

Principles of Engineering System Design
Dr. T. Asokan
Department of Engineering Design
Indian Institute of Technology, Madras

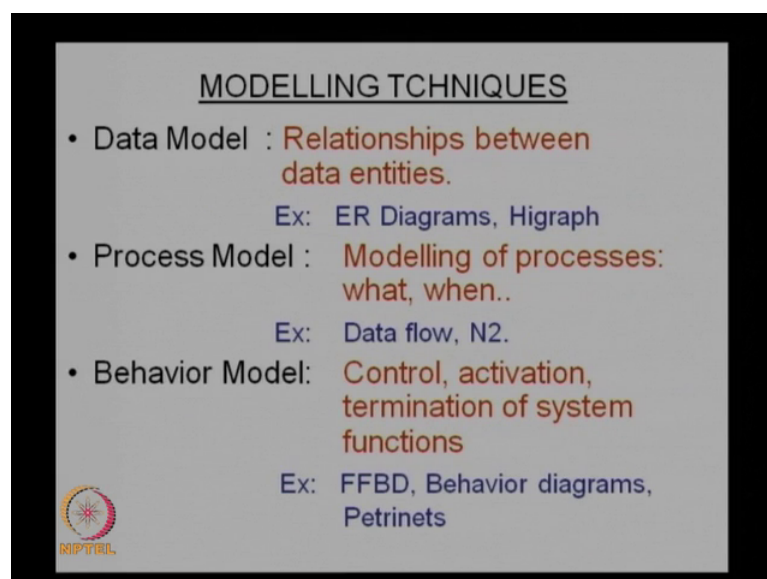
Lecture - 23
Graphical Modelling Techniques

Dear friends, welcome back. Today we will discuss some of the Graphical Modelling Techniques used in the process of Engineering System Design. These are some of the qualitative modeling techniques, widely used in industry for the different stages of design process.

In the last few classes we discussed about the various processes in system design, and we found that there are many tools to be used at various stages; especially when it comes to the functional modeling, architecture development, interface development, etcetera. We need to think of various modeling techniques so that the process becomes much simpler and then gives a much better idea about the whole process what we are doing.


So, in this lecture and then the coming few lectures I will discuss about some of the very commonly used methods for modeling of system as well as some of the processes in the system design.

(Refer Slide Time: 01:16)



MODELLING TECHNIQUES

- Data Model : Relationships between data entities.
Ex: ER Diagrams, Higraph
- Process Model : Modelling of processes: what, when..
Ex: Data flow, N2.
- Behavior Model: Control, activation, termination of system functions
Ex: FFBD, Behavior diagrams, Petrinets



The main techniques what we are going to discuss or the modeling techniques the first one is basically known as the data model. It is the relationship between data entities. So, we have various entities in the system and there are lots of data exchanges between these entities. So, we need to model this data relationship, basically the input output relationships between the entities and we use the data modeling techniques to do this. The mainly used data model techniques are the entity relationship diagrams and higraph. We will go through these methods ER diagrams and higraph, but before that let me explain the other two methods, where we use for modeling the other one is known as process modeling.

So, the first one was data model which actually gives the relationship between the data entities and in the process model, we discuss about the modeling of processes basically the flow of functions in the system. It identifies the processes what processes done and when it is done and so on. So, here in the process modeling, the focus is on identifying the flows between the entities. So, especially in the functional modeling, we need to identify what function is processed by which entity and how this function flow is taking place. So, in order to do that we use the process model and here we have different methods; one is known as a data flow or method the other one is known as an N two diagram these are somewhat similar to the IDEF0 diagram we already discussed, but these are alternative methods for modeling of the process within the system.

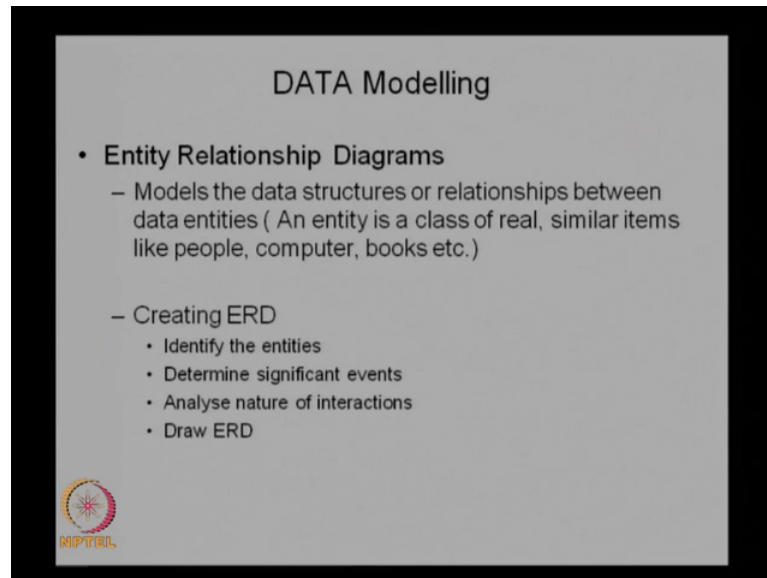
The last one is known as the behavior model. In behavior model we look at the controlled and activation of various processes in the system. So, how is it controlled and how the flow is taking place, what kind of control input is needed those things are actually modelled using the behavior model. Especially the termination of system functions these are modelled using the behavior model. The methods commonly used are the FFBD that is the function flow block diagram, then behavior diagrams and petrinets.

So, all these are used to model the control activation termination of system functions at various stages. So, this is needed to identify, what kind of control action we need to provide and when actually a particular function terminates and what kind of control inputs are needed and what kind of processes are taking place parallely.

So, are these can be modelled using the behavior models. So, these are the three important modeling techniques will be discussing in this lecture. So, we will start with

the data model which is the relationship between data entities. So, we will start with the ER diagram or the entity relationship diagram.

(Refer Slide Time: 04:02)

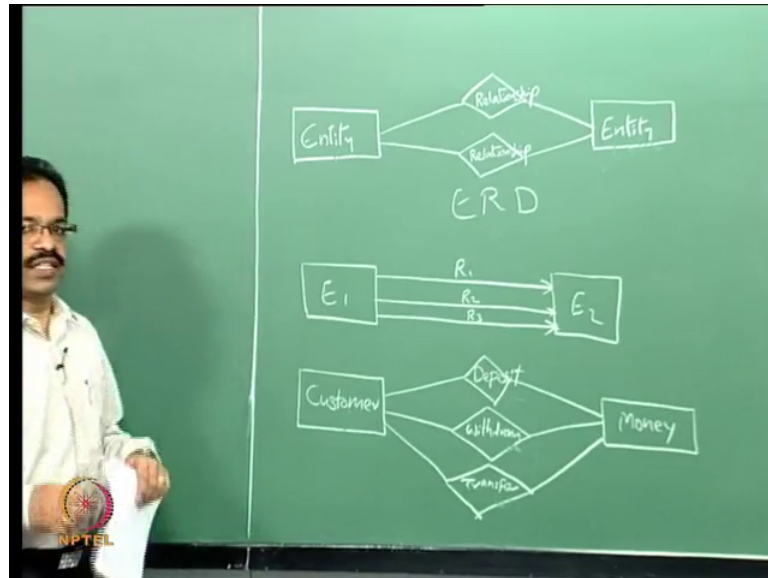


The entity relationship diagrams, which models the data structures or relationships between data entities. So, here the relationship between the data entities are modelled using the entity relationship diagram and an entity is defined as a class of real similar items like people, computer, books etcetera. So, these entities we model the relationship between this entities using the entity relationship diagram and how do we create ERD. So, here these created basically the entity relationship diagram are created by identifying the entities. So, we identify all the entities in the system, and then we identify the significant events.

So, what are the significant events taking place and then analyze the nature of interactions and then draw the entity relationship diagram. So, as I mentioned it is basically explains the relationship between the entities. So, in order to draw the data flow diagram, we need to identify the entities and the relationship between the entities. And once we identify this interaction between the entities we can actually draw the entity relationship diagram. I will explain some of the methods used for entity relationship diagram to draw the diagram. So, let us go to the board and then see how the entities are related or represented and how the relationships are explained using entity relationship diagram.

So, in entity relationship diagram, all the entities are represented using a square block. So, all the entities are represented using a square block.

(Refer Slide Time: 05:37)



So, this is the entity and the relationship between two entities. So, we have another entity over here, the relationship between these two entities are represented using a diamond shaped box. So, this is the relationship and it is connected like this. So, if you have more relationships, you can actually draw more diamond shaped boxes to represent the relationship. So, this is the basic entity relationship structure. So, here as you can see we can have many entities and many kind of relationships. So, the entities are normally represented using the square blocks and the relationship is represented using diamond shaped boxes. Another way of doing this is basically you can instead of having the diamond boxes we can actually have directed the relationships, where we simply show an arrow a directed arrow to represent the relationship.

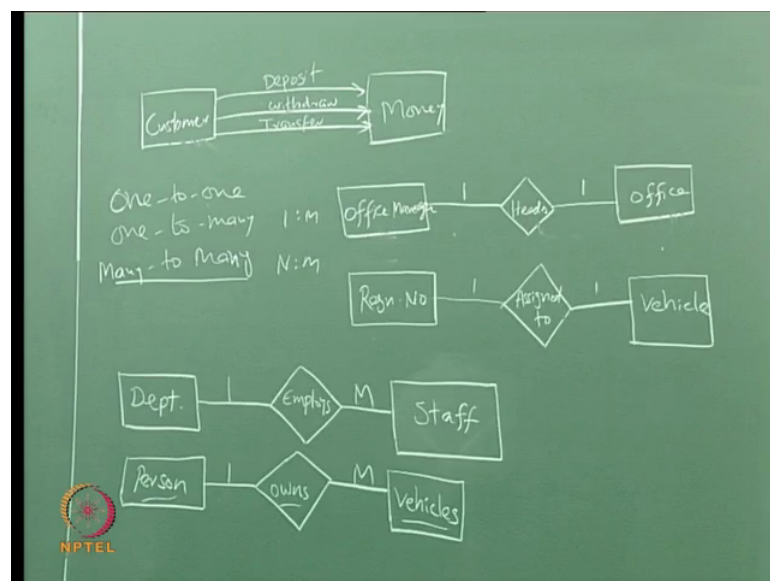
So, this is entity 1 this is entity 2 and we can have the relationships written over here R1 R2 R3 the relationship. So, in this case if you are using the diamond shaped box to represent the relationship, then we do not use the arrows, but when we are not using the relationship we will use arrows to represent the directional relationship. For example, we take a case of a customer making deposits, how do we represent the relationship between the customer and the money. So, customer is an entity here is a very simple example. So, we take the customer and money at two entities. So, as we mentioned an entity can be

anything it is a man machine material or any other thing, which can actually be represented.

So, this is one entity and this is another entity and when a customer makes a deposit. So, we can actually write down the relationship like this. So, here actually customer deposit money. So, that can be one relationship customer can deposit money and the other one is customer withdraws money maybe another relationship of course, you can have many this kind of relationships and transfer money. So, these are the three relationship we can identify customer deposit money customer withdraws money customer transfers money. So, this actually shows that these are the kind of relationship a customer can have with money and that actually represented using a graphical methods. So, it is model of a qualitative modeling technique where we represent the relationship between the customer and the money or between different entities.

We can have different relationship depending on the scenarios, we can actually identify many other scenarios like customer wants to make a print out of the account or the account statement, then the entities will be different here the customer will be the entity here the entity instead of money will be having a different entity. And the same relationship can be represented using the directional arrows also. So, instead of the diamond boxes you will be having the directional arrows.

(Refer Slide Time: 09:38)



So, these are the two entities. So, this is another way of representing the same ER diagram customer money and here you will be having the deposit at the relationship deposit and transfer or withdraw.

So, we can represent the relationship is either using this methods or using this method in both cases, you will see that there is a relationship here the arrows are shown here in this case, but in this case arrows are not shown. This is the way how the entity relationship diagram is drawn. These are very simple cases then there is methods were used for a long time to represent the relationships. And here you can actually have different kinds of relationship also you can have a one-to-one relationship between entities or you can have a one to many relationship or you can have a many to many relationship between the entities. So, that also can be represented using the entity relationship diagram. So, here you can see if you have two entities like a office manager and office.

Suppose these are the two entities then we can have a relationship here, office manager manages the office or heads the office. So, this is the relationship between the entities. So, this is entity 1 office manager and this office is the entity 2. So, here you can have a relationship office manager heads office. So, normally we will have a one-to-one relationship for this. So, it is one office manager heads one office. So, that is the one-to-one relationship in entity relationship diagram. So, in some cases like you can have again an office or we can have take another example for the same one-to-one diagram one-to-one relationship is vehicle registration number of vehicle this is again an example for one-to-one relationship.

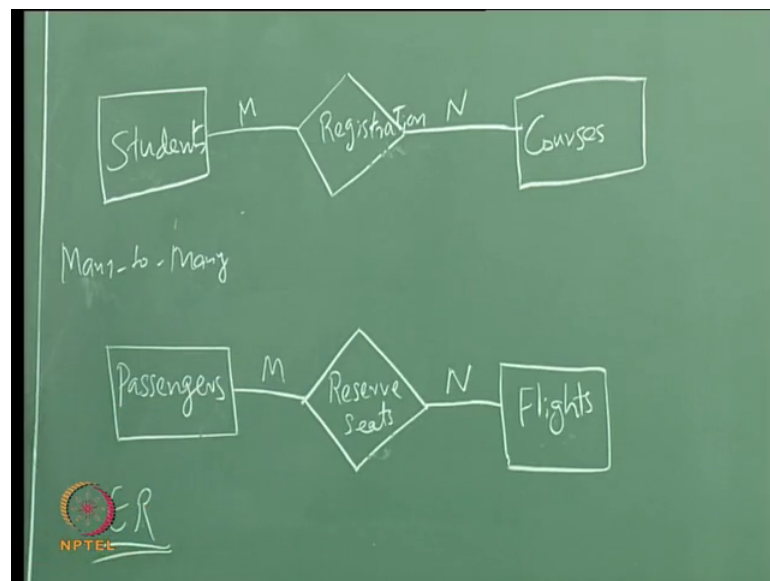
So, registration number assigned to vehicles. So, here you can see that one registration number is assigned to one vehicle. So, again this is a one-to-one relationship. So, the relationship where is only one entity, related to one entity that is represented by one-to-one relationship. And if you have one to many relationship then we will be representing in a different way the entity relationship diagram will be almost the same except that we will be having a relationship shown with the numbers over there. So, one to many is normal represented as 1 to M and this will be sometimes at the present N to M as many to many that is N entities related to M entities.

So, here you can take an example for the one to many relationships. So, if you have a department suppose we have concerning a department in an engineering college. So, we

can see that department employs sorry. So, this is again entity say department employees staff. So, here you can see that one department will employ many staff. So, this is a 1 to M relationship similarly we can have one person you can have a person owns vehicles. So, you can have that person as an entity owns as a relationship and vehicles has other entity. So, here again the relationship is that one person can on many vehicles. If it is only one vehicle then it would be one person owns one vehicle, but in this case one person can on many vehicles.

So, the relationship will be a 1 to M relationship in the entity relationship diagram that is the 1 to M relationship. The same way you can actually have a many to many relationship also, in many to many relationship you will be having many entities over here interacting with many other entities through the a single relationship. So, if you take this as an example we can see here.

(Refer Slide Time: 15:10)



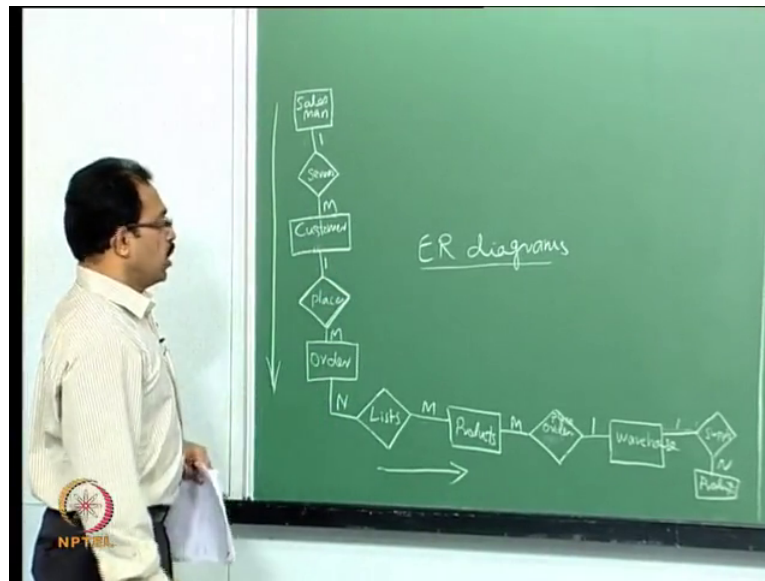
You take the example of reservation system or the students registering to a two courses; the students are an entity then courses. So, you can see that there are M students can register for N courses. So, this is registration for courses; so registration as an entity I as a relationship. So, M students register for N course that is M to N relationship. If you take a single student then it will be 1 to M relationship one student registering for N courses then it will be like a 1 to M relationship.

So, in this case when you take M students they register for N courses. So, M students register for N courses in a say a many to many relationships. So, these are known as many to many relationships. Similarly you can see the ticket booking of by passengers. So, passengers reserve seats. So, flights or trains, you can see again this will be an M to N relationship, you have M passengers reserving seats for N flights; so again M to N relationship.

So, if you take a single passenger, then it will be a N to N relationship otherwise it will be M to N relationship. So, you can see here the ER diagrams can represent various scenarios, basically the relationship between various entities are shown here and in this entity relationship we can actually represent the entities as rectangular boxes and relationship as diamond shaped boxes, and then can have this relationship either using a diamond box relationship or you can have a directed arrows. So, when you are using directed arrows, we do not use the boxes over here we describe the relationship as R1, R2, R3 whatever may be the relationship.

So, these are some of the examples for the ER diagram simple ER diagrams and in ER diagram you can have one-to-one relationship or one to many relationship or N to M relationship. So, any relationship can be represented using the ER diagram and we saw some of the examples where you can have a one-to-one relationship or 1 to N relationship or N to M relationship. So, these are the simple ER diagrams, now if I combined these relationships 1 to N and N to M relationship I can show you a simple example how do we actually combined these things to make a scenario. So, take the example of a salesman basically nowadays you can see lot of salesmen come directly to the houses and do a direct selling. So, if you take an example of a salesman.

(Refer Slide Time: 18:29)



So, if you consider a salesman as an entity. So, a salesman will actually serve the customers or the when I visits the homes or the shops were of whatever maybe the situation. So, he serves customers, we can take customer as another entity. So, here you can see this is a 1 to M relationship one salesman will serve M customers one salesman can serve many customers. Therefore, this will be a 1 to M relationship and then one customer will make base many orders. So, a customer places. So, again he can see that one customer now we are taking only one customer here, one customer places M orders. So, he is again a 1 to M relationship, because the customer can make many orders and many orders. So, the once the salesman is taking the order. So, he will be taking order from many people. So, finally, he will be getting many orders with many items.

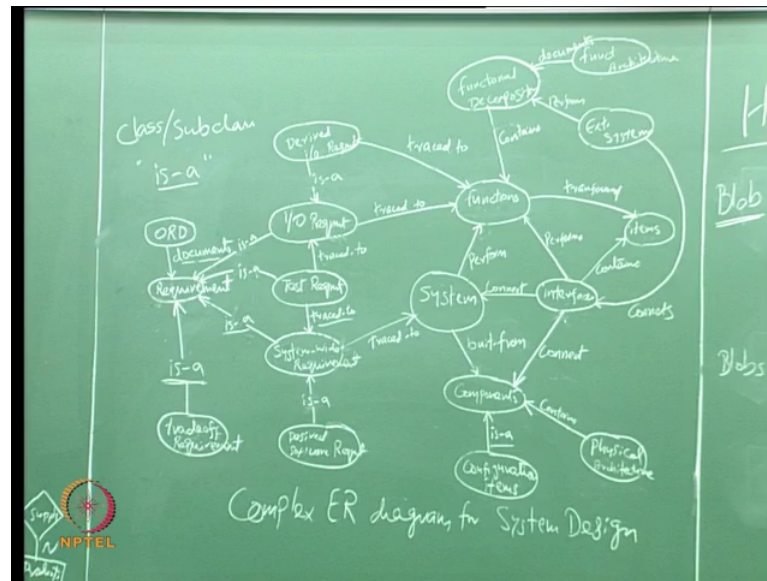
So, he will actually list all the orders, this is the relationship the orders are listed together. So, he will be getting list with the many products; so the products that of final product relationship, the list will be having many products. So, he will be having N list or I am using N for to say the many relationship. So, N list with M products; this is again a many to many relationship. So, from multiple customers will be getting orders and there will be N list with M products and this will be to the store. So, this list will go to the stores or. So, here this is the warehouse entity. So, all these list will go to the warehouse. So, you will be having when one warehouse where the items are stored.

So, you will be having one warehouse with M product and when the order placed a place order. For the M products now order will be placed to one warehouse. So, this is the relationship from the salesman, one salesman serves many customers. So, a 1 to M relationship for the salesman to customer and then one customer places many orders many orders are listed in to the M products. So, M products are listed in this M to M relationship between the order and the products there are N list and M products and these M products are ordered to the one warehouse. So, one warehouse will be serving all the customers and then again you can see that one warehouse sends the product to the customer, individual customers. So, supply again products. So, you can see one warehouse supplying N products.

So, this is the way how the flow is represented the relationship not represented using ER diagrams of course, this is a very simple example, I have taken a very simple example to show the utility of the ER diagram to represent the data flow in a system. We can actually extend these two complex systems also. So, we will see some of the complex ER diagrams how the simple ER diagrams can be actually replaced with the complex ER diagram. So, some of the things will be simplifying instead of showing all the relationship in the diamond boxes, we will try to represent them using directed arrows and when we do this you will be getting the complex ER diagrams, which can actually represent many complex relationship between the entities. Now let us look at one of the complex ER diagram.

In complex ER diagram what we try to do is to represent the subclass relationship class or subclass relationship.

(Refer Slide Time: 23:19)



Class subclass relationships are represented using an M called is a relationship. So, a subset is represented as is a subset of other one. So, this is how we actually represent the subclasses in complex ER diagrams. We will take an example of system design itself we know that system design is a complex process. So, the system designs itself there are various entities in the system design. So, we will try to identify the relationship between this entity is using a complex ER diagram. Now here you can start with the various entities. So, here the entities are represented the using oval shape instead of the square boxes because again this is complex one. So, you need to have many entities. So, we will try to simplify it. So, the ORD will give you the requirement. So, the relationship between ORD and the requirement is represented by this.

So, this is the requirement. So, the relation between ORD and requirement is that, ORD the documents the requirement. So, this is an entity ORD is an entity requirement is an entity and the relationship is given by the document. So, ORD documents the requirements and then that is another entities, you can write it as a trade of requirement. Now, we can see that trade off requirement is a requirement. So, the relationship is given like this between these two entities, the tradeoff requirement is a subclass of the requirement and this subclass is represented by a relationship known as is a relationship. So, you can see trade off requirement is a requirement and ORD documents are the requirements.

Similarly, you can have many sub classes for the requirement because this is just one of the requirement you can actually have here in the relationship for the system wide requirement as another entity again this is a relationship. So, here system wide requirement is a requirement similarly there are test requirements, this is again is a relationship. So, test requirement is a requirement, then you have the input output requirements is a requirement. So, these are all the subclass of requirements trade off requirement system wide requirement test requirement input output requirement and then the state wide requirement again you can write the relationship.

So, here if you have some other requirement like decide software requirement. So, in some cases you may be having a specific requirement of software. So, the decide software requirement will be another requirement, but then again it is a part of state wide requirement. So, we can write it as the decide software requirement is a system wide requirement, it is a subclass of the system wide requirements.

Now again you can actually have another relationship can be identified here between the test requirement and the system wide requirement. So, the test requirement can be traced to the. So, this is the relationship between these two entity. So, this is a test requirement is an entity system wide requirement is an entity. So, we can have a relationship between test requirement and system wide requirement traced to; that is the test requirement can be traced to the system wide requirement.

Similarly, the test requirement can be traced to the input output requirement also. Basically the test requirement comes from all these systematic requirement input output requirement and all other requirements. So, we can always have a relationship this is traced to the input output requirement. And then we can have another entity here that is the derived input output requirements. So, derived input output requirement then we have a in a relationship it is a subclass of input output requirement.

So, derived input output requirement is a input output requirement or it is traced coming from this say subclass of this requirement therefore, you can actually write it as a is a relationship. So, the subclasses are represented I say using their relationship is a relationship, and the other cases we have used the trace to relationship or whatever may be the relationship will be represented as trace to or documents like that. And again in this you can see that the system wide requirements.

So, you have the main system here. So, this is the system as an entity. So, the system wide requirements can be traced to the system. So, again he had say sorry this is system wide requirements are traced to the system. So, this is a trace to relationship and then the system performs the functions. So, we know that there are many functions in the system and the system basically tries to or system is basically designed to perform this function. So, the system performs the functions, that the relationship is here perform. The systems perform the functions and of course are these can be traced to the functions. So, here this is relationship is raised to. So, again the input output requirement can be traced to the functions, derived input output requirement can be traced to the functions.

Basically, we know that the functions are there to provide these requirements or we that is why we can actually trace these to the functions. Then again we know that there are functional decompositions, they are actually different functions are identified this is a functional decomposition. So, this actually contains all the functions the functional decomposition will contain all the function. So, the relationship here is that it contains the functions. So, that is a relationship between these entities. So, all those shown in this circle or the oval shape or the entities and the relationship are expressed using the directed arrows.

Now, the system will be having many components to provide the function. So, this is the physical architecture we are talking about the components and the subsystems, and then components will be having we can actually have the relationship between system and component a system is built from the components. So, we can have the relationships here the system is built from components, the relationship is built from. So, here you can see the components actually form a system. So, system is built from components, and we have the smallest item in the physical architecture which is known as the configuration items and we know that configuration item is a subset of components. So, this is a relationship.

So, configuration item is a component. So, this is a subclass or configuration item is a subclass of component, that is why we have a is a relationship here. And then we have the physical architecture something similar to the functional architecture, we have the physical architecture and again we have the relationship components are part of the physical architecture. So, it actually physical architecture contains the components of course; it can be components or the subsystems. So, we write the relationship as physical

architecture contains the components and then we have the interfaces as another entity; here the interface is basically used to connect the components.

So, we say that components. So, the arrow should be in this direction. So, the relationship is connected and then again you can have interfaces with between systems or system connects with the external systems. So, we can have a interfaces connecting the system also. So, relationship between interface and system again is connected and of course, interface will perform many functions. Therefore, there is a relationship interface performs functions and again there will be interface items that basically the physical elements of interface items. So, this actually interface contains items the configuration items it can be components or the software or any other configuration item which is used to form the interface these are the items and the functions are produced by this items for the interface. So, this is actually functions are transformed by the interfaces.

So, that is the relation between items and transforms; and then we have the external systems as another entity external system another entity, and which actually performs the any functions and here again interface is connected to the interface. So, the interface connects the external systems. So, the external systems again the interfaces are used for connecting this. So, you have the external interface external system connected by the interface and again there are external system perform the decompositions and then you have this last item is the functional architecture which actually documents the functional decomposition. So, that is the relationship here is documents. So, this is the complex ER diagram for system design.

As we can see here as now the system design is a complex process there are many entities involved in the system design and therefore, we need to represent them using the relationships what we are trying to do is to identify all the entities and then try to see what kind of relationships are existing between these entities and as I mentioned there are different kind of relationship between the entities. So, we try to identify the relationship as well as the class or subclass relationship and the subclass relationship is represented using a is a relationship. And all other relationships are not using directed arrows as we can here if you take the system as the main entity you can have there are different relationship between the entity system as well as the requirements.

So, you have there are different requirements the input output requirement derived requirement all can be traced to the functions of the system because the system needs to perform the functions and therefore, we have many requirements identified from there and apart from the function the system has got some system wide requirement especially in terms of technology and other requirements, that is where the system wide requirement can be traced to the system, similarly there are requirements for desired software or hardware. So, that can actually be considered as a subclass of the system wide requirement.

Therefore, you are getting an is a relationship as part of the relationship between these entities; and ORD is the document. So, originating requirement document we identify or we prepare and the system design, when we are ready with all the system requirements and other identified functions and needs. So, the ORD basically documents all the requirements and again you can see these are all the requirement, which are the all these are subclass of the requirement that is why the relationship between these entities and the requirement is a relationship because it is a subclass of this entity.

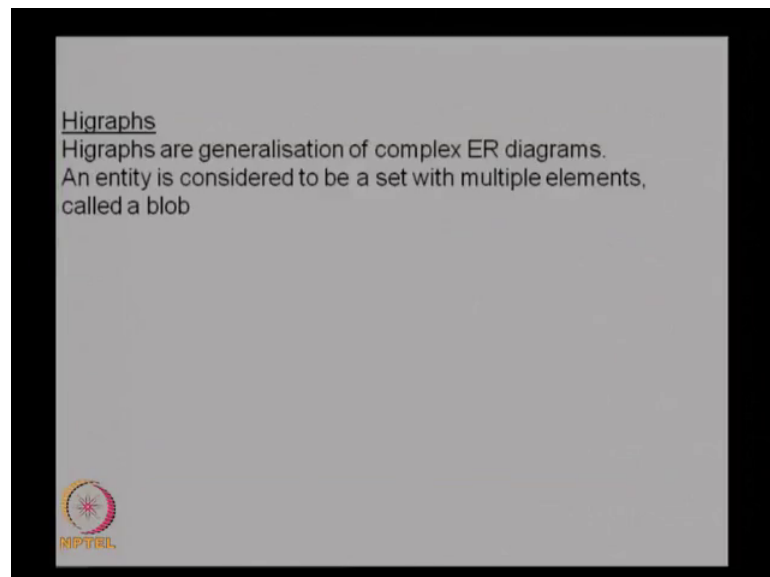
Similarly, the trade of requirement again is a subclass of the requirement that is why we can actually have the relationship as is a relationship. So, we can identify which are the subclasses just by looking at the diagram, we can see that all these are the subclass of these particular entity requirements. Similarly if you look at here you can see the relationships, the components, the interface and the functions. So, how they are related to each other and they what is the entity relationship between physical architecture in the components, similar to the configuration items as you can see components is a subclass the configuration item is a subclass of components that is why there is an is a relationship. And similarly interface has got many connections with the other entities. So, if you take interface as one entity and the functions as one entity, you can see that interface perform some of the functions what is identified in the function for the system.

Similarly, the interface connects the system as well as it connects the external system also. So, the relationship between that these two entities or the system and the external system is through the interface entity. So, the interface connects the system as well as it connects the external system and it performs many of the functions of the external system also.

And again this is the physical configuration items for the interface. So, it contains the items, and there is items are transferred to function the functions are transformed through the items. So, actually they perform the functions for the system. So, like this you can have any complex system represented using the entity relationships diagram. So, that is the advantage of using entity relationship diagram for representing the data flow in system design.

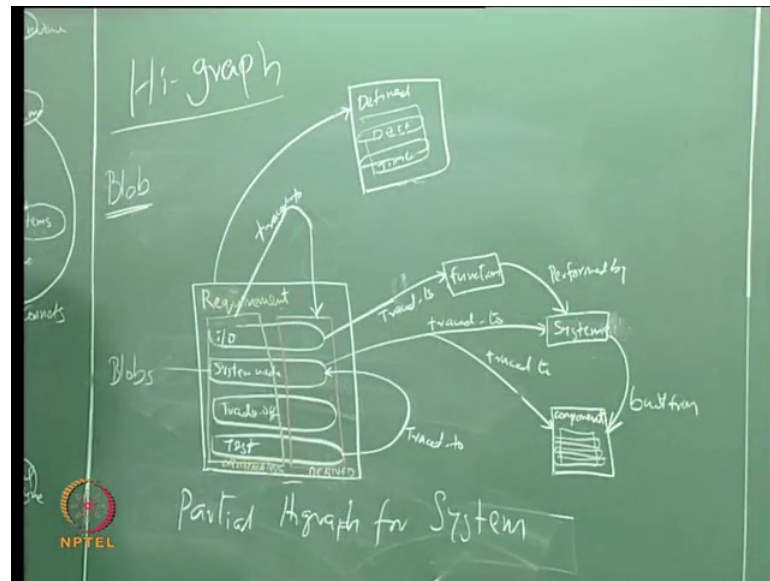
First we can see here this is a complex diagram and it is bit difficult to represent all the entities using this method therefore, there is another method called high graph.

(Refer Slide Time: 40:05)



So, this is actually an extension of the complex ER diagram. So, this is known as higraph.

(Refer Slide Time: 40:10)



Which is widely used nowadays for representing the a complex relationship between entities. So, the higraphs are generalization of complex ER diagrams, here then entity is considered to be a collection of many small elements or a collection of multiple elements which are known as blobs. So, one important terms you use here is the blob. So, here a single entity is considered to be a collection of small elements in the system.

So, that is the blob multiple elements are bind together in a blob. So, here we represent the higraph using a and one entity will be representing many elements. So, if you take this as an entity one entity. So, we will represent the entity with many collections of elements called blobs. So, these are known as the blobs. So, one entity or a collection of many things can be considered as a single entity. So, as you can see here there are many entities. So, some of these entities can be combined together to make a single entity and this collection of the small elements will be known as blobs. For example, if you take the entity here as the requirements. So, instead of this entity if I consider the requirement, here you can see requirement is one entity which actually consists of many subclasses.

So, probably we can actually represent all the subclass into a single entity, and then represent requirement as a single entity showing all these subclasses. So, that is the blob. So, here instead of this entity I can actually write requirement as a an entity here. So, requirement as an entity and then I can have many requirements over here because I can have originating requirements as well as derived requirements and in originating

requirement itself I can have. So, this is the originating requirement and this is the derived requirement and here we can see that the originating in derived requirements are here these are the derived requirements and the originating requirements are here.

So, we can actually represent all of them into a single entity, and here we can see this is the input output requirement and this is the system wide requirement then trade off requirement and then test requirements. So, now, what we have done is that we have combined all these elements. So, many elements have been combined here to form a single entity and this combination of the elements on known as the blobs. So, we have many blobs combined into a single entity and we are getting a single entity is represented by this. In the higraph we try to simplify the complex ER diagram using this kind of blobs and entities. So, here you can see input output requirement has got originating requirement as well as derived requirement.

So, both are represented using the same diagram. So, this input output is common for both originating as well as for derived similarly system wide requirement can be originating requirement or derived requirement, trade off requirement can be originating or derived similarly test also originating or derived. So, all these requirements are combined into a single block and this is known as the one entity in the higraph. So, that is the difference between higraph and the complex ER diagram we represent some of the is a relationship. So, most of this is a relationship can be combined to this one to represent the entity. So, these are known as blobs and this is the entity. Now if you make a the similar diagram of the system partial system diagram can be shown using this here we can write the relationship; so the derived requirements.

So, these are the test requirements can be. So, this is suppose this is a test requirement and the test requirement can be traced to this one. So, we can actually write this relationship as traced to; similarly the input output requirement can be traced to we can write the arrows to the functions. So, this is another entity. So, this can be traced to functions and then the functions are performed by this is the system this is the relationship is performed by the functions are performed by the system and then the system wide requirements are traced to the system.

So, now you have the system wide requirement since can be traced to system and the system is built from this is built from components. So, components can be another entity

with many blobs. So, we put it as components. So, again you can have it as a entity with many blobs. So, this is a built from. So, a similar entity can be identified for this also components and then this can be traced to the system wide requirement. So, the require and all the requirements can agree ability again be traced to this one components also. Of course, these two requirements, the originating requirement and can be traced to the derived requirements this is a relationship traced to and we can say that all the requirements are defined on a particular date or time.

So, defined and again this can be a block with many data like date time etcetera. This is defined on that particular date that is the relationship. So, this is actually a partial higraph partial higraph for systems. What we are trying to do is to reduce the complexity of the ER diagram the complex ER diagram is simplified to a another diagram called higraph where we try to see all the subclass relationship and all the subclass relationships are combined to a single entity and this single entity will be having multiple blobs to represent the subclasses in the system and then we can actually have many entities like that and the relationship between the entities can be represented in the same way as the ER diagram. So, this is the higraph which is very commonly used for representing the relationship between entities.

So, to summarize whatever we discussed today, we discussed about the qualitative modeling techniques graphical modeling techniques for engineering system design. As we know there are many entities in the system and there are relationships between these entities and therefore, we need to represent these relationships using graphical methods. So, that it is easy for us to understand the relationship and it will help us in the design of the system. And various methods are used the data model is one method which actually represent the relationship between entities and in data model we have simple ER diagrams where actually the relations between two entities can be represented as a one-to-one relationship or one to many or many to many relationship.

And then we have complex ER diagrams where we try to represent the subclasses using a relationship called is a relationship and all other relationships are directed or directly represented using directed arrows with the corresponding functions. You know to simplify the complex ER diagram what we are doing is to combine some of those subclasses, the is a relationship in the complex ER diagram is combined to make blobs and many blobs are combined to form a single entity. And we can have multiple entities

like that in a complex ER diagram and the relationships can be represented using in a higraph. So, the three methods we learned today are the ER diagram the complex ER diagram and the higraph.

So, these are the basic methods for data modeling used for system engineering. The other two methods are process modeling as well as the behavior modeling these two we will discuss in the next class.

Till we meet goodbye to all of you.