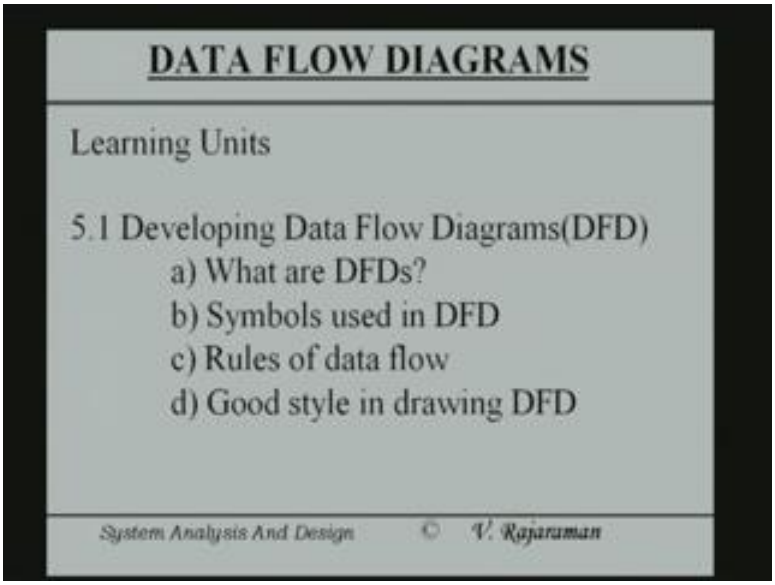
System Analysis and Design
Prof. V. Rajaraman
Department of Super Computer Education and Research
Indian Institute Of Science, Bangalore

Module - 05
Lecture - 15

Last time, we completed the discussion of feasibility analysis. This lecture is going to be on great dataflow diagrams.

(Refer Slide Time: 01:21)



In fact, we had earlier did mention this dataflow diagrams and we did use them in some situations. But, we did not go in to the great details, of how to deal with dataflow diagrams in detail. The data flow diagrams are important tool for a systems analyst and designer. It is something like drawing, which an architect uses the advantage of a drawing is on one hand, it should be understood by the person who is going to implement it namely the engineers and so, on. And also by the user who is going to ultimately live in the building.

So, the plan or the drawing of the house is something, which is useful for two different classes of people. Similarly dataflow diagrams or something, which is going to be of use to the systems analyst to be given to the system designer, on one hand and to the user of the ultimate system on the other hand. So, in order to be able to come up with the tool of

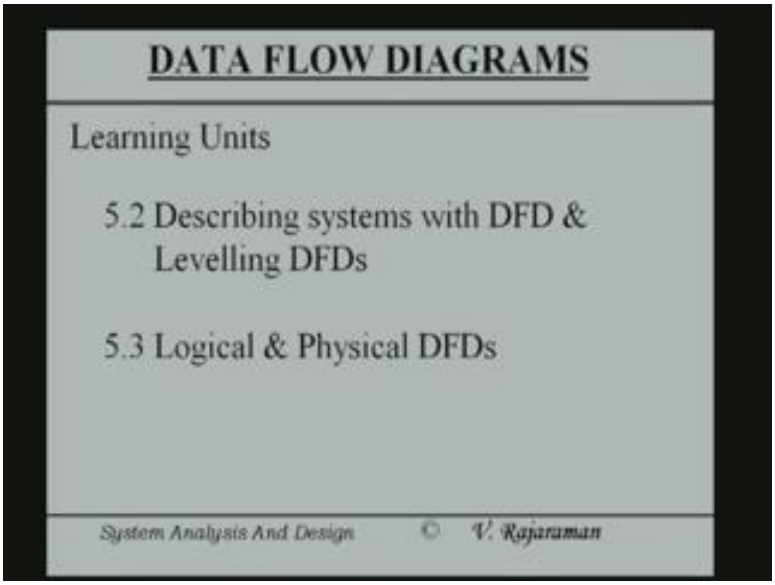
this type we need have some characteristics of this diagram, which is easily understood by both parties.

And it is graphical tool, just like a drawing is graphical tool. And the advantage of graphical tool or there it is more or less self evident, what they really imply. So, the various learning units in this particular module or what are dataflow diagrams. The dataflow diagrams are abbreviated as DFD's. I mean is just for saving time. I would say just turns out in the computer science, we seem to use a lot of acronomics. And even though sometimes I do not like some of the acronomics, you do not understand, still it is a convenience.

So, in this context what I mean by DFD is a dataflow diagram. So, we had to first define what dataflow diagrams is. We, had to look at the symbols used for in the dataflow diagrams rules, which are there to draw these diagrams and not only are the rules important. But, something called a style, styled something which is objective. But, still style is important because, just like when you write English, the grammar should be correct.

But, the style is something which you make a difference, something which can read easily. And something which is not easily read, other words easy on the read you might say style is extremely important. So, the book style is what we are going to be emphasizing.

(Refer Slide Time: 04:53)

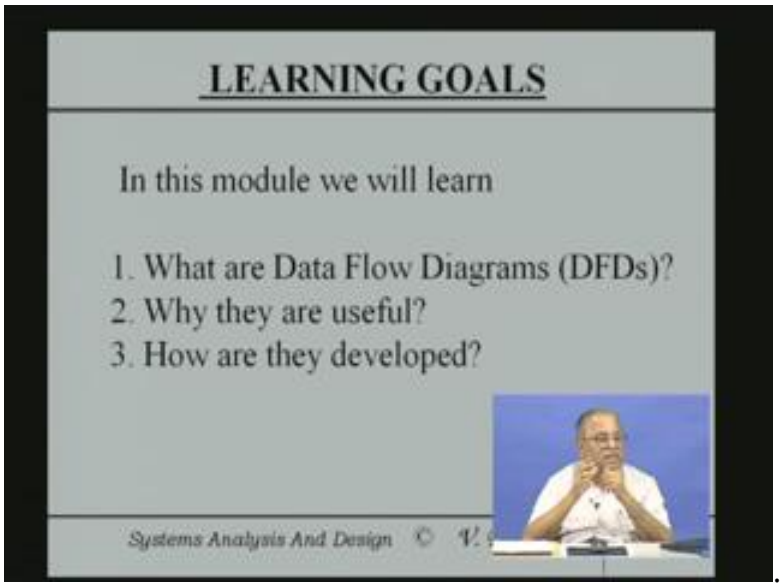


We will describe how to use DFD's for large systems and something called levelling of DFD's. I will describe levelling in more detail. The whole idea levelling is that when you make something very, very complex. And very detail people will find it difficult to absorb, what they really imply. Whereas, if you break it up the smaller parts, then it is easier to understand.

So, levelling is nothing but, breaking up something which is too complicated into something which is you might say hierarchical. Starting with something as an overview, and breaking up into something into smaller parts. So, each part can be explained in greater detail. And there are two types of dataflow diagrams. One is called a logical dataflow diagram, which is primarily for the logic of the program they are going to implement and so on.

And physical dataflow diagrams, which primarily say what physical documents, what physical goods and items and so on, flow in the system. So, that is a lot of difference between physical goods and the equivalent of physical goods as a document telling giving a description. So, dataflow diagram in the logical part, only deals with the description.

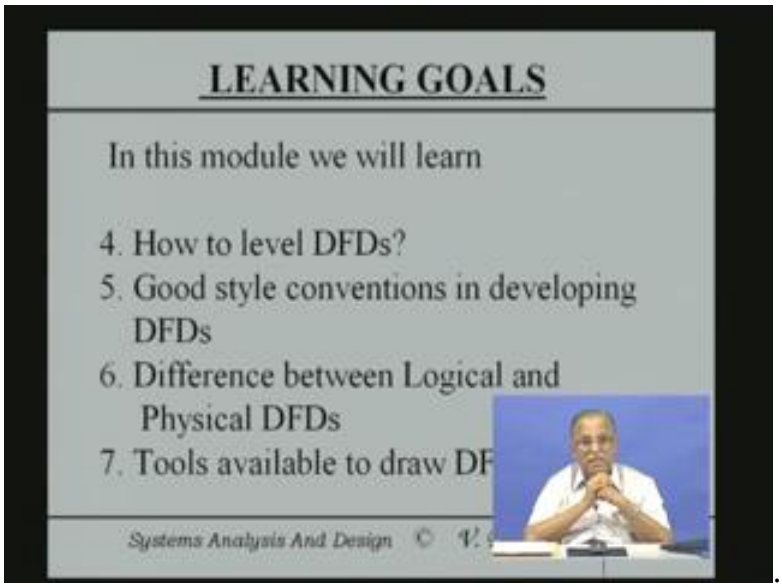
(Refer Slide Time: 06:24)



.

So, again to emphasize learning units, lot of data flow diagrams, why are they useful, how are they developed.

(Refer Slide Time: 06:34)



How to level DFDs, good style conventions in developing DFDs, difference between logical and physical DFDs and tools which are available to draw DFDs. I will not go into great details about the tools. Because, the tools or something which it make it easier, to do the drawing of these things. And of course, as you know today in computers many drawing tools available. So, those drawing tools can be used and it is not a part of this course, talk about how to use paint, brush, draw and things like that in computers.

But, there also other tools available, to check the logical correctness of DFD's. And this is some of the details about this is given in the written notes, which accompanies this lectures particularly sites, which are there where you can look at download. Some of the cheap or no cost tools, which a student can use because there are tools, which are made by professional companies and they cost a lot of money. And for students and for colleges to own them is expensive. It is always alright to get an idea of what these tools mean, how they are used from the by using the, so called open domain tools.

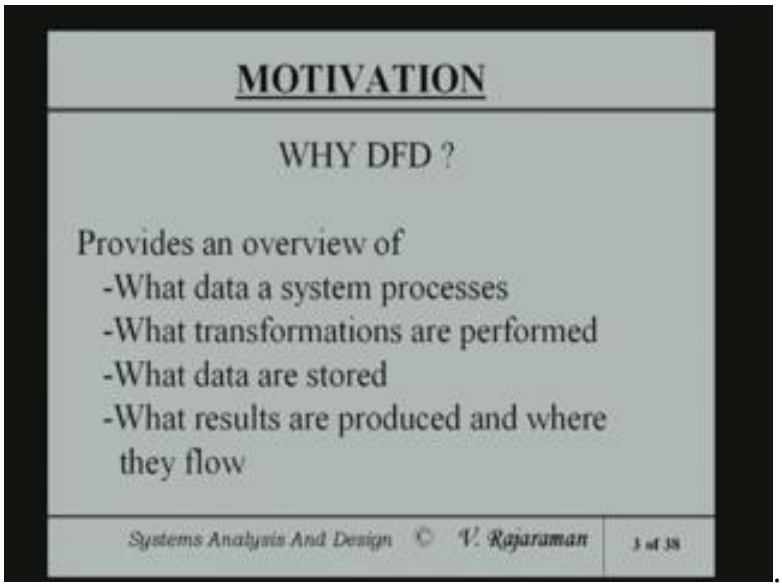
And that will be, the websites will be given in the lecture notes. Apart from the lecture notes and the PPTs, which accompany this lecture as the written part of this course. I am also using this as you know, this book analysis and design of information systems. And I

am essentially going to cover chapter 7, which is data flow diagrams, which is about the entire lecture is about.

A book has of course lot more details, that what I can really hope to say in one or two hours. So, one should actually go back and read the book also, after listening to the lecture, that will be better before you come to the lecture. But, I know that most students do not do that. Even it read after the lecture, it is good student who does it and many students do not worry about it.

In any case, it is important to be able to have a support material. In this case, support material is in terms of the web support material, which is part of this program. And also I have talked about the availability of text book, which is going to be closely followed by me.

(Refer Slide Time: 09:43)



The question of why should we use DFD. I already said something about this and essentially provides an overview of what data as a system processes. And that means, what are all the different data elements, which are essential to get the reports, which have to be generated by a system reports. That I mean by operational reports for tactical management and reports of strategic management.

So, all of this as I have been always emphazing. All data processing systems are based on some raw input data being processed by certain kind of processing rules. And creating

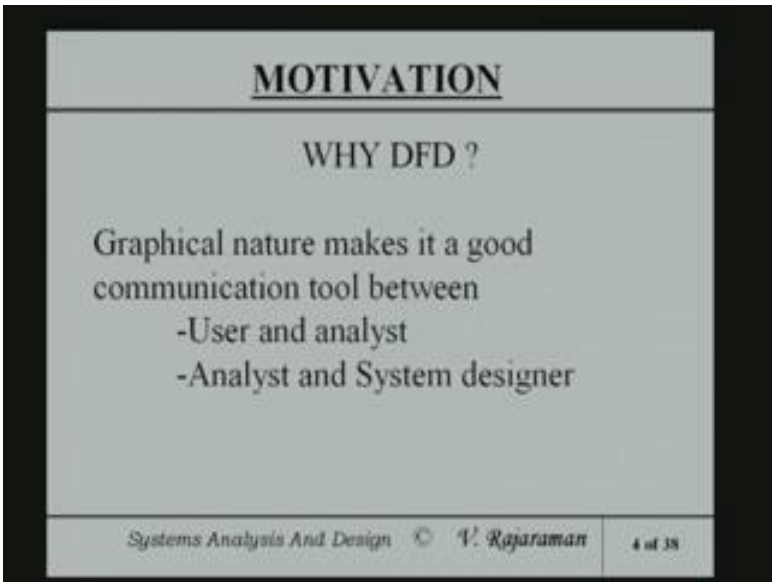
appropriate operational tactical and strategic information. So, the processing of data converts the data into information, which can be for used for taking and formed decisions. So, first of all you have to say what are all the inputs. Let us say what are the data which is processed by system. What transformation that are performed as transformation as what leads to your information.

And we have to describe this transformation and what data are stored in the system. In order to transform the data, some will flow from outside and some will be stored. For instance, if you are processing student examination results. And you already would have stored all details about the student, in terms of roll number, name, may be college. And all the information, all that will be there, in some kind of a file. And the results are the ones which are processed, when the marks arrive you process with these marks, add them. And then you find out the class, then you enter it and then print out the file.

So, in other words the point is that the file is already ready and kept then the last minute time taken to process will be much smaller. Otherwise you start entering everything at the last minute. It will not be very, it will take unnecessary long time you already keep a lot of things, which are not really going to change and keep it in files. Of course you read from the file, you process data from the file based on some inputs and you write back in the file.

So, what data has to be stored, other words stored data is normally called a file or database. In fact, data base is better word terminology to be used and later on in this course will look at the databases. Now, to organize database systems, in this particular part we only assume. But, there is a store for the data and what results are produced and where do they flow. The ultimately there must be consumer of whatever processed information you have and where do they flow.

(Refer Slide Time: 13:32)



As I have said, the DFD is a graphical nature and it is a, because it is a graphical nature. It is a good communication tool between the user and the analyst and system designer. See, just like an architect deals with two sets of people one is a user. Who is the ultimate person, who is going to live in that house. So, he has to get the approval of the user about the plan and things like that.

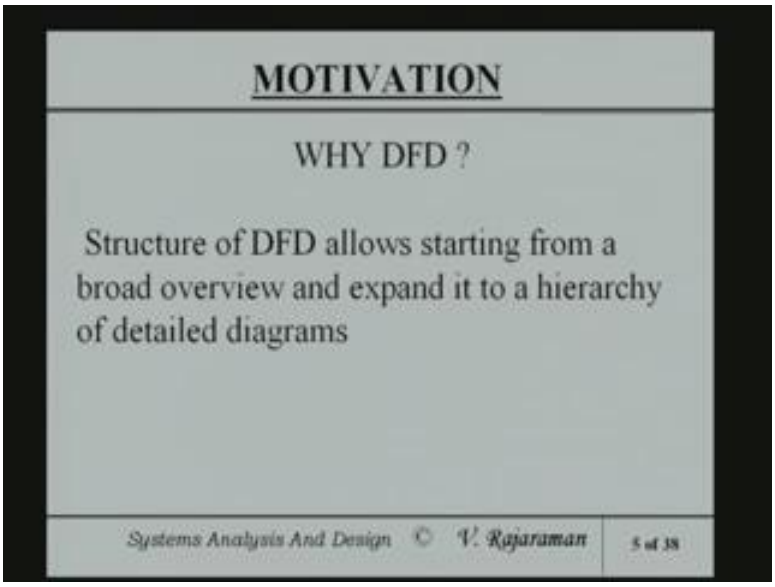
And of course, the structural engineer and designer, who has to actually construct it. Similarly, system designer somebody is going to write the programs, write the clear the databases and all the litigated details of processing the data to produce results. As far as user is concerned, the user is only concerned about in this case. What data needs to be provided and what kind of process results will he get. And with the data which has been provided is something, which he can easily provide or you should not asked, which data cannot be provided.

So, at that stage the user has a role to play and similarly the information which has created should be of used to the customer. So, in other words it is all based on the system requirements specification, which will effectively tell what results of the system would be or the process information will be which will satisfy the requirement of the user. Once this overview is given to the user then from this overview. The designer has to set as I said litigating details are writing the program and creating the database. And checking

out the program, testing it. All that goes with creating the system, it is a very long process.

But the process of creating a systems, SRS that is systems requirement specification as I said is very, very important, because if you start changing them in the middle then the designer will have a lot of difficulty. So, that is in any case, the point is that there is two sets of people have to use it. So, the tool must be such that, it should be usable by both.

(Refer Slide Time: 16:16)



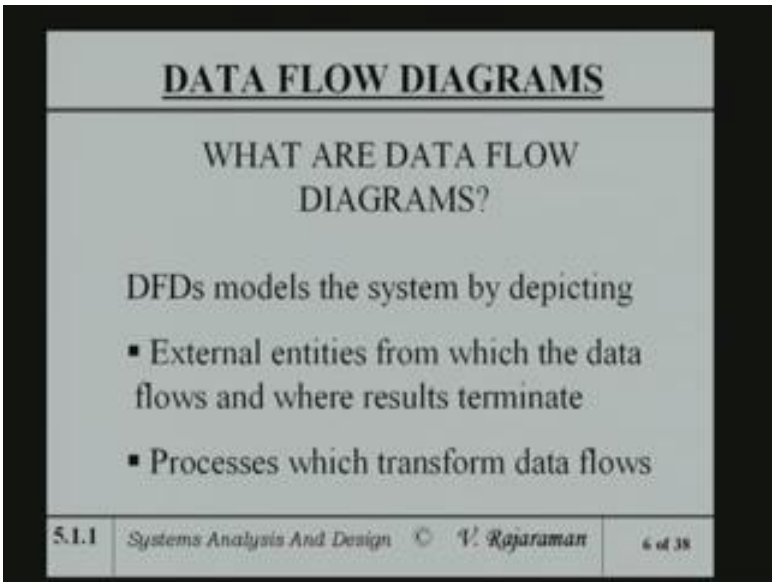
So, the structure of the DFD are going to design is to starts from a very broad overview and expand it into a hierarchy of detailed diagrams, I said that earlier. But, effectively it is like you know if you have a huge building to plan. You start with the overview of the plan, how it look like, you might even give a picture of house and things like that. But then when you go out each part of the house, you have to do a greater birth of design.

For a simple house, for instance a house will consist of many living room, kitchen bathroom and all that. And so the overview is just the plan, during the size of the various rooms, where they are located and so on. Then next time kind of look at the kitchen. Say for instance and detail of the kitchen because, with all kinds of gadgets, which you use in kitchen. You must proper electrical points, you must provide the cupboards and you must provide a place for your gas stove, it should be convenient all those things.

Other words similarly bathrooms, where hundreds of varieties of bathrooms depending upon the amount of money you are willing to spend and size of the bathroom and so on. So, you gain detail out, the bathroom in their detail other words. Overview on this, the structure of the general layout of the buildings. And then the details give each room, what they will be and details can go to the extent of telling the electrical points. Where, I have to be kept and at what height and what type of paint to be used of like that.

I mean, is like lot of like a other details which goes in to that. Similarly in a DFD you have a overview of structure of what the entire system would look like. And then each step you got to kind of expand it and then I will show it by example. For house example, it is very obvious that because all of you live in houses. Whereas, in this case you got to learn, how to look at an overview, and how to divide up into smaller parts.

(Refer Slide Time: 18:42)



And DFDs module systems by depicting external entities, there are number parts to a dataflow diagram, these parts are called external entities and processes, there are files and there are data flows. So, there are numbers of different what I would say elements we have to really learn, what the elements mean and why the elements are required. So, external entities are the ones from which, data flows and in which data terminates.

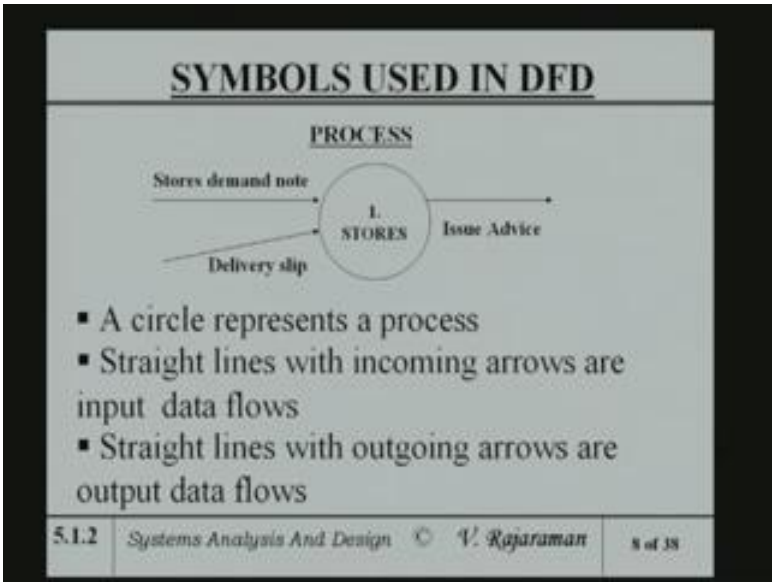
So, the originate of the data is called an external entity, the reason why it is called a external entity it is not part of the entire process, the person who is going to provide you the input data process like vendor, vendor an external entity. Vendor also may be a, the

data can flow from in terms of the invoice, which he sends and data also flow to him as I check for payment which is made.

So, the same external entity can both send data and receive data, but anyhow they do not generate any data of their own, what I mean is they do not process any data of their own, they only send data and receive data. And the process which transform data flows, process are which one they are allowed to actually transform it by in the case of computer and so, on transformation by means of programs.

Programs are the ones which are take the input data and process them, there are data stores from which the data is read or into which data is written as I have said, data stored I explained, why a data stored is required. Store is something you will store data, you retrieve data, you process data and write back data, write back process data you might say and the data stores are ones you read data from and write.

(Refer Slide Time: 21:26)



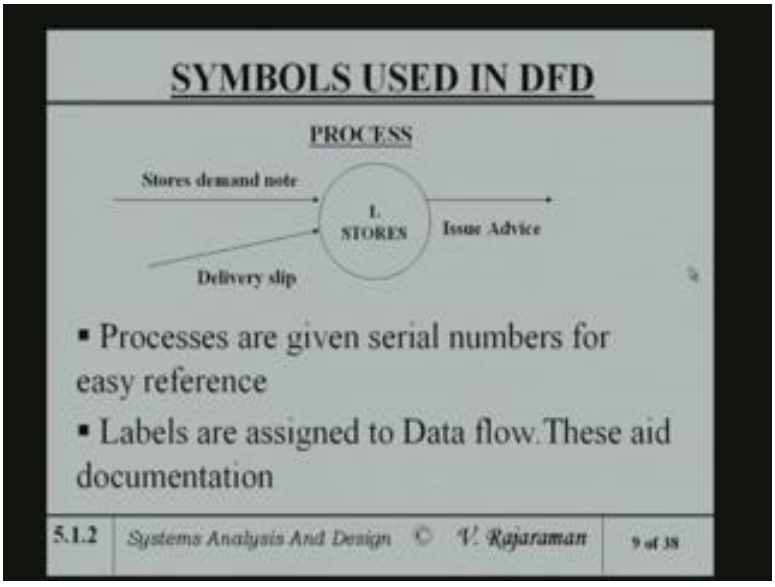
Now, as I said there are things I talked about as I said, the external entities, data flows, processing steps and stores these are the essentially only four items I mention. Now, each one of them has got a certain symbol in the notation for dataflow diagram, the symbol for the process is normally a circle, as you know the symbols to some extend are standardize. Standardize in the sense that almost everybody uses, but there are some variations in the symbol.

So, using a circle some people use in an ellipse and normally, it is a circle easy to draw and circle easy to kind of understand and so on. So, by in large circles is what I am going to use in and most standards use circles. And data flows are lines with arrows, which can be flowing to the process or flowing out of the process and in this case, the stores will being demand note and store demand note for the stores to be able to issue the items.

And the store also receive deliveries, so there are two data flows are showed here, one is the demand note of a customer or a user of the store and delivery slip and an issue advise, which is the after this issued along with it there is a document which is sent. So, these are data, the input data flows and output data flows. A circle represents the process straight lines with incoming arrows are input data flows, straight lines with outgoing arrows are output data flows.

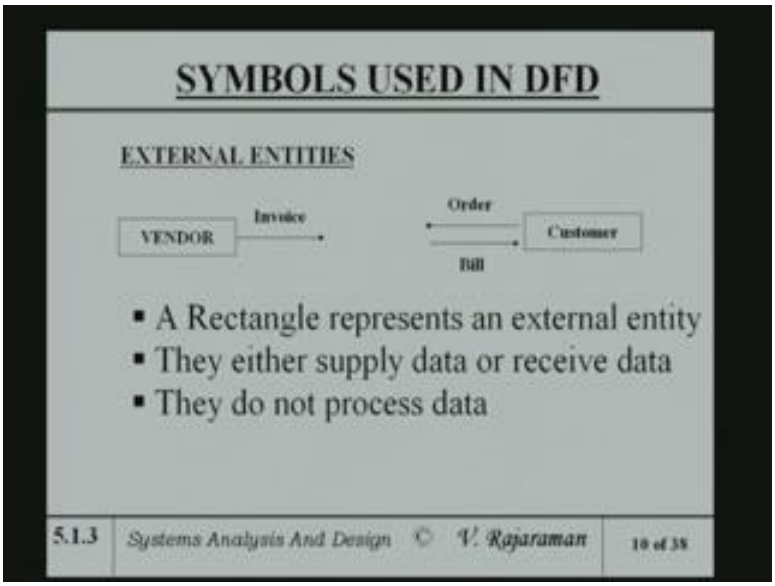
And as you seen here, the data flows there is some kind of a write up above it or below it, this essentially gives a label for the data flow. What the data flow represents, I am not giving the great detail about the label will be detail in later on, but one thing just says that something called the demand note is going to go in to the stores.

(Refer Slide Time: 24:12)



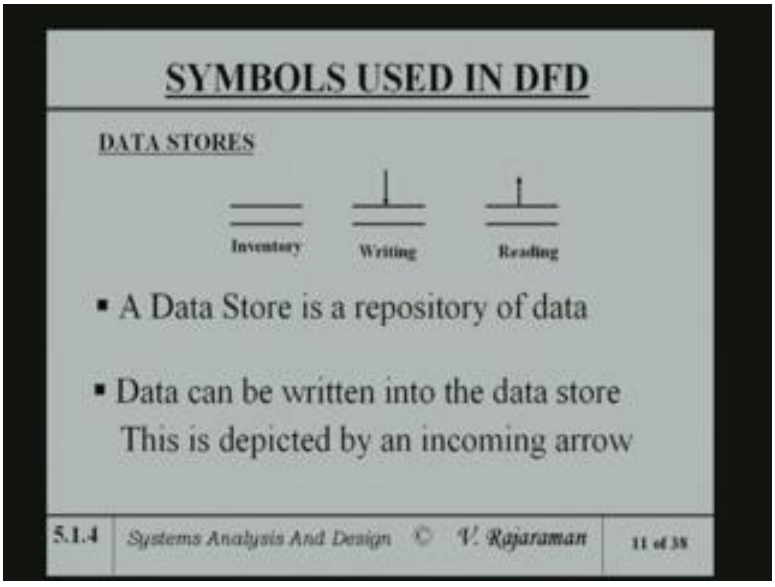
Processes are given serial numbers for easy reference, for of course, the number 1, I put there arbitrarily. I mean serial number is starting with some number because later on of you want to identify it, serial number is useful, labels are assigned to dataflow these aid documentation, primarily the numbering and the labels are for documentation.

(Refer Slide Time: 24:36)



External entities are represented by rectangles and they either supply or receive data as I said they do not process data. So, in this case, the entire the invoice which originate from him may be and customer sends around order and receives the bill. As I said same external entity can have arrows pointing inside as well as arrows pointing outside.

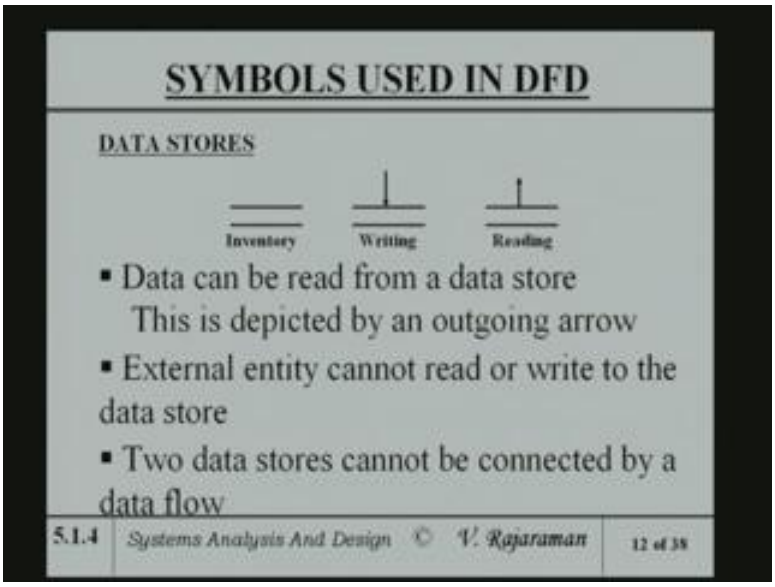
(Refer Slide Time: 25:12)



Now, data store is represented by two parallel lines. In this case, I said inventory for instance that data store is repository of data, stores data, one can write as I said and to

that repository or read from the repository. So, one can very easily say it is a file, I mean or a database, data can be written and just shown by an arrow, an arrow points inside that means, you are writing if your arrows pointing outside from the store that means, you are reading.

(Refer Slide Time: 25:53)



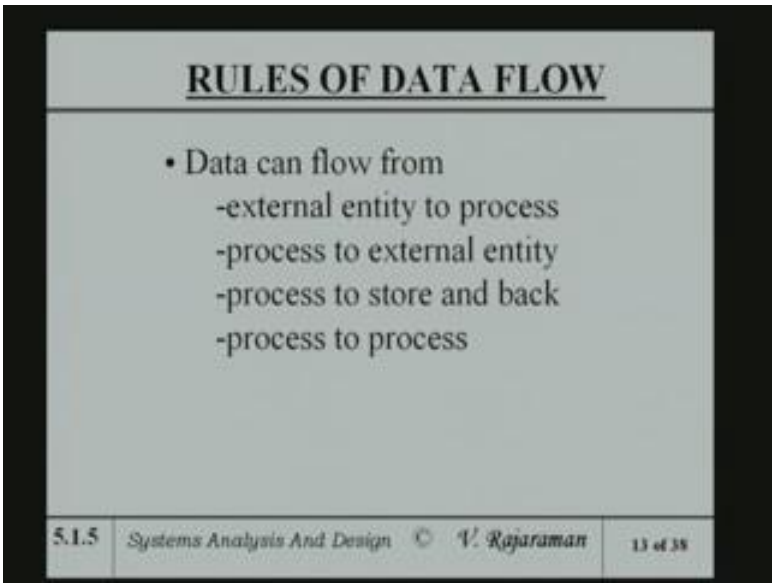
As I said, data can be read from data store depicted by an outgoing arrow, external entity cannot read or write in to the data store. In other words the data store is required for the process, because it required by the process. If you write directly into the file from outside that will not be you know that there is no process of writing, which is there in the entity, entity is just an you might say dumb, animal in the sense that entity is vendor and so, on.

So, there is question of entity being able write or read from a file, it got to be done only by a process. And two data store cannot be connected by data flow that is also obvious in the sense that in order to read or write into a file, you require some processor, which a process runs and that processor has the IO. And so the processor, the process gives the request to read or to write and so reading or writing of file can be done only by a process or processor.

Processor acting on behalf of a process, because by now you know the difference between a process and the processor, because in a processor many processes can be running, one of the processes which run in a processor is an IO process. So, you cannot connect to dataflow data stores because both of them are dumb files, there is only two

disk drives. And I can say, like how I can write from one disk drive to another disk drive without any intermediate, intermediate processor being available to take this, so you cannot, but you have to remember.

(Refer Slide Time: 28:26)



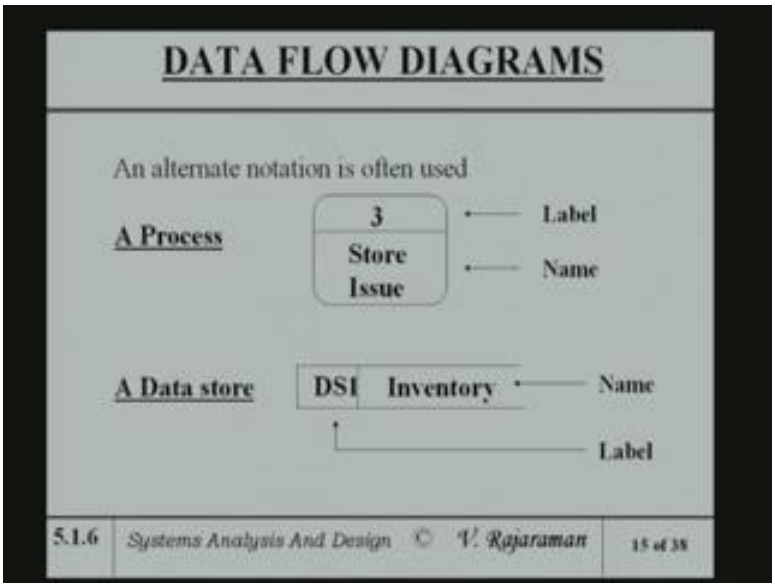
But, data can flow from external entity to a process, process to external entity, process to a store and back and process to a process. So, in order to check the dataflow diagram you drawn on these rules are followed. The tools which are there normally will do some of automatically. But then this time the rules are so simple that should not be very difficult for you to kind of inspect the diagram and see whether you can correctly drawn it.

If you are not followed any of the rules that means, there are logical error, logically you are not understood writing from file to file obliviously is without an intermediate processing is obviously, logical error. Data flow cannot flow from external entity to external entity, that is also obvious when ultimately see the external entity creates data and consumes data and if you send one to other without any processing, you have to just not going transformation, just forwarding it.

So, that it is not the purpose of an external entity in a dataflow diagram. The external entity, data cannot flow from external entity to store. By as I said the external entity is a dumb, and store to external entity, store has to only talk with the processor and store to store which again I am again ((Refer Time: 30:14)) that. In other words, primarily the

intermediary in all file operation is the process. And similarly intermediary between two external entity is again a process.

(Refer Slide Time: 30:33)



There is an another alternative symbol, which is used in some books for the dataflow diagram as I said some people use ellipses. And another one is some kind of rounded rectangle with the label, which gives the serial number of that process and the store issue is the process name. Just like you has the use process name like, store processing and so on. So, instead of circle ((Refer Time: 31:06)) use the rectangle with the rounded edges.

And data store is shown as the open rectangle with giving the name of the particular file and also some label for that file, this is another way of doing it. Again of course, in this case you could read and write either, you can show it vertically coming up or coming down, is like you can show it like this or you can show it like this. Both are alright I mean, but then it will be the notation will be obvious.

The good style in drawing DFDs is to use meaningful names for data flows, processes and data stores. The whole idea of using meaningful names is that is the documentation name, it is also aid putting the customer and the designer. So, something which is descriptive will be immediately understood, if you just use x y z, it is like any program

when you write, we always tell you that the variable names, which you use in program should not be x y z and stuff like that.

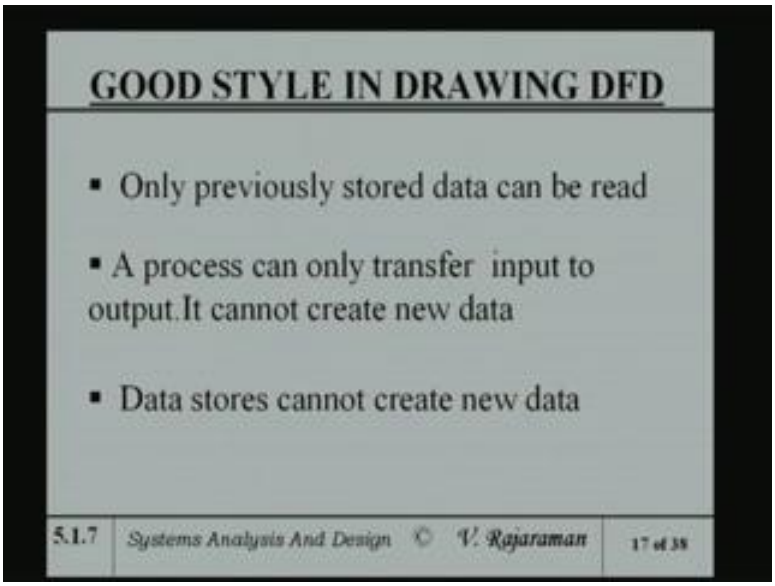
You normally use meaningful variable name, that is why new many of the programming languages allow you names for the variables, which as long as you wish to have it. By and large you use the reasonable name, so that the semantics of the meaning is obvious is somebody who read the program. Because in program, situation is like you may be have written the program and I may left the company and somebody else will comes later on to maintain your program.

So, if you write meaningless names, then the other person cannot even understand what you wrote. So, meaningful names are essential and many companies have standards in terms of the importance of meaning full names. And the manager will not accept the program unless you use reasonable meaningful name and also have dictionary to explaining these things. Similarly, in a data flow diagram in use of meaningful names of data flows.

And the dataflow itself, will consist of an number of individual fields or data items and they also go it to be described, which will come later currently we only talking about the we are not talking about the structure if the data. But, we only talking about the name of the data, what data is used top level down developments contact from the starting diagram and successively leveling DFDs.

What is meant by top down design or hierarchically, explain in terms of the house building plan, you will have an overall plan and then you have to look at the each of the separate rules and details. Similarly you look at an overview and then go down wards.

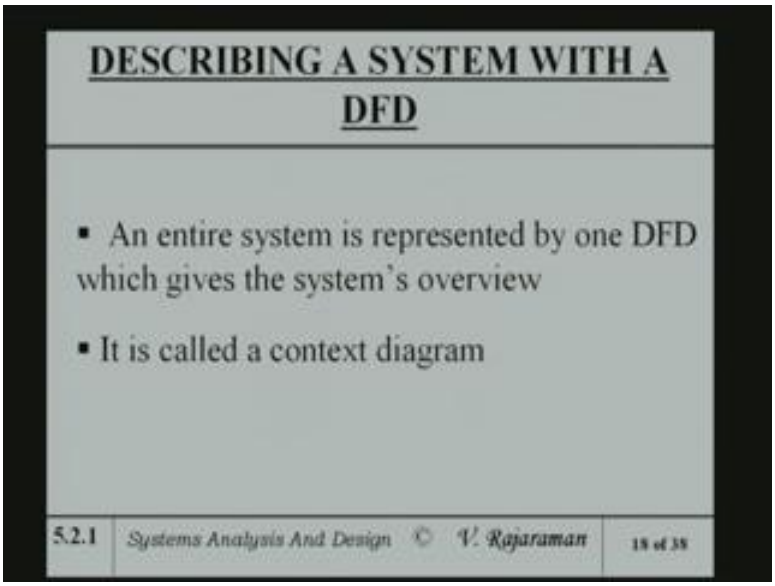
(Refer Slide Time: 34:49)



Only previously stored data can be read obviously, you cannot something read which is not store. A process can only transfer input to output, it cannot create new data. What is meant by, it cannot create new data I mean, I am saying because one might question me it is creating a new data it is processing. What I mean is the some input data, there is some data in the file and that data is coming from the input and that with the file together, you create the output data.

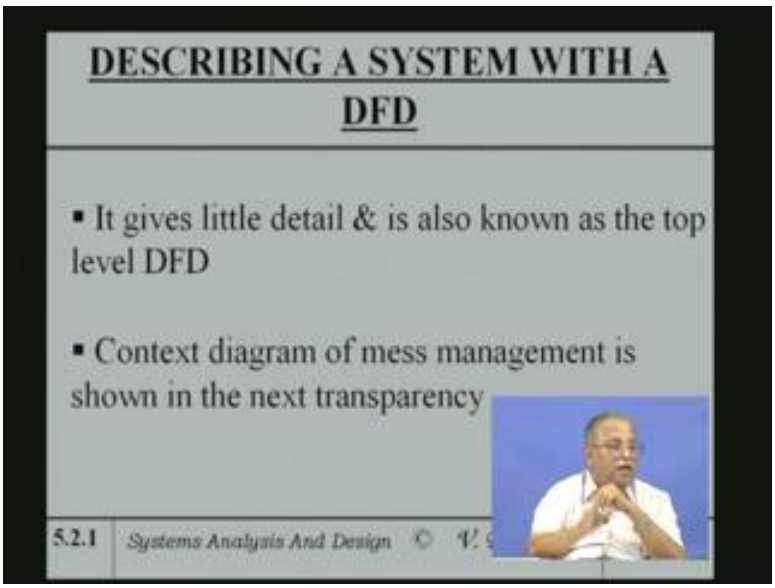
You cannot have something which coming from the output, which is neither in the input nor in the store that what I mean by saying it cannot create new data, that is new data element I might say. But, it can create process data, data store cannot create new data, again for obviously reason, you can store them, you can read, you can write, but it cannot by itself generate a new data item.

(Refer Slide Time: 36:11)



In order to kind of generate the DFD, first you have to look at an entire system and represented by one DFD which is use a system overview, it is called the context diagram.

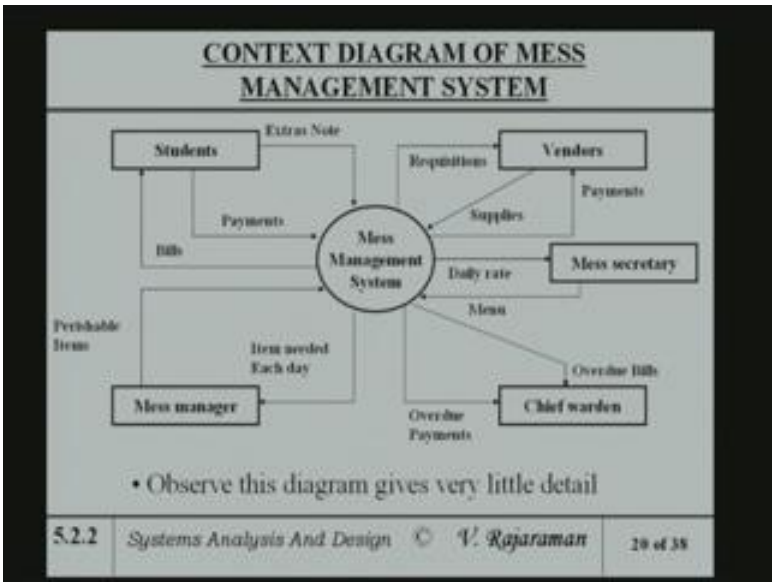
(Refer Slide Time: 36:31)



The context diagram gives little detail is also known as top level DFD, I explain I will give a example, of what I mean by context diagram. And what how to expand, this is what the whole idea is about, so called levelling.

I will look at the context diagram mess management system, which we talking about the mess management system.

(Refer Slide Time: 37:04)



We said that this is the running case, which I have been using all along. So, in the context diagram you only give an overview of what the mess management system is expected to do. And which what external entities it is suppose be interactive, you do not say anything about the details about the details of process. And you have to, kind of get down to very details by breaking up the mess management system, into sub parts which do this visual processing.

Observe this diagram it gives very little detail, there are the mess management system has as its users student, mess manager, chief warden, mess secretary and vendors. So, now you look at, what are all the important data elements, which each of the entities is going to provide. Now, a student if he is a regular member of the mess whenever he takes extras and so, on or as guest he has to provide that information otherwise, we assume there is a member.

As a member having use of certain facility of certain number of days, for those of days will be charged. So, the student also has to pay payments and based on the bills which are send to him because include both the regular bill, plus additions are extras. So, as for as student interaction is concerned these are the interactions. Vendor get requisitions purchase requisitions for purchasing items for the mess and he supplies the items and he gets payments back.

So, primarily the three most important things as for as the vendor is concerned or requisitions, supplies and payments. Another way of telling requisitions in order to the vendor and the mess secretary has supposed to be giving a daily menu to the mess management system. And as I have been talking about, the menu should be such that the daily rate should not widely vary and the foods you give should be balanced.

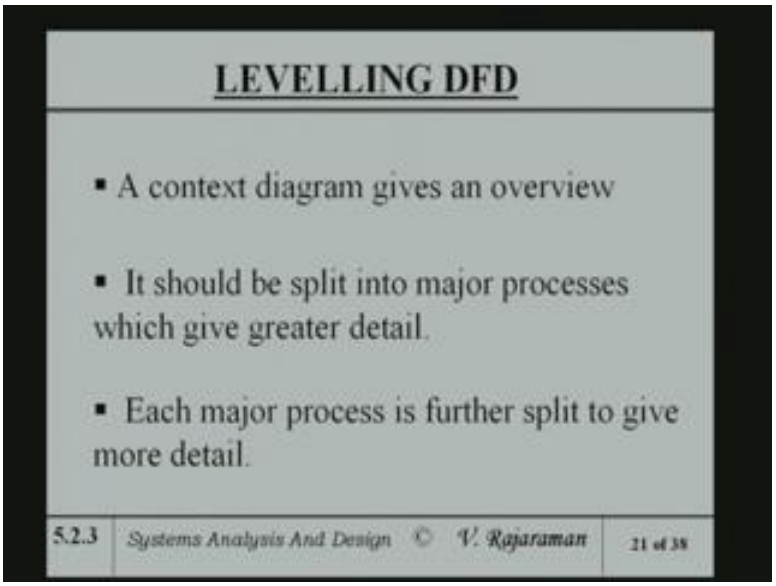
So, these are the requirements of the system and so in order to able to give a reasonable menu, he has a running daily rate available then he can adjust. So, that there is no wide variations in the daily rate. Now, the warden, the main warden's major job is to of course, is to make sure is going to be looking at the exceptional report primarily, because the day to day running is not of his concern.

If there is a overdue payment from students, then other words we said students are given five days or something whatever to clear the bill. If he does not clear the bill then the information goes to the warden, so you can call the student and do something about it. And similarly bills have to be paid in time to vendors and so on and anything, which is overdue that also is got to be mentioned, these are the things which you require for what I would say not operational, the reports are smaller in number, but they are require to kind of have a overall control the system.

And the other entity we have shown is a mess manager, base in the menu the items in a everyday has to be given to the mess manager, based on that you have to find out what perishable and things like that. So, perishable goods processing of somewhat different from the non-perishable goods processing, all those things we discussed, but this gives an overview of what the interaction system is about.

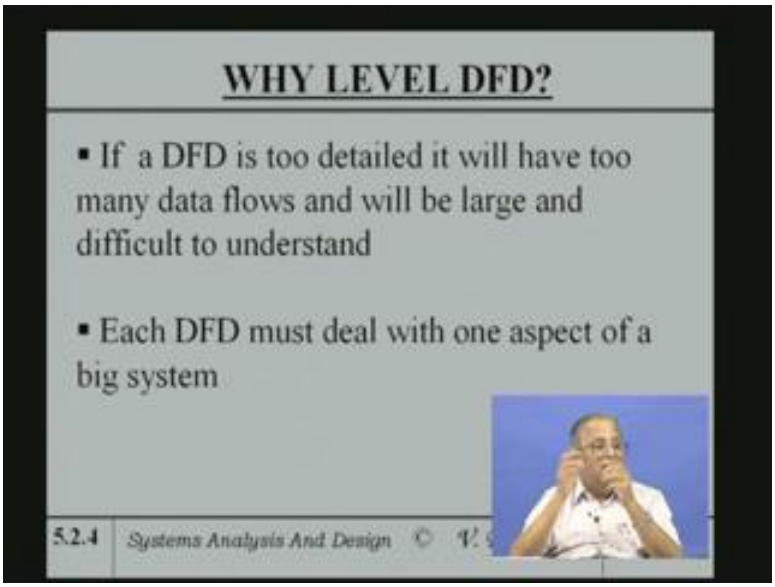
And of course, probably you should have some kind of a write up to accompany that, if you are going to talk to a manager or user, this is going to become part of the SRS, system requirement specification saying that this is what your system will do. So this is the kind of entities and process management system and give some of the details of what the management system will do.

(Refer Slide Time: 42:53)



So, in context diagram gives an overview, should we split into major process which will give greater detail each major process is further split into give more and more detail. Each one, this become number of process each process for them.

(Refer Slide Time: 43:05)

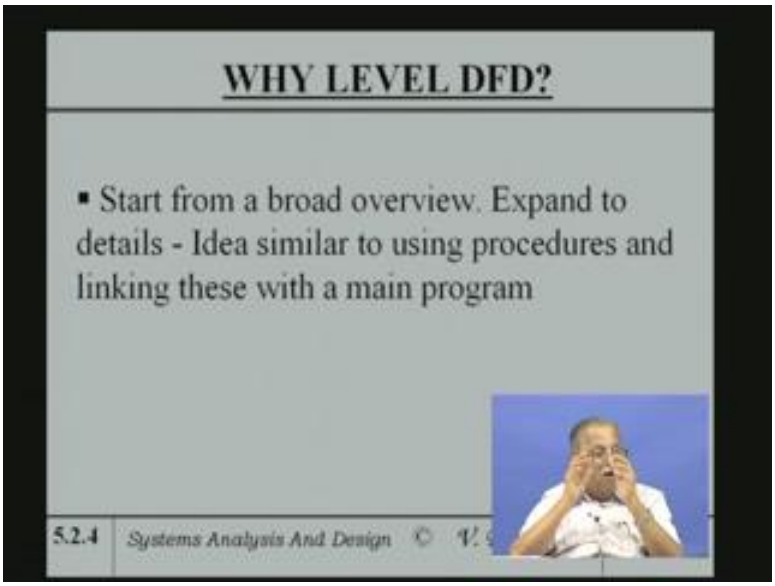


See, if the DFD is too detailed it will have too many data flows; will be large and difficult to understand, it is obvious. In other words the best thing is something which you can fit into a page; you can fit something into a page, at a glance you can see what is going on. And you can connect it to other pages, like the flow chart, you should draw

effectively look at not exceeding large number, nobody likes to have a big sheet of paper this big, where all the data flows are given utterly confused.

So, the point is that, each DFD has filled only the one aspect of the big system, that's how we can divide up into not more than one page.

(Refer Slide Time: 44:00)



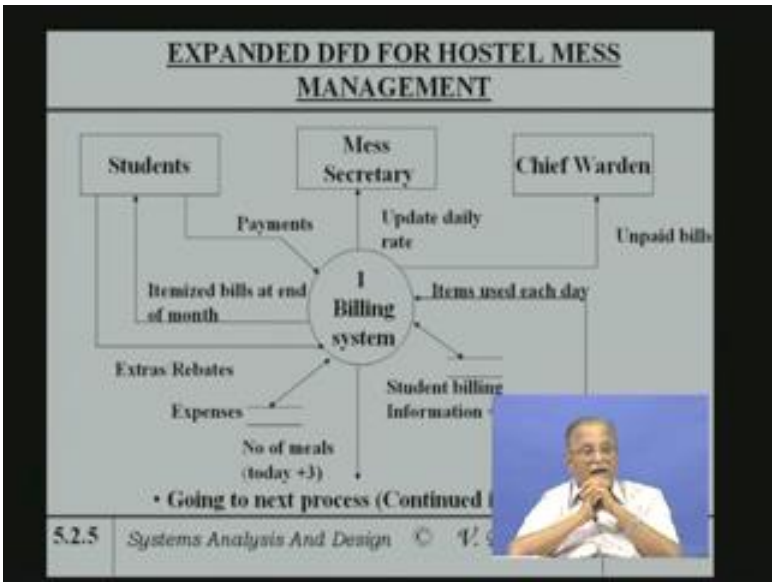
Start from broad overview, expand details idea similar to using procedures and linking these with a main program as you know in a programming in similar situation. If you have very complex program, one can say monolithic program, it is very difficult to change the program. And so we always tell students when they learn programming, that divide and conquer, that is divide a large problem into smaller parts and each smaller parts you can write a procedure and you can link all these procedures or functions whatever.

And of course, the newer method again to come up with objects, which are reusable, objects and so on. So, you create a set of objects and you link all the objects to get there to back up the big program. Again it is splitting something into smaller set of parts, which are all can be independently design. So, it is a similar idea, this idea recurs again and again in computer science, but of course, in the common sense nobody would like to look at a big you know monolithic thing and always try thing to make it up into smaller parts and will be able digest the smaller parts.

So, the later on in this course, we will talk about object orientation also and that would be also required from the same point of view, there is in a large system, what are the important objects in that system. And all those objects can de design and how to create reusable objects, things that type. So, similarly one might even think about reusable parts of a DFD for similar systems.

(Refer Slide Time: 46:14)

.



Now, let us look at the expanded DFD for a hostel management system, now the overview in the mess management system. We had no greater details, we had number of external entities, which is set which you process, one of the things which are being done in the mess manager system, so billing system. The billing system is to produce rules for the students primarily and so the students can make payments.

Another part of billing system, which also will deal with the vendors, because as I said the payments also to the vendors. But, I am looking at the billing system specific to a student, as you can see here the bill itemized bills at the end of the month is what is being generated by this system. And in order to do that of course, it has got to kind of a get inputs from many different entities.

Now, the mess manger have to give the billing system, items use these day and of course, there should be also in some file the cost of the items, which are used. Normally the cost

of the items would normally be stored in a file, I am not shown that in great detail here, because it will be continued on future figures and so, on. The students are going to be sent itemized bills at the end of the month because that is the one of the requirements we said previously they are not getting any itemized bills.

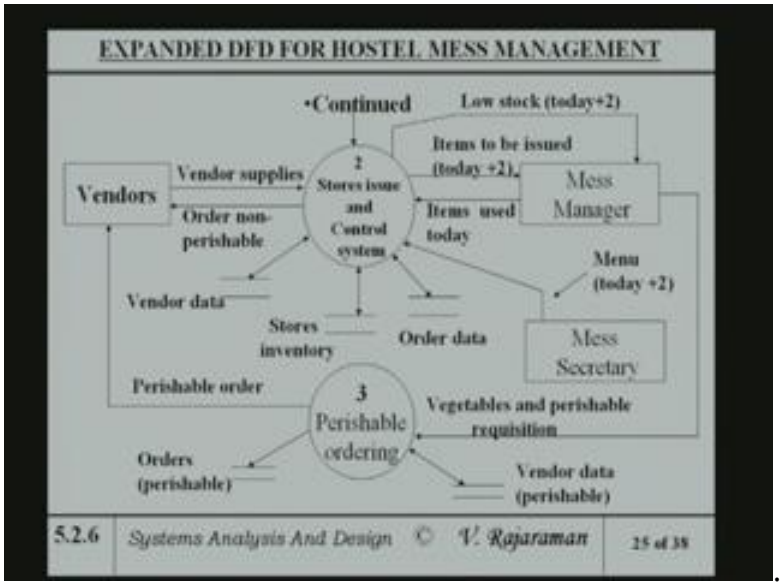
And payments come to the billing system and extras and rebates, the student's extras and rebates in terms of is not being member of the mess and so on also flows in the system. And the billing system should update the daily rate, in order for the mess secretary to clear the proper menu, so that is the updation is done by the billing system. And the billing system also sends for chief warden, un-paid bills and also late payment by students.

And apart from that expense, number of meals, the interesting thing here is there are two files. One is the student billing information plus bills, what is meant by student billing system you can see the arrows lowercase that means, reading from that file and writing, what is being read from the file is a permanent data about the members and mess. Roll number, whatever room number, whatever and then as a men they are using guest, so on bringing guest and whenever they get going to be absent, all the information sent to the billing file.

So, that number of days he actually use the mess, number of extras he use, all that is immediately updated in that file, that is what it writes in that file and that what it is used later on to bill. And the expenses, on the based on the cost of the items used and the menu used and so on is written in to that expense file and also expense file as we read, to be able to generate daily rate update.

Now, apart from the billing system there is a store issue and control system also a perishable ordering system, there is a vendor order for both perishable as well as non perishable items. We will again divide up the ordering system into two parts one is perishable order and non perishable order. But, any case in this case have an example, I have shown the stores issue and control system as one DFD, as you just connect it to the previous one, other words as you can see as I said, continued that means, the going attribute in next process it says….

(Refer Slide Time: 51:46)



So, that line will come as an input to this, so what should be the label there, label as you know which actually an error. The billing system to generate to the store system, items used and items requested, because that is the one which is going to use by the stores issue. So, the vendor supplies coming and items require is what comes from the previous one and control system will create an order.

Non perishable order probably made once in a fortnight or once in a month, whatever the period will be determined by the variability or the number of members in the mess. The number of members in the mess remains reasonably constant month after month, one may decide on a timing of one month. Otherwise if it is variable, if it desire to be half a month depends on the situation.

But in many case it is a bulk order, once in a few days whereas perishable order something which got to be done every day because milk and vegetables and so on you cannot order a month ahead of time. Because also we have said that the vegetable prices change from day to date, food price change from day to date and so the billing system also has to have that data.

So, there is a separate perishable ordering system, apart from the stores itself can do an order periodically once a month or whatever. Now, the stores issue system, the mess secretary has to give a menu. In fact, if it is give the daily menu for difficult for the stores issue to kind of issue the proper amount of items to the mess manager. So, I am assuming

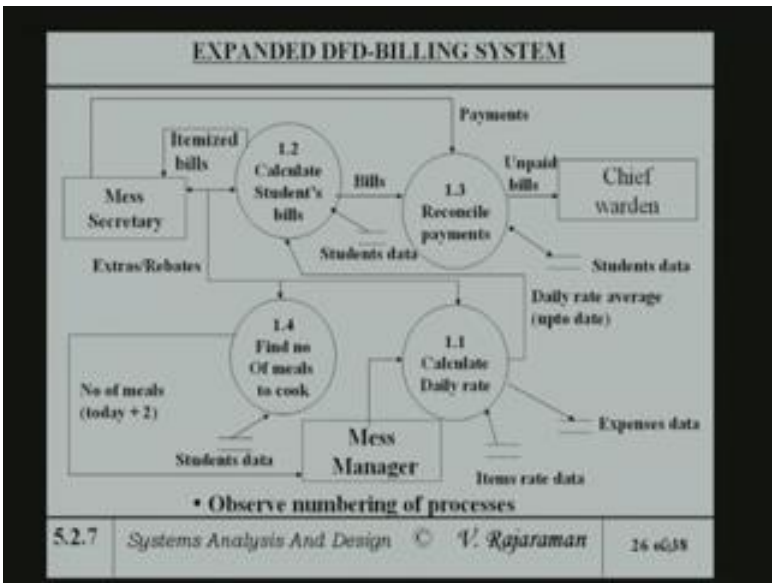
as a two day buffer. Other words, you should at least tell what the menu has to be 2 days from now.

In order to able to order the right amount of perishable goods particularly to avoid wastage. So, the menu of today plus 2 must be given and the items to be issued to 2 days buffer as we told to the mess manager. And the mess manager has to in term send a perishable ordering system as you can see here, vegetables and perishable requisition have to go and the ordering system will order this, separate order file is here, perishable order and vendor data for perishable item is there.

So, the point is have taken a large context diagram and now I am breaking it up into number of parts and each part is to some extend explainable and some extend try to be self contained. And gave an idea of what situation like, suppose the stores issue system is like find out the low stock for whatever items have been asked, the menu then the low stock has got to be told to the mess manager. So, that he can order appropriately either here or he can send an information here for the vendor, for ordering non perishable.

Normally, as I said non perishable will not really go low stock, it may, the low stock may be only for something which is perishable. But anyhow that is matter of detail, the point as for us this particular talk is concerned is that you divide up into a multiple parts.

(Refer Slide Time: 56:56)



And now one itself there is the one which I took billing system, itself can be speed into number parts. They calculate students bills, reconcile payments, find number of meals to cook, calculate daily rates there are number things are mentioned in that, in billing. Now, the billing system itself, the billing system is numbered one. And the billing system itself consists of number of parts which I explained in words. So, explain you put it in greater detail into expandable DFD in the billing system.

So, there are point is that just like breaking up a large problem into smaller problems. And each smaller problem is further smaller problem, which you can tackle it so. It could turn out the reach of these processes, in very simple programs to write, actually turn out to be simple objects in an object oriented system you are looking at or it could be simple function or whatever, depending upon the method you are using or programming. But that is something for the system designer to decide.

But the broken up thing, the initiation is make it somewhat simpler for the person down below, who is going to use it to create program, it is also kind of some use to the manager or some user. But this may not be of the great use to user because I am getting into greater details, but maybe because at this stage you can kind of any logical error, which may be there in the system you can point out, saying that you have not done this.

So, it is definitely understandable by both the user and systems designer. So, each one of them has to be described in greater details and I think. So, I will essentially tell that the point is that you have a billing part, which is divided into 1.1 1.2 1.3 and 1.4. And if you are further going to take calculate student bills to something lower, it will become 2.1 things like that and some numbering system like in a book, you got section numbers, sub-section numbers things like that.

So, with this I conclude this talk and look at levelling, next time.