

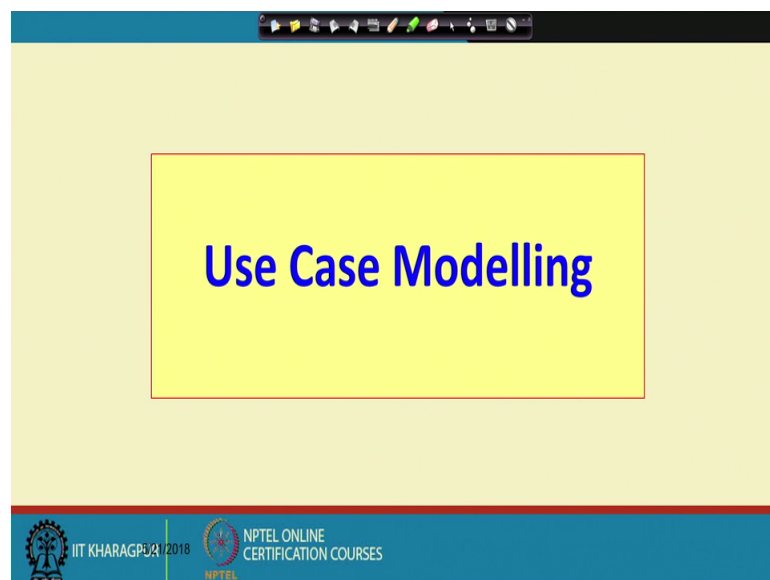
Software Engineering
Prof. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 31
Use Case Modelling

Welcome to this lecture. In the last lecture we had discussed about introductory topic on object oriented design. And, we had said that UML the unified modeling language has become a defector standard for object oriented design. We saw that UML has evolved over the years from the techniques that existed in those days, and we also saw that modelling using UML consists of creating many types of diagrams.

And, the first diagram to be constructed while modelling problem is the use case diagram, use case diagram is the central diagram because that portrays the customer's view of the system. And therefore, that is the model the use case model needs to be constructed first and all other models that are constructed subsequently must confirm to the use case model. Let us first discuss about the use case model how to construct, the use case model for a given problem?

(Refer Slide Time: 01:38)



And, then we will discuss about the other models that need to be developed as part of software development exercise, let us first look at the use case modelling.

(Refer Slide Time: 02:01)

Use Case Model

- Consists of a set of “use cases”
- It is the central model:
 - Other models must conform to this model
 - Not really an object-oriented model, it is a functional model of a system

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

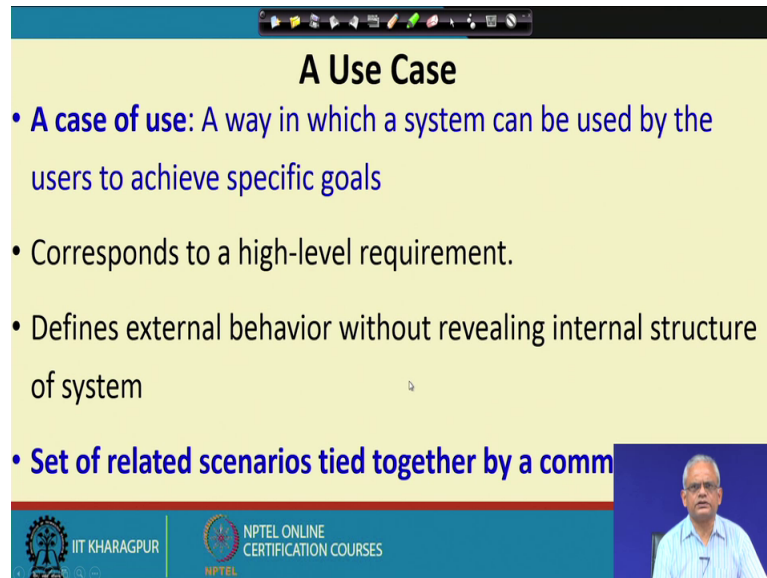
The use case modelling captures the users view of the system and it consists of a set of use cases. The use cases are something similar to a high level requirement, but then here we have a graphical formalism to model a use case. And, also have a text description for each use case that gives the specifics or the details of what happens during execution of the use case.

There are various models that need to be developed as part of a software development exercise, the structural view in terms of class diagram, the behavioral view etcetera, but the see here that we have drawn the users view at the centre to imply that this is the most fundamental view of the system, it captures the requirements of the system as the user views the system and that is basically the requirements.

And therefore, for every development the use case model is the one to be constructed first. And, once the use case model is developed the other models can be constructed, but they must confirm to the use case model. Another, thing we need to point out that even though this is part of an object oriented development an object oriented design technique, but then the users view is not really object oriented, it captures the various functionalities that the system performs. And therefore, we can say that all other models are object oriented model in the true sense, whereas the use case model is a is not a object oriented model actually we do not show the objects and so on.

It is in fact, a functional model, where we model what are the functions that we performed using the system.

(Refer Slide Time: 04:30)



A Use Case

- **A case of use:** A way in which a system can be used by the users to achieve specific goals
- Corresponds to a high-level requirement.
- Defines external behavior without revealing internal structure of system
- **Set of related scenarios tied together by a comm**

The slide features a blue header with a navigation bar, a yellow main content area, and a blue footer. The footer contains the IIT KHARAGPUR logo, the NPTEL ONLINE CERTIFICATION COURSES logo, and a small video inset of a man in a light blue shirt speaking.

But, then what exactly is a use case, a use case as we had said is a similar to a high level requirement, but then the way it is defined here is a way in which the system can be used by users to achieve some specific goals. That basically is a high level requirement for example, let us say a library software, it has various categories of users the members of the library the librarian the accounts department and so on.

The users of the library that is the members they use the software to issue book, return book, renew book, search for book etcetera. So, we can say that those are the use cases for which the user is the member. As far as the librarian is concerned the librarian also uses the library software, but for him he does member record creation, member record deletion, book record creation, book record deletion, checking the availability of a book etcetera.

So, those are the use cases or the high level requirements from the perspective of the librarian. And, similarly for the account checking the total investment in the library, the total fines collected over a period and so on. So, the use case is similar to a high level requirement, but here we represent it differently we have a graphical representation of a high level requirement, which you call as a use case and also we have a text description. As is any functional requirement it only details the functional behavior, without

revealing the internal structure of the system, but as we will see that each use case has a set of scenarios.

We will see what are the scenarios for each use case how to identify and how to document those scenarios as we proceed in this lecture.

(Refer Slide Time: 07:22)

The slide displays a list of use cases for a library information system. The use cases are: issue-book, query-book, return-book, create-member, and add-book, etc. A diagram shows three actors: Account, Member, and Librarian, connected to a central system box. A yellow box on the right is labeled 'Example Use Cases'. The slide footer includes the IIT Kharagpur logo and the text 'NPTEL ONLINE CERTIFICATION COURSES'. A small video inset of the presenter is visible in the bottom right corner.

As we were just mentioning, that for any software we can ask the question that how do the users use it. Each software has various categories of users, we just took the example of a library information system and we said that the users are the members the librarian and also the account. As far as the members are concerned they issue book, query book, return book, renew book and so on. As, far as the librarian is concerned the use cases are create member add book etcetera.

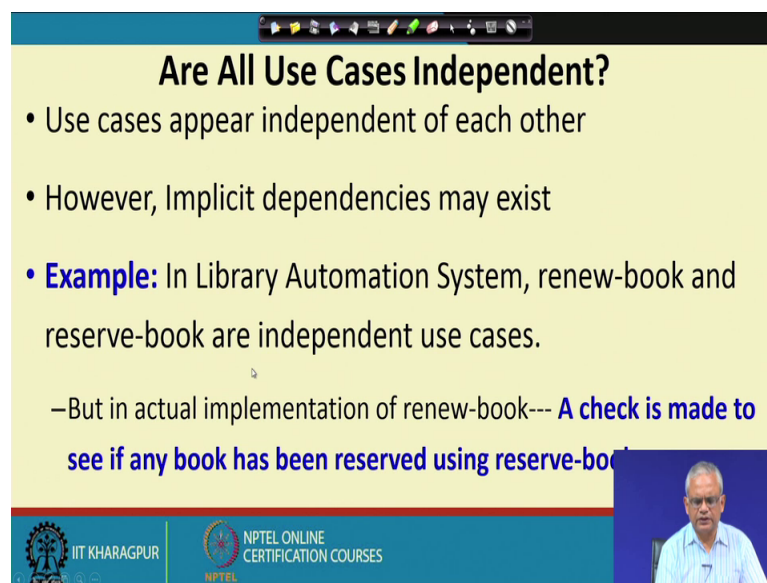
So, given any software we can consider it to be like a black box and we can think of it as offering various functionalities to the different categories of users. So, if these are the functionalities being offered by the software and the different users for those software, those functionalities. We can represent it in this form these are the functionalities, we will write the name of the functionality and also represent the specific users for those functionalities.

And, this becomes the graphical representation of the use cases, for this example we will write issue book, query book, return book, etcetera. As, the functionality for the member

and for the librarian create book, add book, sorry create member add book, delete member record etcetera. So, those are as far as the librarian is concerned and this is the member and this is the account.

So, we get a very impressive view a very simple view of a system, where just by looking at the diagram we can relate that what does the system have to offer to different categories of users, who are the users? And, what are the different categories of functions that they can perform using the system.

(Refer Slide Time: 10:20)



Are All Use Cases Independent?

- Use cases appear independent of each other
- However, Implicit dependencies may exist
- **Example:** In Library Automation System, renew-book and reserve-book are independent use cases.

–But in actual implementation of renew-book--- **A check is made to see if any book has been reserved using reserve-book**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The slide features a yellow background with a blue header and footer. A small video inset in the bottom right corner shows a man in a light blue shirt speaking. The footer contains the logos for IIT Khargapur and NPTEL Online Certification Courses.

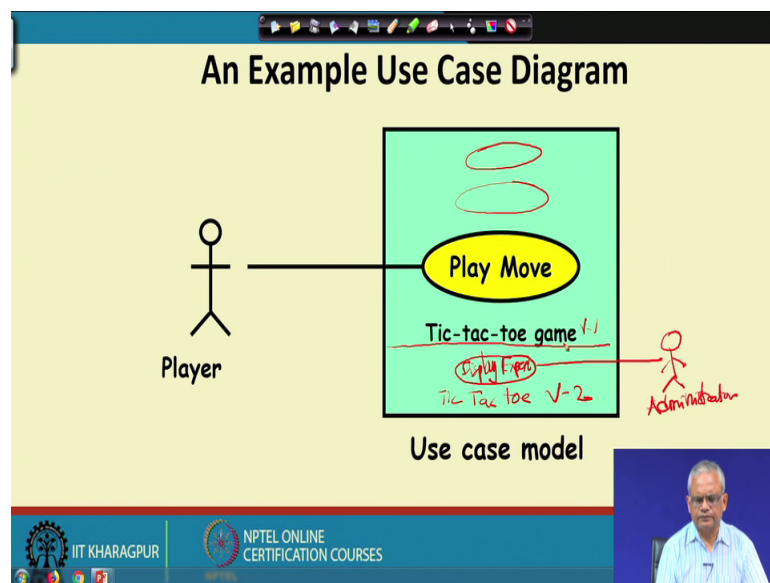
But, one question that if we identify the different use cases. Can I say that all use cases are independent of each other? For example, let us take about the library information system, you can issue book, return book, renew book, etcetera. Let say from the members perspective library members perspective, but are these different use cases really independent, because we drew them as independent use cases they have no dependencies that is what we drew them separately so, no dependency between them ok.

To answer this question is that as far as a first level view of the use cases are considered they are all independent, but if we broke down there may be dependencies. For example, issuing a book by a member may be dependent on another member, whether he has reserved.

So, if a member reservation use case there will be a dependency of the issue use case with the reservation use case, because if the same book is reserved by another member then it cannot be issued out. So, if we look at the details of various use cases, we might see that there are dependencies, but then we do not show those dependencies here in the use case diagram, but it is good to remember that there can exist dependencies between different use cases.

We just saying about renew book and reserve book this is a another example actually, that if somebody member wants to renew a book, then he goes and presents the book, but then in between if another member has reserved that book then he cannot renew it. So, the success of the renew book depends on whether another member has reserved the same book. So, there is a dependency of the reserve book and the renew book is dependent under reserve book.

(Refer Slide Time: 12:59)



This is the model very simple elegant model here for the use case model. Every use case represent by a ellipse write the name of the use case. Typically the name of the use case is a verb, because each use case depends functionality or action being performed. Write the name of the system here and then write the use cases and the different user's different categories of users actually, because there may be many players, but then we just represent the class of players here. Similarly, for a library system there may be many

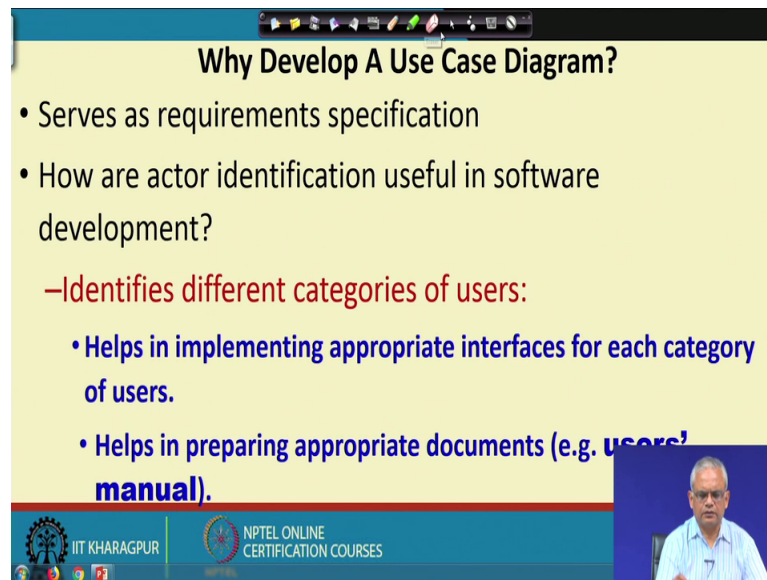
members, but we just draw one stick icon this called as a stick icon and write member here.

So, that denotes the member category of users. And, there is a line connecting from the user to the use case indicating that they are associated or the user can invoke the play move use case. And, there is the system boundary it indicates that, what are the functionalities that are available within this system boundary? If a system has multiple religious then we can draw different boundaries here for example, I might draw a boundary here. And, write that let us say find expert player or display expert player. In the next version I will write here Tic-tac-toe game version 1 and Tic-tac-toe game version 2 display expert and this will be the user is a administrator.

The administrator can display all the expert category of players and this will be done in the version 2. So, this boundary specifies all the functionalities there can be other functionalities. The functionalities that will be done in version 1 and the functionality that will be done in version 2, but in any case the boundary can be drawn to indicate even if there are no versions to indicate that what are the functionalities that are available? But, sometimes the boundaries are also not drawn as you can see in different books literature tools and so on sometimes the boundary is not drawn only the use case and the actors.

So, the boundary is not really a mandatory requirement for modelling a use case it is the ellipse indicating the use case the association relation and the user the boundary is optional, but it helps to show clearly what are the functionalities are available and we normally write the name of the software.

(Refer Slide Time: 16:55)



Why Develop A Use Case Diagram?

- Serves as requirements specification
- How are actor identification useful in software development?
 - Identifies different categories of users:
 - Helps in implementing appropriate interfaces for each category of users.
 - Helps in preparing appropriate documents (e.g. users' manual).

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, before we look at the netegrity of the use case diagram let us see, what are the uses of the use case diagram? We can easily guess that it serves as the requirement specification very elegant one, it first gives a graphical view of all the requirements and then it is also accompanied by a text description, which gives the detailed behavior associated with the requirement. That is what does the system? Do given a input how does the system behave or what does the output that it produces?.

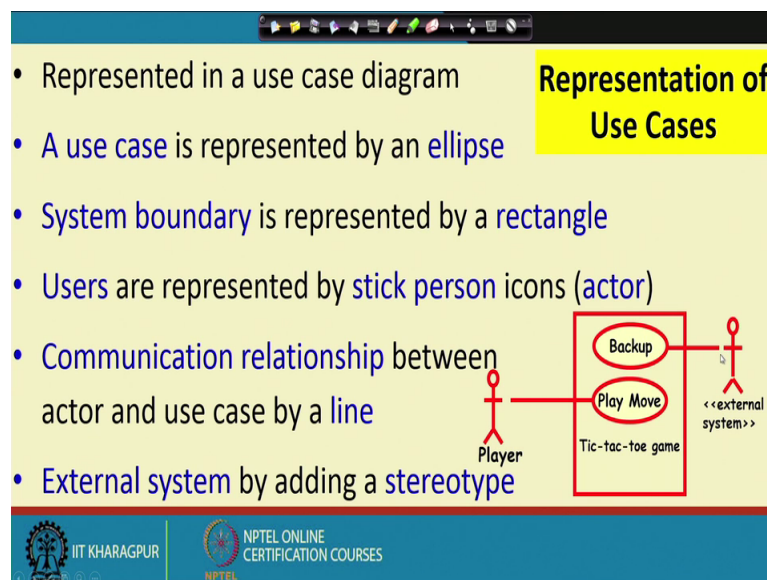
But, that is understandable that we identify the requirements which are the ellipses, but how do identifying the different categories of users? That is the different categories of users are called as actors in the UML terminology, how do how does indentifying the actors help in developing the software, does it have any rule? Yes, it has rule when we identify different categories of users, we can imagine the different user interface that each category needs, let us say software has factory workers as a category of user and the system administrator is another category of user.

The factory worker they need a very simple interface, because they are not very familiar with the software very elaborate user interface where they are prompted to enter some data they enter it and so on. Whereas, the system administrator he does not need a very elaborate interface he needs a interface which would work very fast. So, he can just press some keys and get things done very fast, whereas for the factory workers there will be prompt for everything they need to do and that has to be in very simple format.

The second use of identifying the actors is that it also helps in preparing the user manual, if you can identify who are the users the user manual can be developed by targeting that those. For example, if factory workers are the user of the software, then the user manual has to be written in their language in a very very low level description, very very simple understandable description will be there, but if the software will be used by system administrators and computer experts, then the user manual can be can assume certain background knowledge and it can be presented at that level.

So, identifying the category of users is helpful as the development proceeds. First is in developing the gui part and second is preparing the documents specially the user's manual.

(Refer Slide Time: 20:33)



The representation of the use cases we had seen it is drawn as a ellipse, the system boundary is drawn by a rectangle, the users are called as actor, they are represented by stick person. These are the terminologies we must be familiar while using the UML the actor, the use case and the association relation between the actor and use case this is also called as a communication relation or association relation so, drawn by a line.

Sometimes we need to represent an external system, because one software one system might interact with another system. So, we can consider the other system to be a kind of a user here and we represent the external system, using the same stick person icon and

we just annotate that which we call as a stereotype we just annotate that saying that it is an external system.

For example, let us say in the game backup need to be taken for every game the backup is made on a external system. So, we represent the external system using a stick icon, but we just write here within these 2 symbols these are called as the (Refer Time: 22:26) and this is called as the stereotype. The external system this is actually a system not a human user, but then for simplicity we represent it as a stick icon and we write that, but the backup the external system is involved and this backup is taken under external system.

(Refer Slide Time: 22:57)

What is a Connection?

- A connection is an association between an actor and a use case.
- Depicts a usage relationship
- Connection does not indicate data flow

The diagram shows a stick figure actor connected by a line to a use case box labeled 'Play Move' within a 'Tic-tac-toe game' system boundary.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us see what does the connection mean the line between the actor or the user and the use case? The line indicates an association relation or a communication relation the user invokes the use case or the user uses the use case, but then it does not indicate that he will input some data etcetera. He may just invoke it we do not capture here data flow unlike in a DFD, where we discussed about external entities and we are interested in what kind of data they input? Here, we do not capture the data flow we just indicate here that the user uses the use case.

(Refer Slide Time: 23:54)

Relationships between Use Cases and Actors

- Association relation indicates that the actor and the corresponding use case communicate with one another.

The diagram shows a yellow oval use case labeled 'update grades' connected by a blue line to a stick figure actor labeled 'faculty'. The slide includes logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a speaker.

This is another example, that the faculty is a category of user who can update the grade.

(Refer Slide Time: 24:12)

Another Example Use Case Diagram

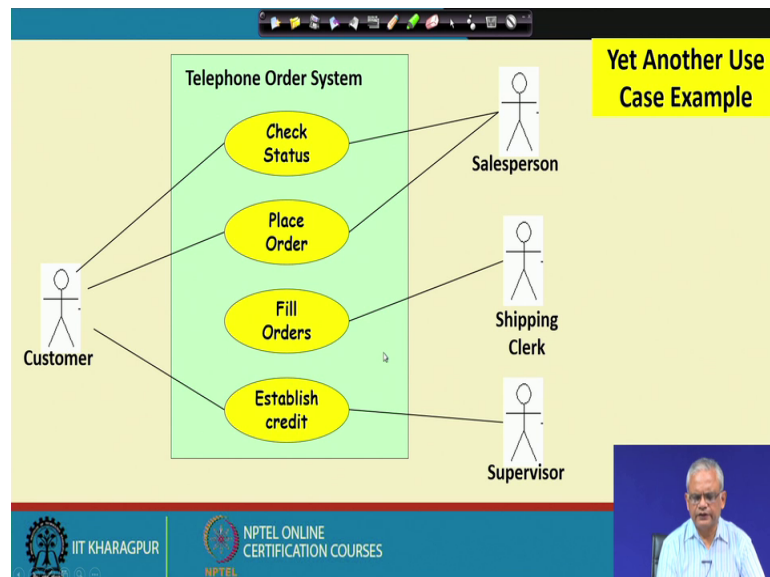
The diagram shows a central green box labeled 'Video Store Information System' containing two yellow oval use cases: 'Rent Videos' and an empty one with three dots. A stick figure actor labeled 'Clerk' is connected to the 'Rent Videos' use case. Another stick figure actor is connected to the same 'Rent Videos' use case, with the label '<<external system>> Credit Authorization Service' below it. The slide includes logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a speaker.

This is yet another example is a video information system, there are many use cases here just represented one of the use cases to explain what is indicated when 2 actors are connected to the same use case. It indicates that this use case can be invoked by both these actors or for the success of the use case execution 2 actors are involved. One implication is that if there is another actor here let say user they can independently

invoke the use case, but here the implication is that both need to participate. The clerk when he rents the videos the help of an external system is taken for credit authorization.

So, before a person who wants to rent the videos presents his video BCR, the clerk text that and enters the details and checks whether the member is creditworthy by using an external system automatically it is checked as part of this use case. And, then the rent video will be success otherwise, if he is not creditworthy the rent video will say that the video cannot be issued. So, we will often draw multiple actors using the same use case they collaborate for successful execution of the use case.

(Refer Slide Time: 26:12)

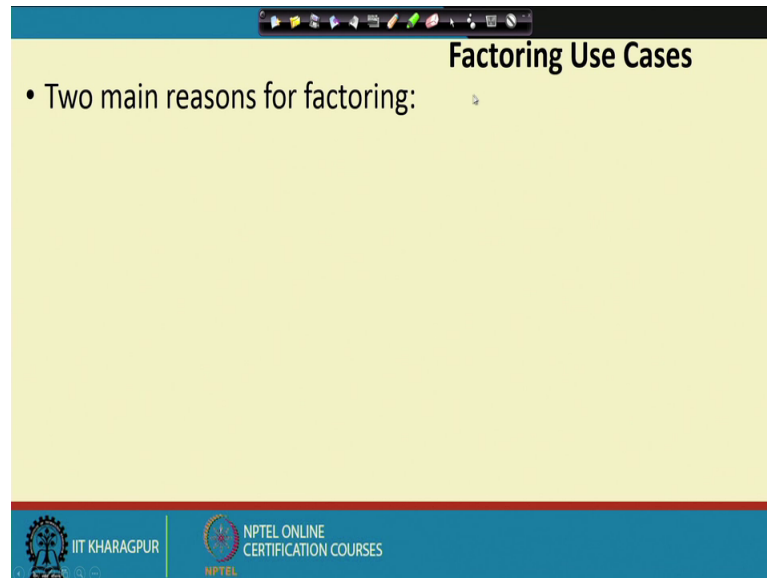


This is another example here, this is a telephone order system here this commerce application we have the customer can place order through telephone, can check the status of the order, the customer can place order and check the status of the order and also while placing the order he needs to establish the credit possibly by entering the credit card number etcetera. The details of those what really happens will be given as separate text description.

But, here by looking at the diagram we can see that the customer can invoke 3 functionalities of the telephone order system and check the status place order and establish the credit. The salesperson can help in placing the order and the salesperson also can check the status the salesperson can himself or herself place an order the

shipping clerk fills the orders and dispatches the supervisors he is involved in establishing the credit.

(Refer Slide Time: 27:41)



The image shows a presentation slide with a yellow background and a blue header. The header contains the title "Factoring Use Cases" in white text. Below the title, there is a single bullet point: "• Two main reasons for factoring:". At the bottom of the slide, there is a blue footer containing the logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

So, far we have looked at some very basic concepts in use case modelling looked at the graphical representation of a use case diagram.

Right now, we are nearly at the end of this lecture we will stop here and in the next lecture we will discuss about some details of use case modelling namely; how to decompose a use case sometimes the use cases are complex and we need to represent them as separate use cases by decomposing them, we will see how to decompose? What are the different techniques? And, we will also see the text description how to write the text description for a use case. We will stop now.

Thank you.