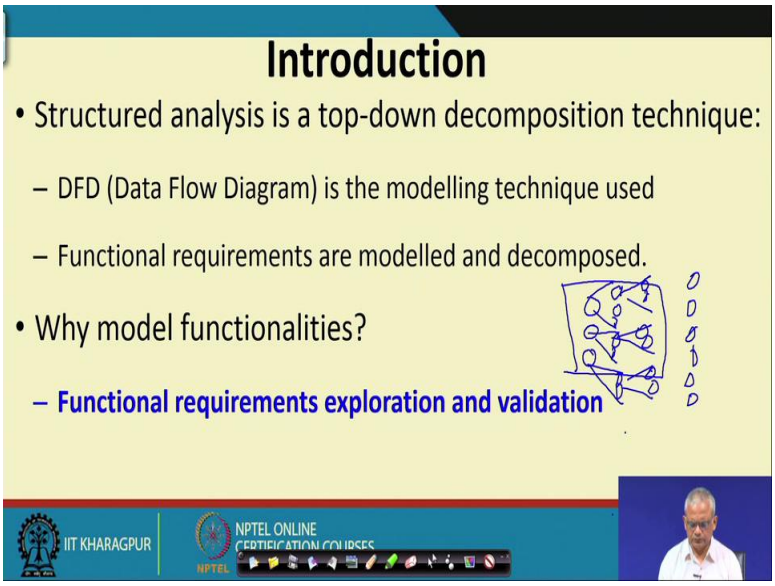
Software Engineering
Prof. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 23
Introduction to structured analysis and structured design

Welcome to this lecture. In the last few lectures we had looked at requirement specification and we saw that, that it is a very important issue. Before starting to develop it's very important to know the exact customer requirements, find out if there are any difficult is with the requirements and then eliminate them and finally, document that in an acceptable format. And we had seen the IEEE 830 format which is accepted across all industries and academics. And we had seen the different elements of the requirements specification, the functional requirements, non-functional requirements and so on and then how to document this.

Now, let's start looking at the structured analysis and design. We had also seen some very basic concepts about design namely the high-level design, detailed design and also what can be considered is a good design in terms of modularity, cohesion, coupling, layering and so on. Now, let's start to discuss how to go about doing procedural design of a problem. We assume that the requirement specification is done to complete documentation and we would like to start doing the procedural design for the problem. Let's see how we go about and address this issue. In this lecture and next few lectures, we will address the topic of Structured Analysis and Design.

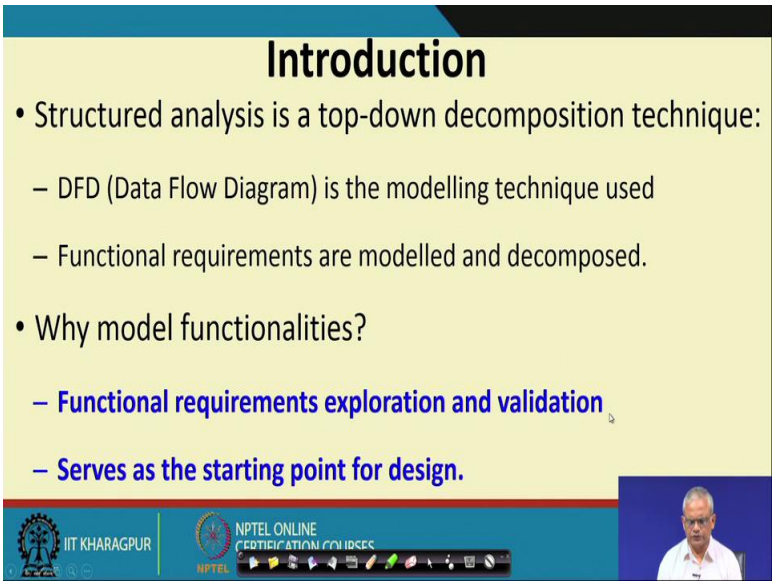
(Refer Slide Time: 02:25)



Structured analysis is a top down decomposition technique, in the sense that we will start with very gross or course level requirements that is as specified in the SRS document. The functionalities that the system should perform we start from their and then we do a top down decomposition; meaning that we break the required functionalities of a system into finer levels. And for this purpose, we will use the data flow diagram technique. We will use the data flow diagram (DFD) technique to do the functional requirements analysis. We will model the requirements and we will decompose into smaller functions.

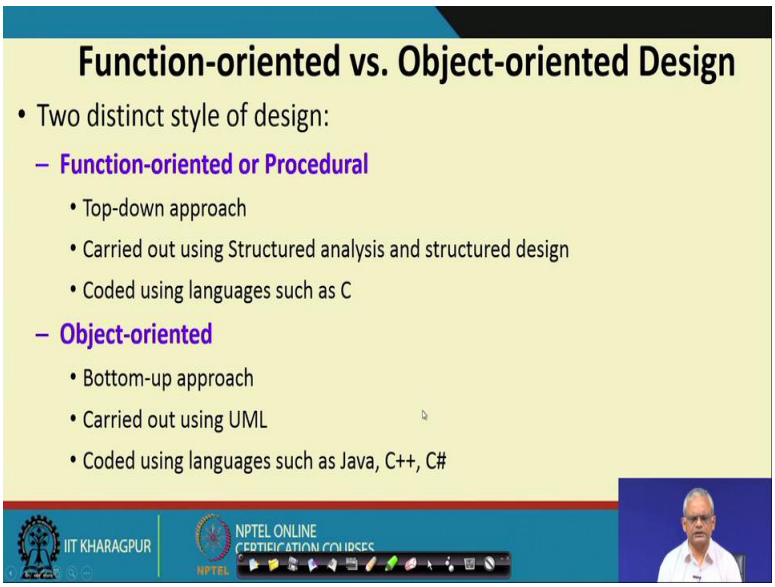
But before we started it's good to know that why we need to model the functionalities? We need to model the functionalities because we will do the requirements exploration and validation. By this we mean that the functional requirements as specified in the SRS document will take exactly the same functional requirements and we will break them down into simpler functions in a hierarchical manner. What we mean by that is that if we have SRS document for our system and we have in the functional requirement section (Various function listed here), then we will take each function and we will use DFD to model this system. We will represent all these functions and also, we will decompose this into finer functions. Each of the functions will be decomposed. Sub functions also decomposed through the DFD technique and finally, we will have a set of fine granularity functions. So, this we call as the functional requirements exploration because we starting from very gross level requirements specified in the SRS document, we decompose them into functions which are very small in size.

(Refer Slide Time: 05:26)



Once we achieve this decomposition, we will be ready for starting the design. So, in the structured analysis we do the functional requirements exploration that is break the functions into fine functions and then once we have the fine set of functions, we then start the design.

(Refer Slide Time: 05:55)



But before we start looking at the structured analysis and design, we will do a small digression. We will contrast the function-oriented and the object-oriented styles because right now, in this lecture and next few lectures, we will be looking at the function-

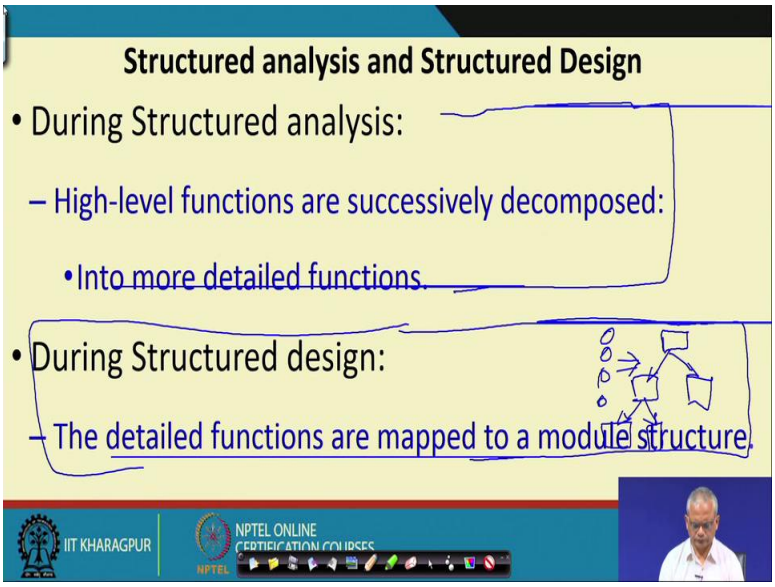
oriented design and later we will be concentrating on the object-oriented design. function-oriented design and the object-oriented design are two distinct design styles. The function-oriented or the procedural design is a top-down design technique as we are saying that we start with the requirements, take the functions, and decompose them hierarchically into a fine set of functions.

After the decomposition, we map them into a design structure and we use the structured analysis and structure design for the procedural design technique. And then once we have got the design then we can easily code it using a language such as C.

On the other hand, object-oriented design technique is a bottom-up technique. Here based on the requirements, we first identify the objects and then we see the object relations if there are any, then we model the object relations to build a hierarchy.

So, we start with the basic objects and then we group the objects into classes and then classes into an inheritance hierarchy and so on. So, this is a big difference that function oriented is very intuitive top down approach and object-oriented is a bottom up approach. In object-oriented approach we start with objects and build from there and we will use the UML notation to do the object-oriented design. Once we have the design ready then this can be easily coded in a language like Java, C++ or C# .

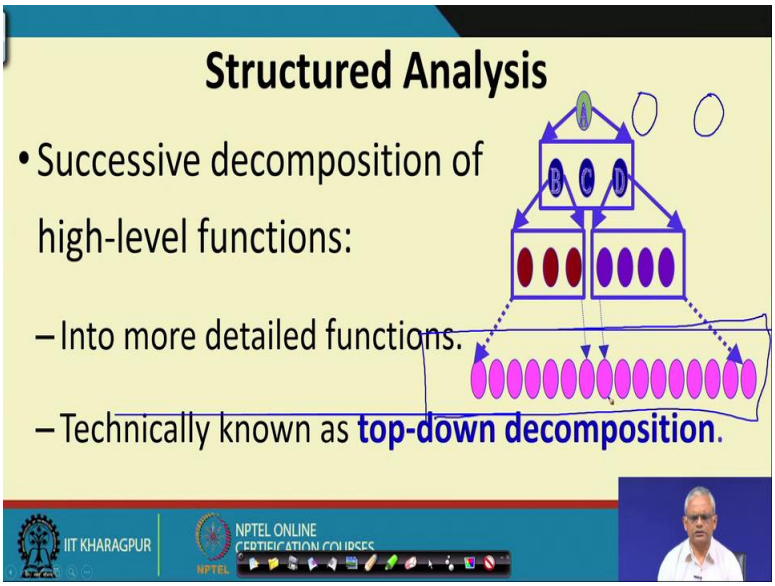
(Refer Slide Time: 08:35)



So, now let's get into our discussion on structured analysis and design. So, here as we are saying that we will start with the high-level functions as specified in the SRS document and then we will decompose into detailed functions. This we will call as the structured analysis. So, this is the structured analysis activity where we do a functional decomposition.

On the other hand, immediately after the functional decomposition we will start the structure design and here we take the functions identified in the structured analysis. After identifying functions in the structured analysis, we will map these set of functions into a module structure. We will discuss a technique called as a structured design technique, which gives a methodology how to map these set of fine level functions into a module structure and this will form our high-level design. We will first discuss the structured analysis where we identify the fine-grained functions and next, we look at the structure design, where we discuss, how to map the fine-grained functions into a module hierarchy.

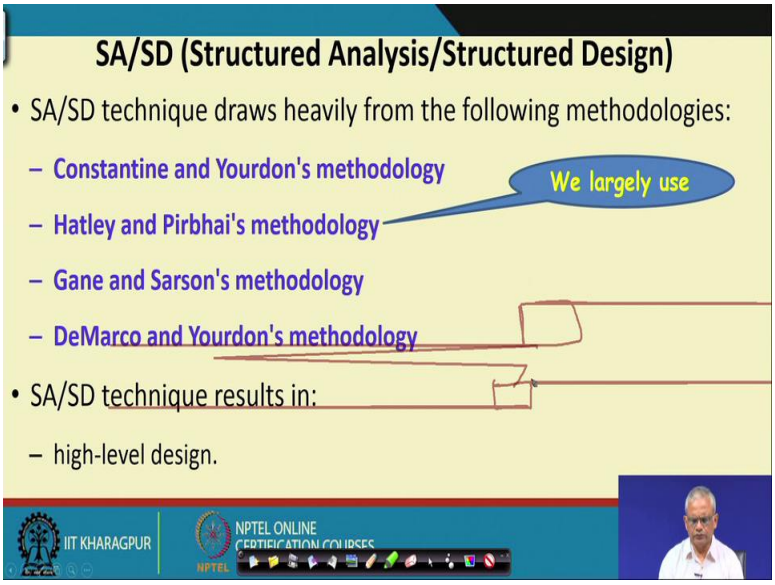
(Refer Slide Time: 10:28)



Now, for the next hour or so we look at the structured analysis where we do a top down decomposition starting with a function identified in the SRS document, we split into sub functions. This diagram (On above slide) just explains this concept. From the diagram, we can see functions are decomposed into sub functions.

For decomposition we will use the DFD. We will use DFD notation for modeling decomposition. And then each sub function into further sub functions and same thing will do for the other high-level functions mentioned in the SRS document. And at the end we will have the set of fine-grained functions and this we will mapped into the design structure.

(Refer Slide Time: 12:00)



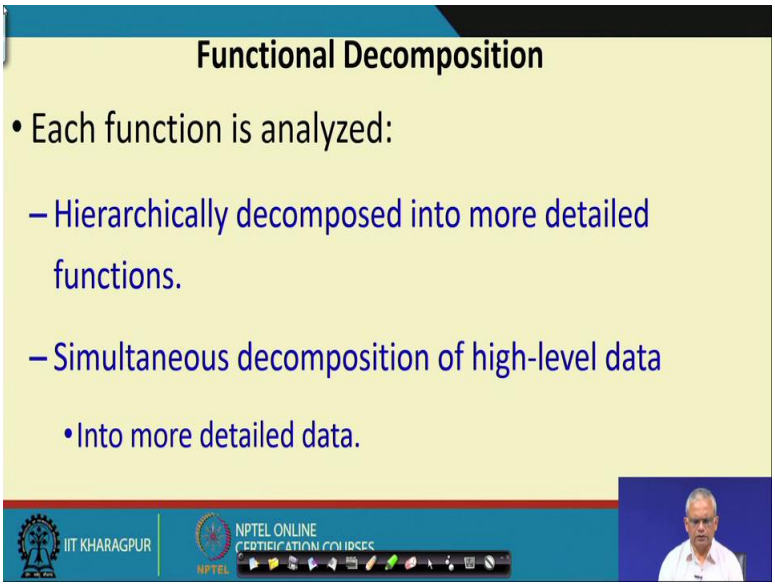
The structured analysis and structure design technique is well accepted by the developers. Anybody wanting to do a procedural design would do the structured analysis and structured design, but then there are various flavors of these design. For example, someone may use the Constantine and Yourdon's methodology or Hatley and Pirbhai's methodology or Gane and Sarson's methodology or DeMarco and Yourdon's methodology.

So, the main point is that there is no standardization of the methodologies. We can find in different industries, using different methodologies. These are largely the same, but then there are small differences in the notations. For our discussion we will be using largely the Hatley and Pirbhai's methodology. We will see that the different tools that are available, they also support number of methodologies.

Hatley and Pirbhai's is a methodology which has come much after Constantine and Yourdon and Gane and Sarson's methodology. So, this Hatley and Pirbhai's methodology is a more recent methodology and obviously, they had the privilege of

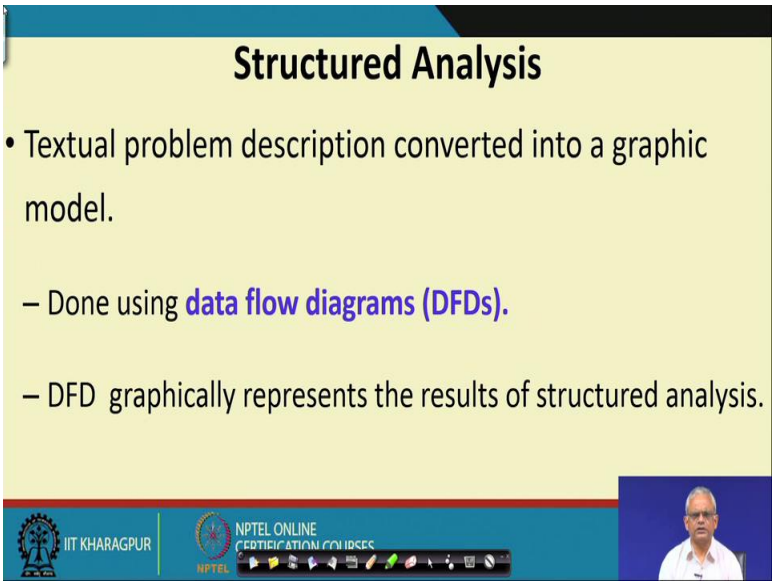
making use of the ideas on the Constantine and Yourdon, Gane and Sarson's, and DeMarco Yourdon methodology. In our discussion we will be using the Hatley Pirbhai's methodology to a large extent. Once we do the structured analysis and structure design, we will have the high-level design. In the high-level design we will have the module structure and in the module structure we can use this methodology for coding and the final implementation.

(Refer Slide Time: 14:17)



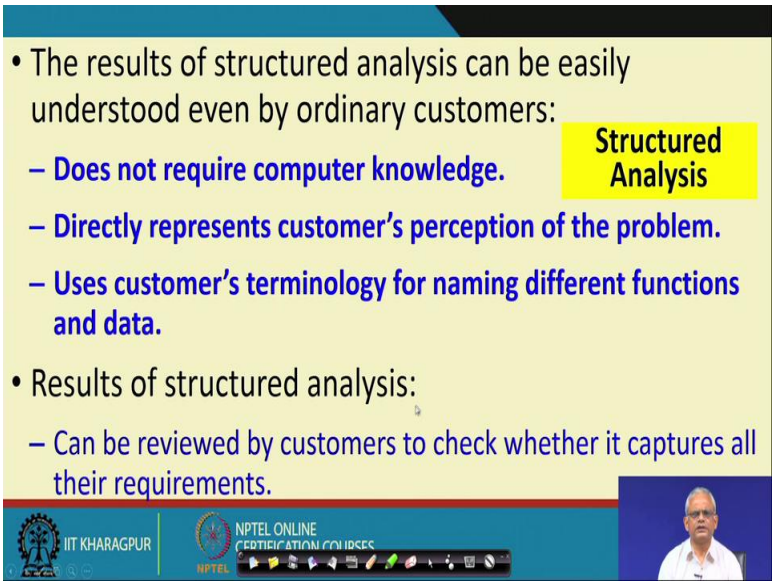
Now, let see how do you achieve functional decomposition because this is one of the main objectives of the structured analysis. We do functional decomposition through a hierarchy of levels where we decompose each function into sub functions and so on. And as we decompose the sub functions at the same time, we will observe that the data also gets decomposed. Starting from the very high-level data that are mentioned in the SRS document, we decompose that simultaneously and automatically into detailed data. And finally, we have primitive data elements at the lowest level in terms of integer, character and so on.

(Refer Slide Time: 15:20)



For doing the structured analysis, we will use the DFD technique. DFD stands for data flow diagrams and this is a very intuitive and simple technique. It takes hardly few minutes to get familiar and get productive and get started using this technique. And because of its simplicity and utility this technique is extremely popular not only in the procedural design, but also in many other applications. For example, in the SRS document itself in many projects they develop the DFDs because this gives a good understanding of the functions into a finer level of activities. And also, the DFD model is a very intuitive model and by just looking at it we get a good idea about what the system needs to do.

(Refer Slide Time: 16:35)

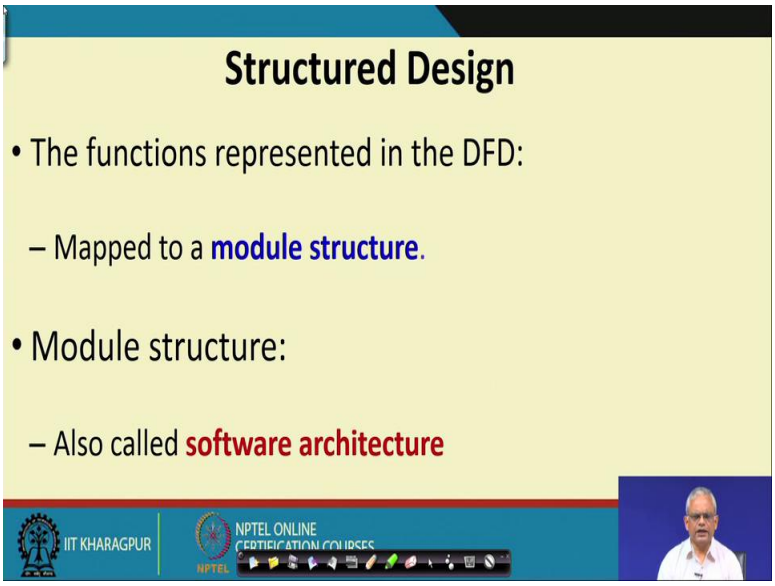


As we were saying that the structured analysis technique is a very simple technique, don't need to be even knowledgeable in computer; anybody can start using this technique. And it represents the customers perception of the problem. It's not really a design technique, it just keeps on elaborating what the customer wants of the problem. So, DFD is a detailed model of what the customer really wants of the problem and for this we also use the customers terminology for giving various names to the data, to the functions and so on.

So, this DFD technique is easily understood. The model's developed using the DFD are easily understood by the customer, even though customer may not have computer exposure. Therefore, once the designers they come up with the DFD model of the system, it can be reviewed by the customers to check whether it missed any of their requirements and whether it has correctly captured their requirements or not. We will learn this technique and start modeling various problems in a very short time.

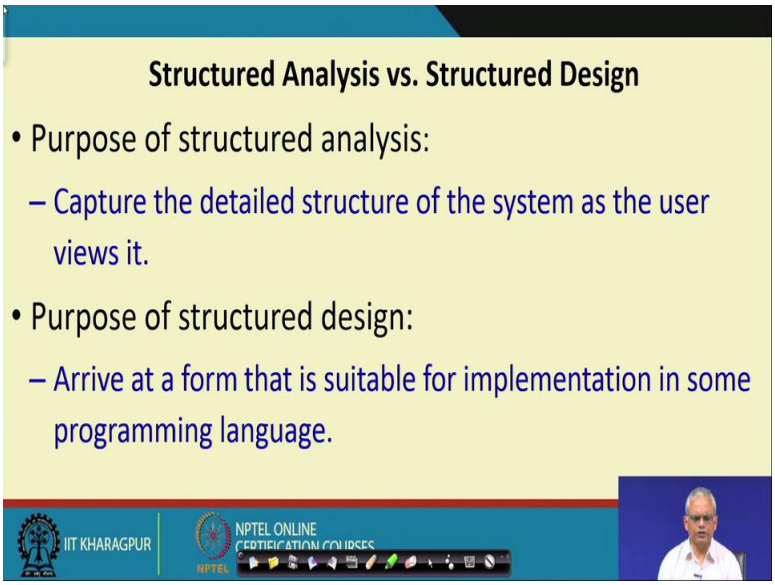
But let me just tell you that it is easy to learn this technique, but then for finally, using this technique to model various problems we need bit a practice.

(Refer Slide Time: 19:03)



Now, once we have completed the structured analysis, we will start the structured design. And here we will map the fine-grained functions into a module structure and this module structure is also called as the software architecture.

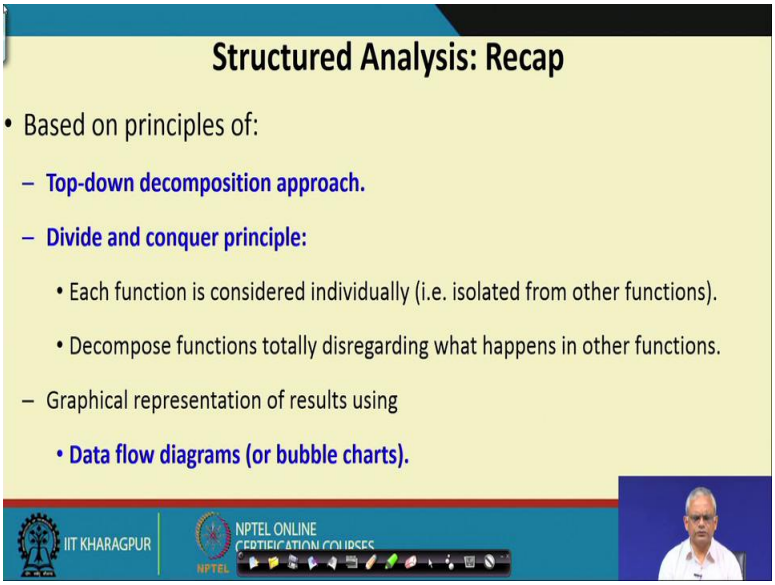
(Refer Slide Time: 19:21)



One thing we must be clear is that the structured analysis is just an elaboration of the things that the customer wants. On the other hand, in structured design we arrive at a form that can be easily implemented, that can be easily coded into a solution. Obviously, if we are done only the structured analysis we cannot derive a code from that. But once

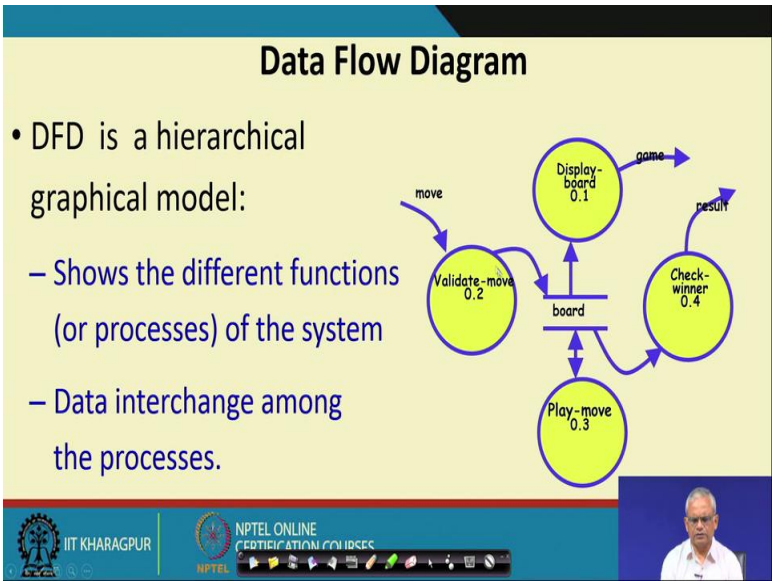
we have converted the structured analysis into a high-level design using the structured design technique that can be easily coded into the solution.

(Refer Slide Time: 20:17)



Now we will just recapitulate what we discussed. We already discussed that the structured analysis is a top-down decomposition approach. We know what we mean by top-down decomposition and it's a divide and conquer principle. It uses the divide and conquer principle because, we look at all the functions in the SRS document, but take each function one by one and then at a time we decompose one function. Decompose one function means decompose it into sub functions and again at a time we look at one sub function and again decompose it and that is exactly is the divide and conquer principle.

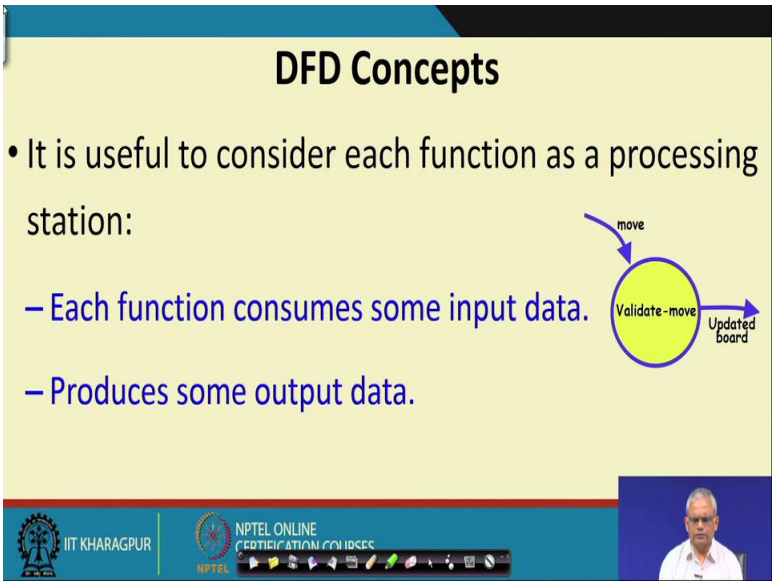
(Refer Slide Time: 21:14)



For this decomposition, we will use the DFD which is also called as bubble chart to represent our result. The DFD allows us to do a hierarchical decomposition and that is why the DFD is a hierarchical graphical model. We will have various levels of model. Some are very high-level model and then these are further decomposed into finer or more detailed levels. And at each level, we will represent the functions and also the data interchange among the processes. This is an example of a DFD model (On the above slide). In DFD, we will use the circles to represent the functions or the processes. Either we can call these as bubbles, processes or functions and then these bubbles or functions takes some input data produced some output data. So, it represents how the data flows within the system through the various functions.

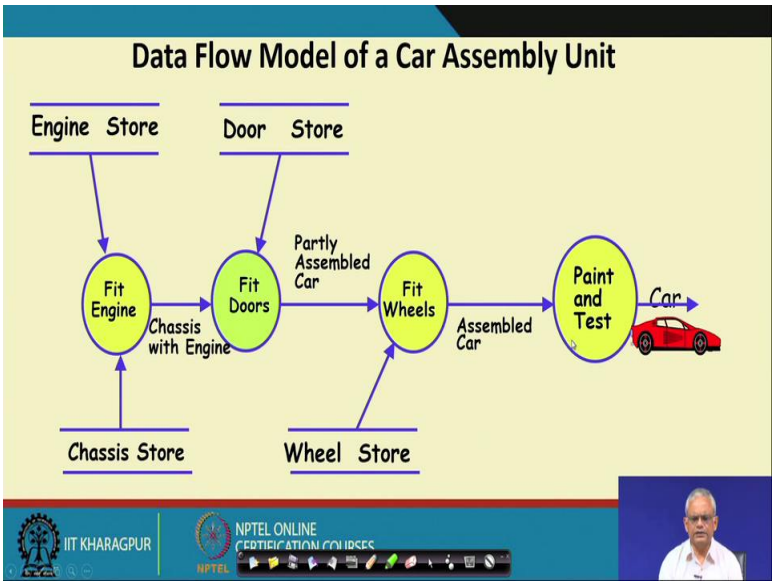
Please remember that this is not a control flow model, it does not say that which flow occurs first and then which one occurs next and so on. It just represents the data flow basically. It represents what are the functionalities at this level and what data they consume and what data they produce. We will see the other notations that are used here in few minutes.

(Refer Slide Time: 23:10)



As we were saying that each of this function is represented using a circle and this represents some processing. It takes some input data, does some processing and produces some output data.

(Refer Slide Time: 23:37)

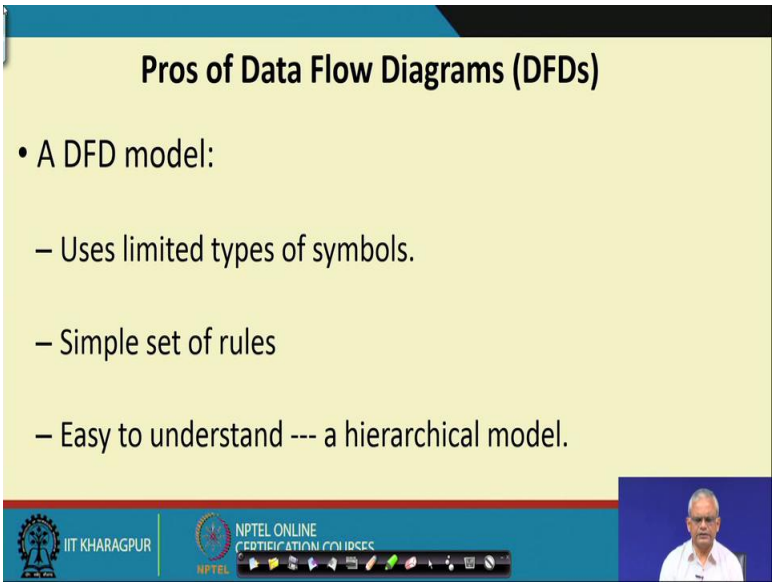


The model is extremely simple, we can easily understand a model developed using this. Now let just look at this car assembly unit modeled using the DFD notation. From the above slide, we can see the processes are fit engine, fit doors, fit wheels, and paint and

test. We can observe here that each of the process represents some activity and that's why these are starting with the verb form fit engine, fit door, fit wheel, paint and test etc.

So, every function represents an activity or processing and is named using a verb form. And then each process represents by bubble here. All these process takes some data as input. For example, 'Fit engine' process takes engines from the engine store, takes chassis from the chassis store and then fits the engine to the chassis. The chassis with the engine goes to the next processing station 'Fit doors' and here the doors are obtained from the door store and to the chassis with engine the doors are fitted. And this we call as the partly assembled car and then in 'Fit Wheels' the wheels are fitted by taking the wheels from the wheel store and then we have the assembled car. This is painted and tested in 'Paint and Test' process and the final output is the car.

(Refer Slide Time: 25:43)



So, the model is very intuitive, just by looking at the model we can make out that what really happens. It's a very simple technique, as we are saying. We will see that in very small amount of time, we can start doing the DFDs for various problems. One of the main reasons why it is very simple technique is there are only five types of symbols. We can learn the five symbols in no time. There have some set of rules by which we combine these symbols to model. We start doing something very simple and slowly add more details to it and therefore, while even developing a very sophisticated model we find that it becomes easy.

So, with this discussion about the DFDs structured and introduction to structured analysis, we will stop here. We will continue from this point in the next lecture. We will see how to use the DFD technique, what are the nitty-gritty of the DFD technique and how to use this to model any given problem.

Thank you.