

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

Lecture – 17

Elements of the Object Model (Minor): Typing, Concurrency and Persistence (Contd.)

Welcome back to module 10 of object-oriented analysis and design. We have been talking about minor elements of object models. We have already discussed about the aspects of typing strongly and weakly type languages, statically and dynamically type languages and we have talked about polymorphism. Next we talk about the second element which is concurrency.

(Refer Slide Time: 00:52)

The screenshot shows a video lecture interface. At the top, there's a title bar with the word 'Concurrency'. Below it, on the left, is a sidebar menu for 'Module 10' by 'Partha Pratim Das', listing topics like 'Objectives & Outline', 'Elements of Object Model', 'Typing', 'Types of Types', 'Polymorphism', 'Concurrency', 'Process & Thread', 'Persistence', 'Concurrency', and 'Summary'. The main content area has a blue header 'Concurrency' and a list of bullet points:

- **Concurrency** is *the property that distinguishes an active object from one that is not active*
- Allows multiple tasks to execute, interact and collaborate at the same time to achieve the global functionality
- Concurrency is critical for *Client-Server Model* of computation

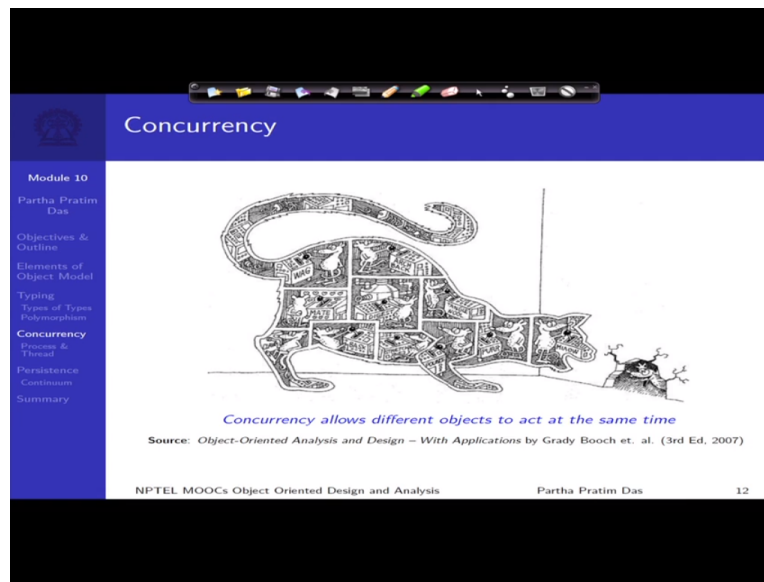
Below the text is a hand-drawn diagram in red ink showing five objects labeled o1, o2, o3, o4, and o5. o1, o2, and o3 are interconnected with arrows, indicating communication. o4 and o5 are shown separately, with o5 being crossed out with a double line. At the bottom right, there is a small circular video feed of the professor, Partha Pratim Das. The footer of the slide reads 'NPTEL MOOCs Object Oriented Design and Analysis' and 'Partha Pratim Das'.

Concurrency is as the name suggests is relates to 2 or more things happening together in the same time. So if I have one object o1, if I have another object o2, I have another object o3 and these are between themselves sending different messages and expecting to get service we have seen in this model. Then certainly you cannot have a sequential or well ordered structure to do this. You want all of these to happen as and when they need to happen.

So while o1 say is sending a message m1 to o2 to get a service at the same time, it is possibly getting message m2 from o3 to provide a service. Both of these have to happen at the same time. Now this is known as a concurrent behavior. And at the same instance it is also possible that I have object like o4, I have objects like o5, what is sitting idle, who are not neither sending messages nor receiving messages. So they are not actually doing anything.

So the concurrency is a property that distinguishes an active object from an inactive one. These are the inactive ones and these are the active objects which are doing something or the other at the same span of time. Certainly you can understand that without concurrency the abstract client server model that I talked of earlier has a basic computing model of object by systems would not at all work. Because I cannot have an ordering of exactly first step 1 has to happen then step 2 has to happen message 1, message 2, message 7, message 6. I will not be able to we will not be able to define this single temporal order for things to happen.

(Refer Slide Time: 03:13)



So concurrency is a very necessary part of the object model of the typical systems that we work with and this is the cartoonist's view of what happens inside the body of a cat is wagging the tail and digesting the food at the same time, trying to hunt something for itself.

(Refer Slide Time: 03:28)

The screenshot shows a presentation slide with a blue header and a white body. The header contains the title 'Concurrency: Example – Mobile Phone'. The body text states 'In a mobile phone many tasks happen at the same time' and lists three scenarios with their respective tasks. A sidebar on the left lists the contents of Module 10, and a small video inset shows the presenter, Partha Pratim Das.

Concurrency:
Example – Mobile Phone

In a mobile phone many tasks happen at the same time

- **Scenario 1:** While editing the address book
 - Connected to the Base Station
 - An SMS is delivered
 - Notification raised
- **Scenario 2:** While on a call
 - Connected to the Base Station
 - An SMS is delivered
 - A second call arrives
- **Scenario 3:** While watching a video
 - Connected to the Base Station
 - An App is downloaded and updated
 - A WhatsApp message delivered
 - Notification raised

Module 10
 Partha Pratim Das
 Objectives & Outline
 Elements of Object Model
 Typing
 Types of Types
 Polymorphism
 Concurrency
 Process & Thread
 Persistence
 Continuum
 Summary

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das

Coming to more mandate example which am sure would be closer to your heart all of you am sure use mobile phones in different aspects. So here am just talking about different scenarios. For example you are in your mobile phone suppose you are editing your address book; you are adding a new friend. Now while you doing this apparently your focus is on editing the address but at the same time, the phone is doing a lot of other things. For example at that same time it has to remain connected to the base station.

I connect being connected to the base station is commonly said and my phone does not have signal, my phone does have signal. That is a kind of visual indication we get. What it means that it is being there are uh land station call base stations to which my phone is currently communicating and is connected and all the time when a phone is live, when a phone has towers, it basically is connected to the base station so it cannot it is not acceptable for us that while am editing my address book, my phone does not remain connected.

So every regularly it is sending certain information to the base station, it is getting some information from the base station again sending information, getting information. So this task and this task has to happen concurrently and while this is going on, while am addressing changing the address book, I suddenly receive an sms. So the delivery of the sms happens when the phone is connected, when I am editing the address book during that time itself the sms get delivered

and as the sms is getting delivered, it is being put to my sd card uh a notification pops up by side of my address book screen to tell me that a sms has come. So that notification raise has to happen. So if you look into all of these, they will need to happen concurrently, there is no specific ordering strictly

between all of them. Of course sms being delivered will be followed by notification then there is an ordering there. But there is no specific ordering in terms of how much I add it in the address book.

And the sms being delivered and the connection of the phone. In a different scenario (()) (05:49) we are in a call the phone certainly needs to remain connected to the base station, so this is the call, this is the task, the sms can get delivered and in fact the second call may arrive. So I get to see that a call waiting is happening. So the ongoing call and the waiting call have to happen in the same time space. So this needs concurrent behavior.

You are watching a video in your mobile phone, you are still connected to the base station, so while you are watching the video, without disturbing you app is getting downloaded and updated, installed in your phone. Your whatsapp message is getting delivered and you get notification for that is raised. All of that is happening in the same time space. There could be you if you start just thinking of different instances of your use of the mobile phone.

You will at for every kind of scenario you will identify 4, 5, 10, 20 different things that need to happen, the task that need to happen at the same time and without a predefined order, a strict predefined order for things to work properly.

(Refer Slide Time: 07:20)

Concurrency:
Example – Mobile Phone

In a mobile phone many tasks happen at the same time

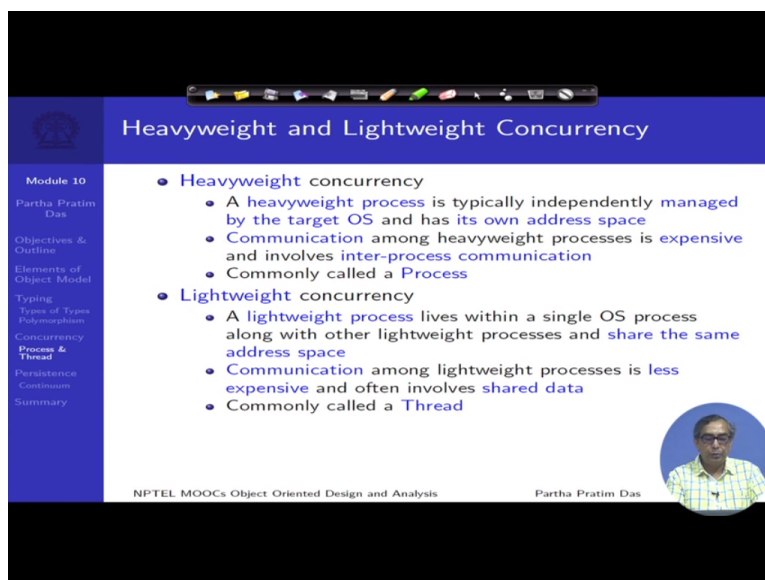
- **Scenario 1:** While editing the address book
 - Connected to the Base Station
 - An SMS is delivered
 - Notification raised
- **Scenario 2:** While on a call
 - Connected to the Base Station
 - An SMS is delivered
 - A second call arrives
- **Scenario 3:** While watching a video
 - Connected to the Base Station
 - An App is downloaded and updated
 - A WhatsApp message delivered
 - Notification raised

NPTEL MOOCs Object Oriented Design and Analysis
Partha Pratim Das

So that is a basic significance of having concurrency in object model. So if you think about all of these address, if I think about say address book is an object, if I think about base station connected is an object, if I think about sms is an object, if I think about notification is an object, if I think about call as

an object, if I think about app as an object and so on.. They all will need to interact, exist and work at the same time domain so that is the basic notion of concurrency that we need to work with.

(Refer Slide Time: 07:43)



The slide is titled "Heavyweight and Lightweight Concurrency" and is part of "Module 10" by Partha Pratim Das. It compares two types of concurrency:

- Heavyweight concurrency**
 - A heavyweight process is typically independently managed by the target OS and has its own address space
 - Communication among heavyweight processes is expensive and involves inter-process communication
 - Commonly called a **Process**
- Lightweight concurrency**
 - A lightweight process lives within a single OS process along with other lightweight processes and share the same address space
 - Communication among lightweight processes is less expensive and often involves shared data
 - Commonly called a **Thread**

The slide also includes a navigation menu on the left with items like "Objectives & Outline", "Elements of Object Model", "Typing", "Types of Types", "Polymorphism", "Concurrency", "Process & Thread", "Persistence", "Continuum", and "Summary". At the bottom, it mentions "NPTEL MOOCs Object Oriented Design and Analysis" and "Partha Pratim Das" with a small circular portrait of the speaker.

Certainly coming to the hmm technical terms, concurrency basically relates to processes in part computing system and we often talk about two kinds of concurrencies more frequently, one is known as the heavyweight concurrency or it's commonly called a process. The other is called a lightweight concurrency which is commonly a called a thread. A heavyweight concurrency or process is an entity that the os deals provide its own address space.

In contrast, a lightweight concurrency happens within the within a single os process which is a heavyweight concurrency item but it shares the address space that the heavyweight process or heavyweight concurrency of the process as it does not get a separate address space. It shares that with other lightweight process and so on. The communication typically is expensive between processes or heavyweight processes really crucial memory in terms of lightweight concurrency items or threads.

So these different kinds of concurrency models has to work with the way I want the objects to interact is so naturally objects which belong to the same module, objects which are closely integrated should work concurrently using threads. Whereas very different modules which are completely independent, which have very little functional overlap or very little code overlap would possibly work as independent processes because they need to handle very different independent resources of the same system.

So as you go to the refinement of the design in terms of the object model, you might need to look into these concurrency aspects of the object models.

(Refer Slide Time: 09:46)

Persistence

Module 10
Partha Pratim Das

- **Persistence** is *the property of an object through which its existence transcends time*

Time Object continues to exist after its creator ceases to exist

Space Object's location moves from the address space in which it was created

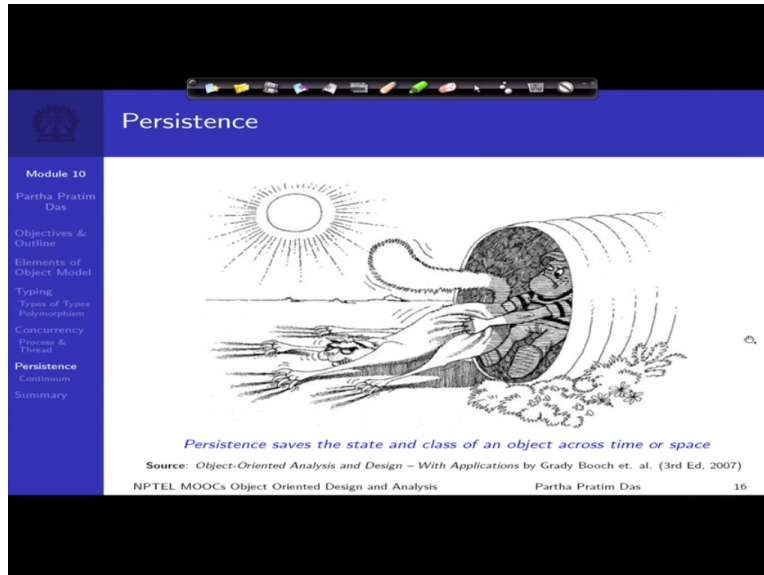
NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 15

We move on to the last minor element of the object model that is called persistence. Persistence basically talks about 2 major axis of time and access of space. We have been talking about objects all the time. Now finally when I realize in a programming system what is an object. An object will be represented by a contiguous sequence of memory location, a collection of bits together. So there is always a beginning of an object, an object gets created by some other object because everything is object.

Once it is created, it occupies certain memory space and its stays for a certain period of time, till it is destroyed. Now in terms of time and space, many things can happen, for example in object may continue to exist even after its creator ceases to exist. Somebody created that object. But that we were created that creating object has been destroyed but the created object will stay. How long it will remain is an independent factor in the design. Similarly an objects location may move, a object may have been created in this memory space memory space of thread 1.

Here I got an object this is my conceptually this is my object and this is where it is got created in terms of memory. I may move it to some other say thread 3 at a different location and say now this object has I want a blue. Now I say that this is where this object has gone. This is a is a same object but I may change its location so the space and time of an object can go through different kinds of transformation and how those happen is addressed in the issue of persistence.

(Refer Slide Time: 12:07)



Persistence

Module 10
Partha Pratim Das
Objectives & Outline
Elements of Object Model
Typing
Types of Types
Polymorphism
Concurrency
Process & Thread
Persistence
Continuum
Summary

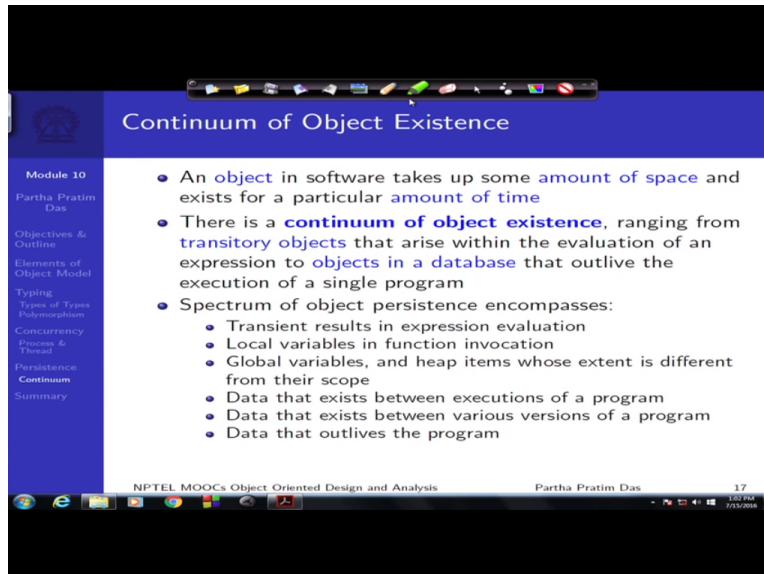
Persistence saves the state and class of an object across time or space

Source: Object-Oriented Analysis and Design – With Applications by Grady Booch et. al. (3rd Ed, 2007)

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 16

The cartoonist's view for you to enjoy, I will not go into explaining this. You should take your time of and try to understand what is being said in persistence layer.

(Refer Slide Time: 12:20)



Continuum of Object Existence

Module 10
Partha Pratim Das
Objectives & Outline
Elements of Object Model
Typing
Types of Types
Polymorphism
Concurrency
Process & Thread
Persistence
Continuum
Summary

- An **object** in software takes up some **amount of space** and exists for a **particular amount of time**
- There is a **continuum of object existence**, ranging from **transitory objects** that arise within the evaluation of an expression to **objects in a database** that outlive the execution of a single program
- Spectrum of object persistence encompasses:
 - Transient results in expression evaluation
 - Local variables in function invocation
 - Global variables, and heap items whose extent is different from their scope
 - Data that exists between executions of a program
 - Data that exists between various versions of a program
 - Data that outlives the program

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 17

Let me focus on the fact that persistence built a continuum of object existence. As I have already explained, an object let me get back to red that is more prominent. So an object in software takes certain amount of space and exist for a certain amount of time. So how much sol if you just look into these two axis, so we say this is time and this is space. So we will have objects in different points in this space. So the further down you go on the right here means is there objects which stays longer from the creation to the destruction is the longer time.

(Refer Slide Time: 13:13)

Continuum of Object Existence

Module 10
Partha Pratim Das

Objectives & Outline
Elements of Object Model
Typing
Types of Types
Polymorphism
Concurrency
Process & Thread
Persistence
Continuum
Summary

- An **object** in software takes up some **amount of space** and exists for a particular **amount of time**
- There is a **continuum of object existence**, ranging from **transitory objects** that arise within the evaluation of an expression to **objects in a database** that outlive the execution of a single program
- Spectrum of object persistence encompasses:
 - Transient results in expression evaluation
 - Local variables in function invocation
 - Global variables, and heap items whose extent is different from their scope
 - Data that exists between executions of a program
 - Data that exists between various versions of a program
 - Data that outlives the program

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 17

The diagram shows a 2D coordinate system with a vertical axis labeled 'S' (Space) and a horizontal axis labeled 'T' (Time). Several red dots are plotted at various points along these axes, representing the existence of objects at different times and in different spaces.

If I go on this, then it says that these are objects which move between more and number of spaces, address spaces. So if I look into this, then we find that there is continuum of object existence that is it varies significantly as to how long an object stays and where. For example the shortest living objects usually the transitory objects, objects that get created because you just wanted to move some value from one place to another. For example if you know C++ the basic of C++

(Refer Slide Time: 14:20)

Continuum of Object Existence

Module 10
Partha Pratim Das

Objectives & Outline
Elements of Object Model
Typing
Types of Types
Polymorphism
Concurrency
Process & Thread
Persistence
Continuum
Summary

- An **object** in software takes up some **amount of space** and exists for a particular **amount of time**
- There is a **continuum of object existence**, ranging from **transitory objects** that arise within the evaluation of an expression to **objects in a database** that outlive the execution of a single program
- Spectrum of object persistence encompasses:
 - Transient results in expression evaluation
 - Local variables in function invocation
 - Global variables, and heap items whose extent is different from their scope
 - Data that exists between executions of a program
 - Data that exists between various versions of a program
 - Data that outlives the program

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 17

Handwritten annotations in red ink are present on the slide, including the equations $x = a + b + c$, $t1 = a + b$, $t2 = t1 + c$, and $x = t2$, which illustrate the flow of data and object creation/evaluation over time.

And you are returning a value by copy then while you return that the actual object that you are trying to return cannot be returned. Because that object has to die as soon as the procedure function call is over what you do you make a transitory you make a temporary object and return that or for that matter if I am trying to evaluate something like $a + b + c$ then I cannot evaluate all of them together so what I do I kind of evaluate something like $t1$ is $a + b$ and then $t2$ say this has to go to x .

Then we say t_2 is $t_1 + c$ then will say x is t_2 because at every time the object can do only one thing make either add 2 numbers a and b add 2 numbers the temporary result of a and b with c and so on. So these kind of objects are the transitory objects or temporary objects. They naturally have very short time span because t_1 gets created here and will get destroyed here start. 2 different objects that we might need in the database think about the leave management system.

We have talked about so many employees of different types and certainly these employees stay in the system for a very long period of time once an executive has joined and has been created in the system database that employee object representing the executive will remain till that executive has been patched from the system possibly because she has left or she has resigned and so on. So the object that exist in the database often outlive.

So this is a very critical concept outlive the execution of a single program which means that the object representing an executive will remain much longer in the system compared to the method that needs to be performed for requesting a leave by that executive or by approving the leave for that executive and so on. So based on this is this is kind of a continuum hierarchy of most transient to most persistent objects these are expression evaluation I just showed you an example that how transient values happen in expression evaluation.

Similar things will happen for local function invocation you know whenever we do call by value the in c the values get copied and those copies get removed when the function invocation is over. there are global variables dynamically heap items that means dynamically allocated with allocated variables which outlive the scope that they related to and so on till you talk about really database persistent objects in the database which will outlive any kind of programs which create, manipulate and destroy them.

So this is it is it is very important to understand this continuum and whenever you are designing the object understand what is the persistence model of that object for very simple things for example if an object is very transitory it occurs and then disappears then we would like to make it very light weight we will try to keep it very light weight because if there are too many transitory objects they are getting created and destroyed, created and destroyed very frequently so system will have a lot of over weight of simply creating and destroying objects.

If an object is very large it is huge it takes a lot of space we need to measure that it does not have ample transitory behavior it should have more persistent behavior and only a small part of that object which might need changes or which might need relocation will have to be treated separately so that you do not get into performance issues so different aspects of persistence or different levels of persistence in time and its space will guide how you should design different objects.

How you should encapsulate and modularize them so that I can design an efficient system. So these factors this is the third factor that you need to keep in mind.

(Refer Slide Time: 18:36)

Module Summary

Module 10
Partha Pratim Das

Objectives & Outline
Elements of Object Model
Typing
Types of Type Information
Concurrency
Process & Thread
Persistence
Summary

- Every Object Model needs to be explored in terms of 4 major and 3 minor elements
- *Typing* is graded between strong, weak, and none; can be enforced by static, dynamic, or mixed modes; and is critical for OOP
- *Concurrency* provides the core ability to build client-server system where both can work concurrently
- *Persistence* decides the fate of objects over time and space

NPTEL MOOCs Object Oriented Design and Analysis Partha Pratim Das 18

So to summarize we have in this module talked about the three minor or useful but not exactly essential aspects of object models but I would show you from my experience that when if you want to do a good object oriented design or rather if you want to do a good design for a complex system then alongside the major element is very critical to take care of the minor elements because minor elements relate to three major aspects of implementation.

Which often turn out to be the core nonfunctional requirements of a system leading to performance in terms of time, performance in terms of space and general behavior of how responsive the system is, how robust the system is and so on. So please keep the aspects of typing in mind which creates between strong, weak and more type between static and dynamic and is critical for oop you will need to certainly be conscious about the concurrency to be able to build client server models.

And persistence will immediately tell you what kind of system you will want to build. For example a

very simple example for persistence would be you would need to decide whether all the objects outlive your application or not. If they outlive then your immediate conclusion is there will have to be a database there will have to be a database to store, there has to be a persistence tool to keep the objects because otherwise all variables.

And objects that you create would deal within a program would certainly disappear will get forcibly destroyed as long as your main function can. So you will not be able to work with them next time you run your program. So these are the considerations that you will need to work with. So with this we will conclude on the elements of object models and next we will start talk about the other details of objects models from the next module.