will say Celeron is a Pentium II because it whatever Celeron can do, whatever Pentium II can do Celeron can also do that. So we are saying here that comp notion of hierarchy is little bit different. We say this is kind of an easy hierarchy that is 1 concept say pent CPU has different kinds of variations all of which are actually CPU. One concept is Pentium has different variations but all of them share the same x86 architecture.

So this kind of a hierarchy of which we will talk lot more as we go along is known as an abstraction hierarchy or as you have seen that expressions use the simple English phrase of is a. so commonly it is called a is-a hierarchy.

(Refer Slide Time: 07:34)



The slide is titled "Class and Object Structures". On the left is a blue navigation menu with the following items: Module 04, Partha Pratim Das, Objectives & Outline, Handling Complexity, Canonical Form, Decomposition Hierarchy, Abstraction Hierarchy, **Class and Object Structures**, Canonical Form, Human Capacity, and Summary. The main content area has a blue header with the title. Below the header, there are two bullet points: "Hierarchies are referred to as:" followed by "Class Structure: Abstraction (IS-A)" and "Object Structure Decomposition (HAS-A)". Below these is a box labeled "Note:" containing three bullet points: "These are class structure and object structure at a higher level of abstraction", "These make up complex systems, for example, a jet engine, an airframe, the various types of seats, an autopilot subsystem, and so forth", and "These are not classes and objects that we create when coding in software". At the bottom of the slide, it says "NPTEL MOOCs Object Oriented Design and Analysis", "Partha Pratim Das", and the number "8".

- Hierarchies are referred to as:
  - **Class Structure:** Abstraction (IS-A)
  - **Object Structure** Decomposition (HAS-A)

**Note:**

- *These are class structure and object structure at a higher level of abstraction*
- *These make up complex systems, for example, a jet engine, an airframe, the various types of seats, an autopilot subsystem, and so forth*
- *These are not classes and objects that we create when coding in software*

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 8

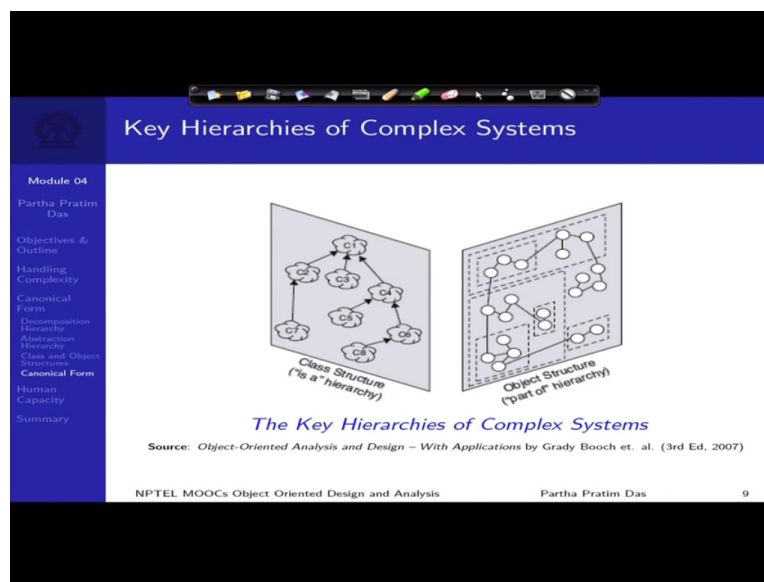
Now these 2 hierarchies are typically referred in terms of 2 structures, known as a class structure and object structure. The class structure gives us the abstraction hierarchy and the object structure gives us the decomposition hierarchy. We will show why it is this way so all that we are saying that kind of the complex system representation has kind of 2 orthogonal access 1 which talks of the abstraction hierarchy the other which talks about decomposition hierarchy.

The one which talks about the abstraction hierarchy we say is defines a class structure, one that talks about the decomposition hierarchy we say talks about the object structure. here please note

a couple of points these are if you are new to, completely new to object oriented analysis, design and programming particularly then you may ignore this note because you will not have this confusion but if you are if you have some background in op oad

Then you will start confusing this use of the term class and object with what you know in say in programming language class in c++, java object doesn't stands and so on. But here we are not talking about those coding details. Here we are talking about high level concepts that are represented in terms of class and object structures.

(Refer Slide Time: 09:10)



So this the very nice diagram from the text book that I have recommended to you from the booch's text book and you can see very clearly that here the 2 different axis are being drawn. This is the axis of abstraction and this is the axis of decomposition. So I can look at but we are looking at the some system if you look at the some system along this axis, then what you are saying is there is something like class c1. There is something concept which is the CPU.

(Refer Slide Time: 09:34)

**Key Hierarchies of Complex Systems**

Module 04  
Partha Pratim Das

Objectives & Outline  
Handling Complexity  
Canonical Form  
Decomposition Hierarchy  
Abstraction Hierarchy  
Class and Object Structures  
Canonical Form  
Human Capacity  
Summary

**The Key Hierarchies of Complex Systems**

Source: *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3)

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

Then there are some classes concepts which have speculations of that that is classes c2, c3, c4 are like c1 but something more. So if c1 is CPU, c2 could be Pentium, c3 could be the mac processor, c4 could be the (()) (10:15) and so on. Then we seen several which is c2 these arrow basically its IS-A. So c7 is the c2, so the Celeron is a Pentium.

(Refer Slide Time: 10:42)

**Key Hierarchies of Complex Systems**

Module 04  
Partha Pratim Das

Objectives & Outline  
Handling Complexity  
Canonical Form  
Decomposition Hierarchy  
Abstraction Hierarchy  
Class and Object Structures  
Canonical Form  
Human Capacity  
Summary

**The Key Hierarchies of Complex Systems**

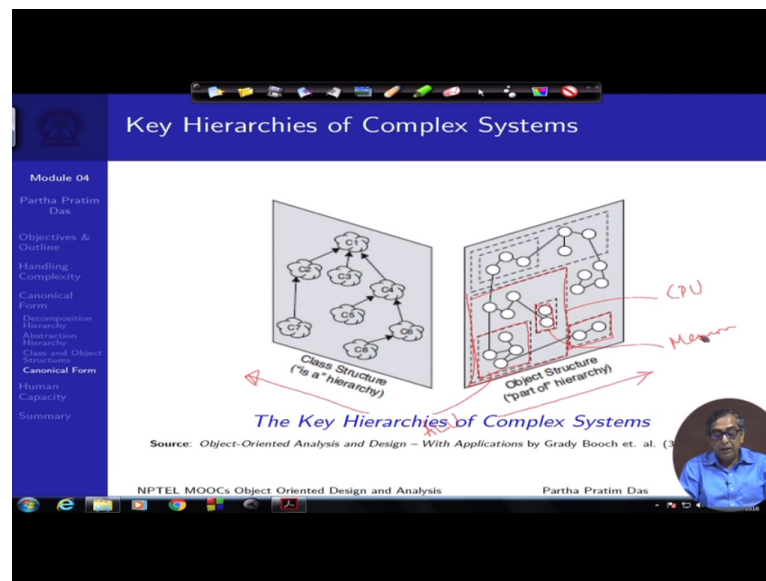
Source: *Object-Oriented Analysis and Design – With Applications* by Grady Booch et. al. (3)

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

So we have c so you have CPU here, we have Pentium we have mac, we have alpha, we have Celeron. So in terms of a personal computing system, say that these are different special kinds of cpu that are possible and that is what gives rise to the class structure which says that Celeron has all the properties that Pentium has which in turn has all the properties that the cpu need to have. Independently alpha has all the properties that CPU need to have.



(Refer Slide Time: 12:25)



And when you look into the whole system in terms of this commonality of behavior, we say we are looking through the class structure that is the abstraction hierarchy. On the other if you look into this, you'll find that there are dotted boxes enclosing this it says basically this is this is one object; this is one entity we are dealing with. So this is another entity. So this if we look into this object or this entity then you have this, you have another.

So this is exactly a kind of decomposition that we are talking about so if this happens to be cpu, then this could be your alu, this could be your memory and so on. So this is basically the decompositional hierarchy known as the part of hierarchy or the object structure so you can look at the whole system in terms of the different objects different component values it has and what is the sub component hierarchy it has. So these are the 2 key hierarchies through which complex systems of today will be characterized will be discussed.

(Refer Slide Time: 12:49)

The screenshot shows a presentation slide with a blue header and a white content area. The header contains the title 'The Canonical Form of a Complex System' and a small logo. The left sidebar lists the module contents, with 'Canonical Form' highlighted. The main content area lists the topics covered in the module, including System Architecture, Five Attributes of a Complex System, and a concluding statement about the canonical form.

## The Canonical Form of a Complex System

- Module 04
  - Partha Pratim Das
  - Objectives & Outline
  - Handling Complexity
  - Canonical Form
  - Decomposition Hierarchy
  - Abstraction Hierarchy
  - Class and Object Structures
  - Canonical Form
  - Human Capacity
  - Summary

- System Architecture
  - **Class Structure:** Abstraction (IS-A)
  - **Object Structure** Decomposition (HAS-A)
- Five Attributes of a Complex System
  - Hierarchic Structure
  - Relative Primitives (multiple levels of abstraction)
  - Separation of Concerns
  - Common Patterns
  - Stable Intermediate Forms
- This is **Organized Complexity**

**Virtually, all complex systems take on the above canonical form**

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 10

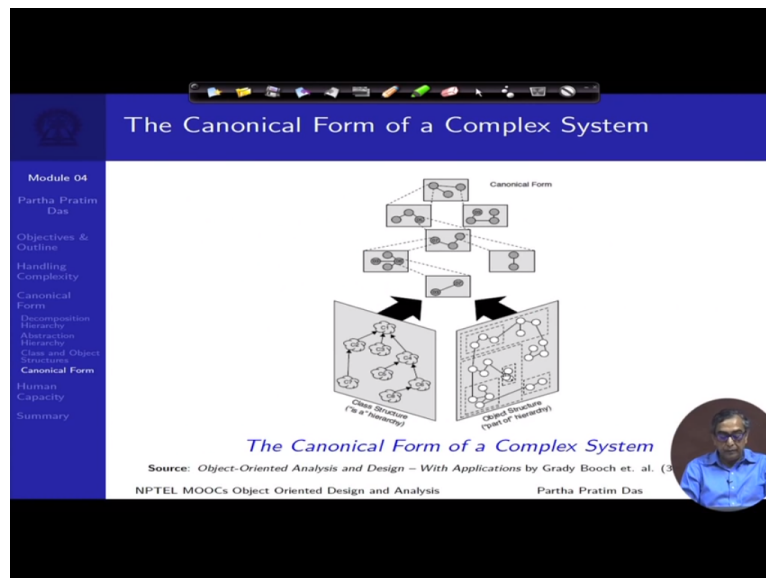
So with this, now we say the canonical form of a complex system, we can now describe that in terms of 2 major properties, 1 what we say is the system architecture that is the 2 orthogonal view of how does the class structure, how does the abstraction hierarchy look in that system and what is the object structure that is how does the decomposition hierarchies look in that system and these 2 together is commonly referred to as a system architecture for the complex system.

The other is what we have studied in the previous module, the 5 different attributes of the complex system and in terms of these attributes, we need to analyze the complex system and represent that so when we do all these together, the 5 attributes in the system architecture then we say that we have placed the complex system in a canonical form. So here the the main observation to be made the main advantage we are trying to drive in is here we are not talking about any specific complex system.

No matter which complex system we talk about, we will be able to represent it in terms of these architecture and attributes and that is why this is called a canonical or standardized form of representation and once this representation can be made complete we will have much better understanding of the complex system. So in this process we have found that the overall complexity of a complex system in terms of understanding it, analyzing it dealing with it.

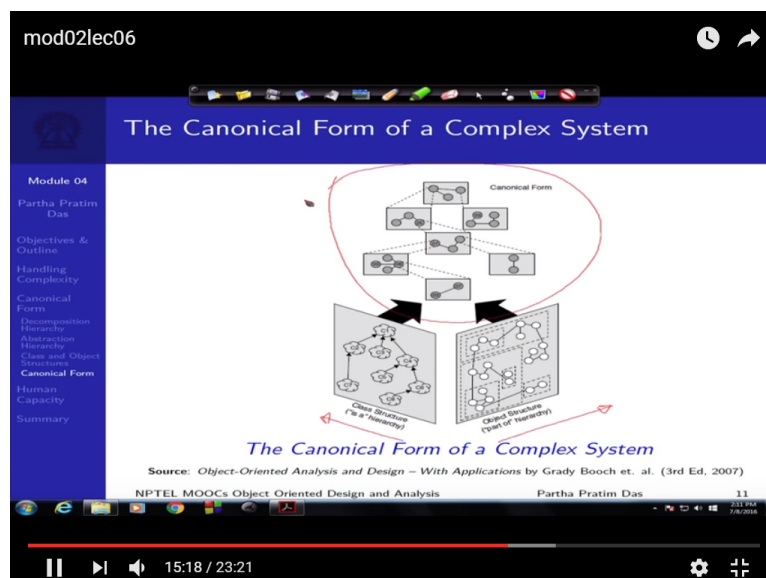
We have been able to somehow organize it under the canonical form and that is the reason we call this an organized complexity of the whole system.

(Refer Slide Time: 14:48)



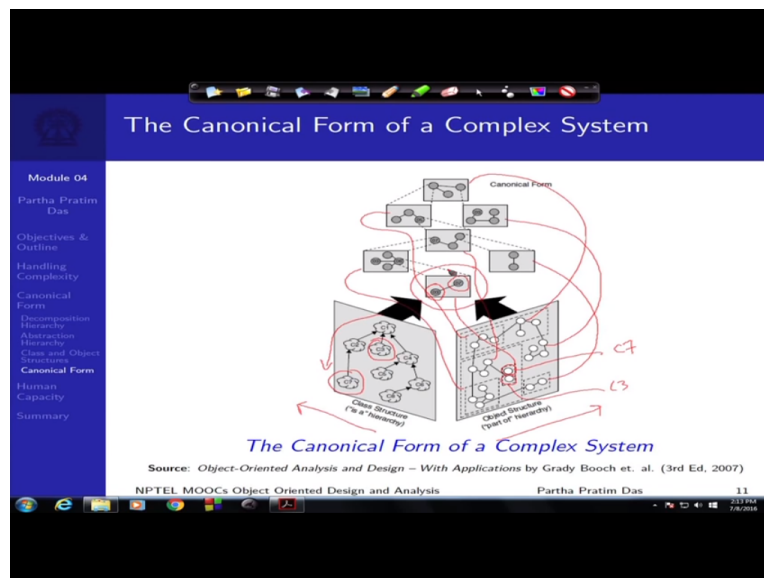
So this is an illustration of the canonical form with this is again from the same text book so if you if you look into this what it means, this is the class hierarchy, class structure that is the abstraction hierarchy, this is the decomposition hierarchy or the object structure and if you look at what these are these are actually the objects. So this part which is which is being added new this part the canonical form part which is actually being added with those attribute is basically a cross product or combination of the these 2 axis.

(Refer Slide Time: 15:20)



So if you look at you have an object here and I think this is this object. The object here is this object; the object here is this object. You can just verify, the object here is this object, the object here is this object, the object here is this object. This means is basically the same the object here is this object. This is basically the same object but what you can read with a little bit of care with some am sorry it is not that clear in terms of the size.

(Refer Slide Time: 15:20)



It is here it is written to be 3 which means that this is this object which in turn is this object is a c3 object from class state this is 7, so this is a class 7 object. So you can say that this is a so this is a c7 object. This is a c3 object. So this hierarchy tells you that how these objects are going to get their behavior from higher downwards and this hierarchy tells you which you represent here as well as here is how they are put together. How their component structure is?

And we will see when we go which of these approaches is into defining object models will see that these are what will start calling them as a basic abstraction or inheritance properties will start calling them as modularity, encapsulation and all those properties. But this is basic system structure that we need to keep in mind. So I would request that please carefully study that diagram. This is a very interesting illustration now.

How should a complex system canonically look like when analyzed from the 2 orthogonal axis of class structure or abstraction hierarchy which say what is the specialization amongst different

concepts in that system and the object structure or the decomposition hierarchy which says which is a component, sub component, sub-sub component relationship and as a intermix of these 2 canonical form emerge in the middle which is the system that we are actually trying to build up.

(Refer Slide Time: 18:00)

The Limitations of the Human Capacity for Dealing with Complexity

Module 04  
Partha Pratim Das

- Maximum number of chunks of information that an individual can simultaneously comprehend is on the order of seven, plus or minus two
- Human channel capacity is related to the capacity of short-term memory
- Processing speed is a limiting factor – it takes the mind about five seconds to accept a new chunk of information
- This leads to **Disorganized Complexity**

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

Now coming to the other side, so this was about the organized complexity. But finally the human beings will have to analyze and actually build up this canonical form from as we discuss mentioned earlier also from some vague descriptions from some vague interactions, half knowledge broken ideas and so on and when someone is thrown into a complex system, it is goes well beyond the comprehension of that person.

And here some of the observations that different psychologist and physiatrist have tried to quantify over a period of time. What is commonly said is a maximum number of chunks of information that is an if you if you deal with the simple information that we can simultaneously comprehend is kind of a 7 for a human individual. Now for some it could be little less for some it could be little more that is around that number. Now this chunk of information could be anything.

It could be independent concepts, it could be number of variables in a in a function, it could be the different components or different important things that you see in a picture and so on. But at the same time, we cannot handle too many concepts together too many junks of information

together which makes it extremely difficult for human mind to be able to analyze or come to terms with the complex system.

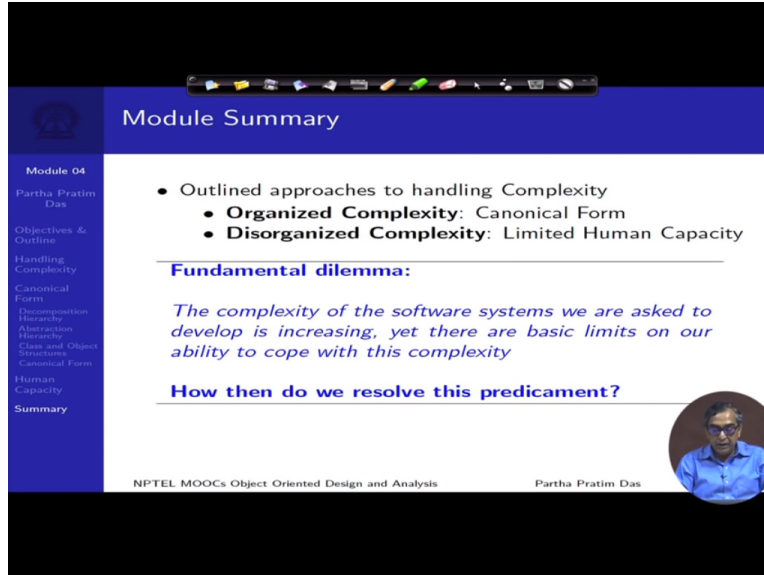
And it is believed that this you know limitation of the human capacity is has to do with limitation of short term memory which is which we quickly acquire and quickly forget and only small number of items can be kept in there. The other limitation which creates a significant bottle neck is the processing speed of the limiting is a limiting factor. The human brain is extremely slow so if you are given with a new chunk of information we take about 5 seconds to come to terms with what is it, what does it mean.

So only 7, 8 chunks of information each being processed with 5 seconds 10 seconds of time, the whole system has a disadvantage of time of not being able to handle the complexity that one need have and that it one reason that the whole process of the human engagement in dealing with complex system has to be done through different development of practices, processes, organized behavior, team building and so on.

And as I mentioned at the beginning of the module is these are still being formed and formulated. You must have heard about different paradigms of development. Some adhoc paradigm, traditional paradigm, waterfall model and then you have agile then you have iterative and all those. These are if you if you look into it these are actually attempts in trying to make better and better system.

So that from this kind of very disorganized behavior of the human mind, we can take it to a mode, structure from where you can handle more information with the less amount of time. But this remains to be a concerned area and really we will not be able to add lot of value in terms of this particular aspect of the whole problem.

(Refer Slide Time: 21:52)



Module Summary

Module 04  
Partha Pratim Das  
Objectives & Outline  
Handling Complexity  
Canonical Form  
Decomposition Hierarchy  
Abstraction  
Class and Object Structures  
Canonical Form  
Human Capacity  
Summary

- Outlined approaches to handling Complexity
  - **Organized Complexity:** Canonical Form
  - **Disorganized Complexity:** Limited Human Capacity

**Fundamental dilemma:**

*The complexity of the software systems we are asked to develop is increasing, yet there are basic limits on our ability to cope with this complexity*

**How then do we resolve this predicament?**

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

So to summarize we have outline here. Some approaches of handling complexity. Primarily we have talked about the canonical form comprising the system architecture and the 5 attributes and the canonical form is what you very clearly understand and try to remember because in the next week when we start talking about object models and so on, we will frequently fall back on this canonical form for explanation and understanding.

And all these together is called the organized complexity and we have observed that due to the limitation of human capacity we have a big part of the complexity which is still quite disorganized and difficult to handle. So in this context in this background we are not posed with fundamental dilemma that the complexity of the software system that we want to develop is regularly increasing, constantly increasing.

And but all one limitations are fixed we or capacity of being to able to cope with this complexity is fixed. So how do you resolve this predicament and that is what will be we will try to outline some of those aspects how probably we can start handling and behaving this in a way so that we can handle it better in the next module, in the module 5.