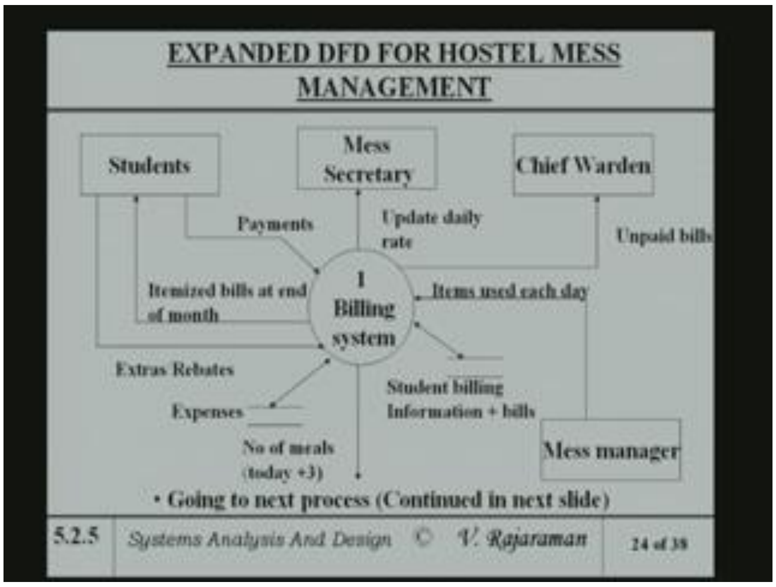
Systems Analysis and Design

Prof. V. Rajaraman
Department of Super Computers Education & Research
Indian Institute Of Science, Bangalore

Lecture - 16

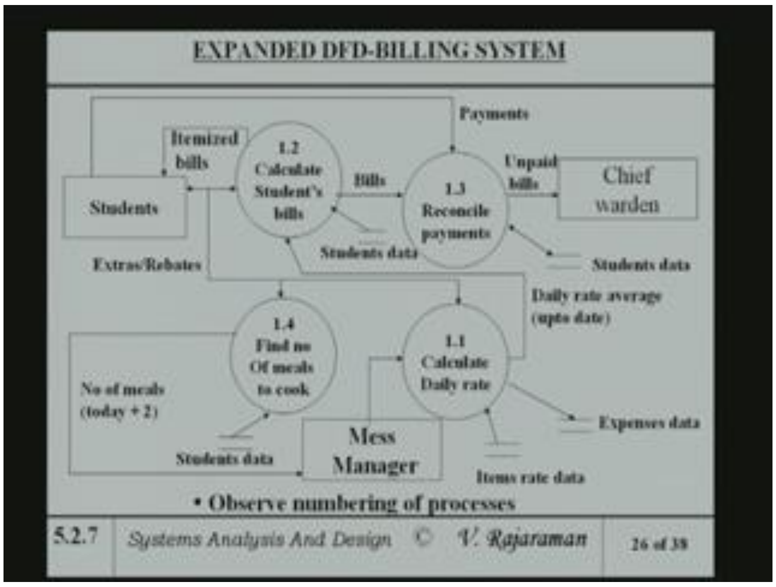
Last time we started looking at expanding a DFD to several DFDs. In fact, we started with hostel mess management system, which is our running example. So, we said billing system among that in that mess management system. Billing system is an important part. The billing system essentially takes the menu from the mess manager based on he gives items and use in that menu.

(Refer Slide Time: 01:44)


EXPANDED DFD FOR HOSTEL MESS MANAGEMENT
Students
Mess Secretary
Chief Warden
Payments
Update daily rate
Unpaid bills
Itemized bills at end of month
1 Billing system
Items used each day
Extras Rebates
Expenses
Student billing Information + bills
No of meals (today +3)
Mess manager
• Going to next process (Continued in next slide)
5.2.5
Systems Analysis And Design © V. Rajaraman
24 of 38

Based on that the update daily rate goes to the mess secretary, who kind of knows what it is going to cost. And at the end of the month payments will be received from the students for the bills, which are sent. And also the bills will have itemization as we said. And the expenses are noted in a file and student information, plus a bill is in another file. And this is the general structure, with which you started and next we went to from there.

(Refer Slide Time: 02:31)



We expanded the billing system into several parts there is, in other words as I said in all complex examples, the basic rule is to divide and conquer. So, for dividing and conquering you split up a very basic high level DFD. A high level dataflow diagram, into a set of connected diagrams, where each one, can be separately looked at. And each the programs or processes in directly. So, that is whole idea of really breaking down, in fact, technically it is called leveling of DFD.

Breaking in to single DFD into sub-components, where each component belongs to one. So, it is a billing system itself consist of number of parts one is calculating student bills, last time there is an error in that student square. And infact, it is written as like remember like mess manager or so. So, we changed it and you just note it actually, because the bills go to the student, not to the mess manager and the payments are received from the students.

So, the billing system can be divided into, we can calculate student's bills. In fact, we are finding the number of meals based on which, the daily rate can be calculated because the daily rate depends upon the number of meals to cook and also the menu. So, the expenses data is apart from the daily wages calculated rate calculated. Total expenses is put on a separate file. And there is an item rate, which is used in order to calculate the daily rate.

Because each item has got certain cost and you have to really find out the billing, based on that cost. And similarly we separate part talking about reconciling payments. In other words, when the payments are received from the students, they got to be checked against the bill, which is sent and that is called reconciling. And if there is payment under payment or overpayment. Then of course, there should an exceptional report or a notice, which you should issue to the student. Saying that, your bill is not properly paid and if it is unpaid bill for long pending period.

We said there is entirely chief warden, for taking action. So, the important point really want to make is that the billing system, itself being divided into number of sub parts as a part of division. The question of how to divide it logically depends on the problem. And it is also some extend arbitrarily, it is not there is a specific way you can break down. You really have to think logically saying that something, which is got to billing has to have something to do with number of meals cooked.

You have to something to do with the rates of item and daily rates. And you got to something to do, which the fact that the really rates in effect are also given; to the based on that students bills are prepared and reconciling part is very important. Because, one little part, which I could have ordered, namely the student mess secretary. Who is suppose to give, the actual menu for the 2 days ahead of time. So, that the actual items can be bought, ahead of time from the vendors.

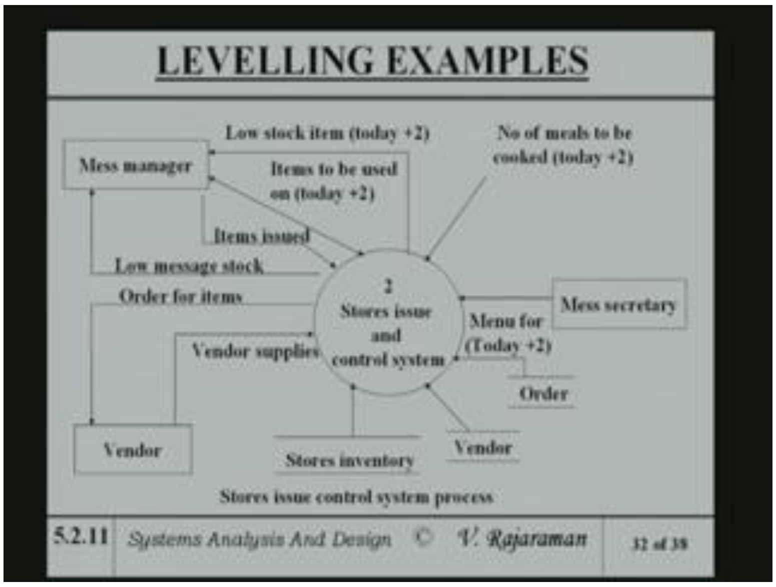
And this, I am not putting this primarily because it is strucking it up, there is goods not enough space, this PPT to be able to put that and clutter up the whole thing. So, I thought I will mention it and not just clutter up the thing you can put it you self. In other words, got to be in some kind of a input to come from the student mess committee manager or student mess committee secretary. Who decides the members and so on. And you should be able to give the items to be able to use two days ahead of time, to calculate daily rate because only based on the item cost the daily rate can be calculated.

And of course, a mess manager also has to kind of be told about of what items to number of meals to be cooked, based on the number of students and also based on the menu and so. In other words, calculate daily rate infact, it is the most complex of the whole set of processes. Because, it has got number of parts will do. I could in fact divided up calculate daily rate into multiple parts. One is to find out the total number of items to be used on a any given day.

And then another module, which essentially says having calculated finding out the total number of items to be used on a any given day, for based on the menu. Calculate the daily rate which can be a day, which uses these. In other words, the point I am trying to make is the number of the processes are circled in which you want to divide. But, divide is depend upon on your convenience. I mean there is something you like if you divide into too many parts.

Then also, it is you know each process will become a three line process is just like the same rule in programming languages. In a program you do not want to make function, just two line or three lines, function must do something substantial. That is the reason I have use this way. But you could try it out, in other words I am not that is nothing sacred in this particular division of the billing DFD into multiple parts.

(Refer Slide Time: 09:25)



LEVELLING EXAMPLES

Stores issue control system process

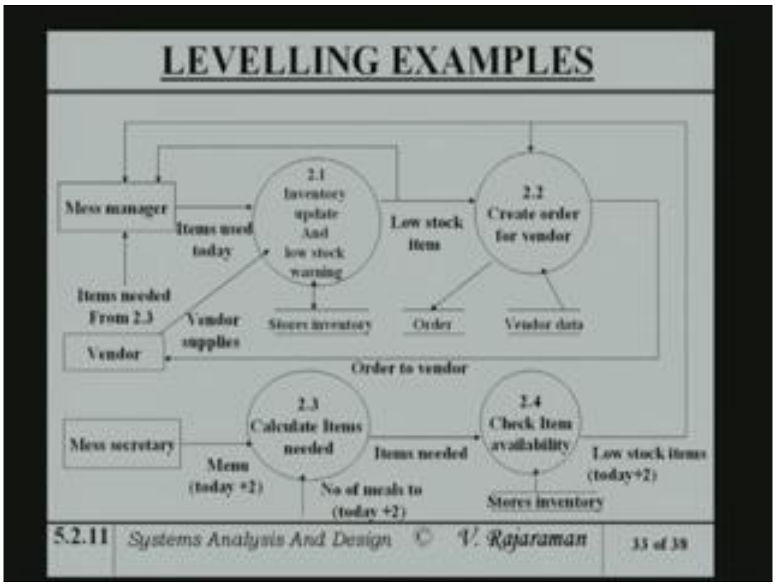
5.2.11 Systems Analysis And Design © V. Rajaraman 32 of 38

Now, let us look at the stores issue, stores issues other part which is actually it came from the mess management system. The mess management system was a single high level chart, then we divided up into multiple parts. So, this particular one, now is the stores issue and control system, there is billing system is one part. In other words we give a number 1 to that billing system. We give a number 2 to the store issue and control system.

Store issue and control system, again based on the issue has to be done by the mess manager to the cooks and so on. And orders also have to be placed. So, the stores issue and control system, primarily looks at the situation and terms of the ordering to be done at the appropriate time. Vendor payments to be made and the orders which you are placed and all that. In other words, there are number of things, which it does and again because there is I am giving a high level part.

Now, that is can be broken up into number of sub-parts and each sub-part will have a role to play like for the….

(Refer Slide Time: 10:50)



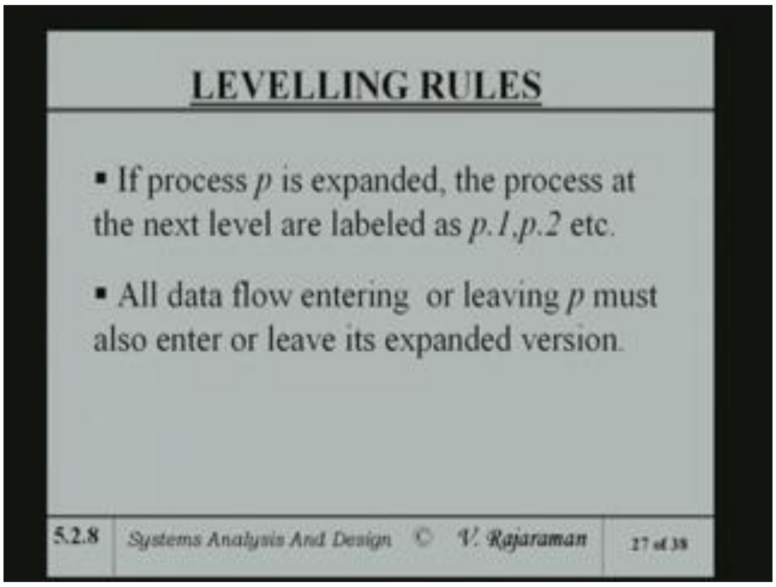
So, let us go to next one 2 itself become 2.1 2.2 2.3 2.4, which are all the parts of the system. In terms of the control we had to have a control of in other words, stores issue and management system must have inventory, update and low stock warning. So that whenever the stock becomes low he can be order in time. In other words as I pointed out, the current system they found that they were not able to predict.

Whereas, we used and the result that the ordering was in trickles and that added to cost, where we order in bike and delivery has to be done few times rather than too often. Then I think total cost will go down, the inventory update and low stock warning system is one part and creating order to the vendors is another function, is got to be done. And calculate such item needed, which is again unless you know the items needed, you cannot really check the item availability.

In other words, if the mess secretary gives certain menu for which items are not available. Then you have to really tell either order it immediately if it is an essential or this got to be feed back with the mess secretary. Saying that this particular item is not available and would you like to change the menu. So, the point is to calculate item needed based on whatever menu is given and based on the availability of that you decide whether you can do it or not. In other words each one is again for the processes or you might say a function.

And so the high level DFD, which is a previous one. Namely the stores issue and control system has been split up into layer level DFD. This is what I said is levelling, so the levelling is what is being done.

(Refer Slide Time: 13:20)



LEVELLING RULES

- If process *p* is expanded, the process at the next level are labeled as *p.1,p.2* etc.
- All data flow entering or leaving *p* must also enter or leave its expanded version.

5.2.8 Systems Analysis And Design © V. Rajaraman 27 of 38

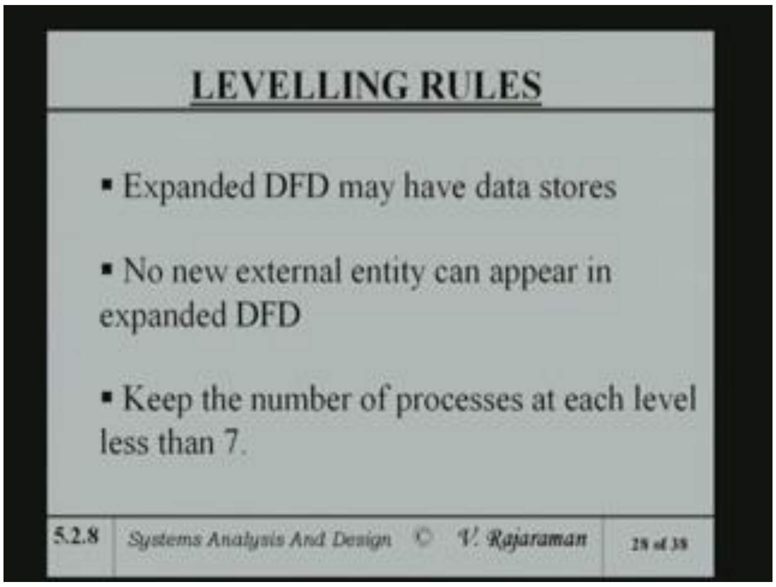
And let us go back to the rules of leveling, in other words, I just given you of examples of leveling. In other words the point the reason why I am doing it is, you have to fix your ideas. And it is always good to start with particular example and starting with particular example and generalize it. After generalize it, you come back and look at the particular example, to see if the generalization works or not and that is where, which actually presented in the web and the book and so on.

But, in the lecture I thought I could really go back and forth and be able to tell you that where do, the levelling rules arise I mean is it arbitrarily. But, there is logic behind all these rules and the logic has to be explained. And when you actually, do a levelling you have to check, whether you have to following the levelling rules. And you are not really contributing any of the rules which have been put down, one is the process p is expanded, the next level or label p point, p point 1, p point 2 and so, on.

In other words I said 1.1 or 1.2, 2.1 and 2.2 and so, on. The whole idea is nothing but, labelling to kind of know that it belongs to starting with a high level number 2, it goes to lower levels. All data flows entering the first one that is the high level should also enter the lower levels. In other words, unless you are there actually, you are breaking up single one into multiple ones, single one basically said data input.

And the data input has to be preserved; you cannot have a system with new data coming in which is not put in the original, unless you made a mistake in the original. Some times what happens is that when you do the levelling and you start expanding it. You find suddenly that you made an error in the original. So, you back to the original and change it there, but you make sure at the end of the game after your level. Whatever the data entered top level should also enter the low level, in other words expanded level.

(Refer Slide Time: 15:55)



LEVELLING RULES

- Expanded DFD may have data stores
- No new external entity can appear in expanded DFD
- Keep the number of processes at each level less than 7.

5.2.8 Systems Analysis And Design © V. Rajaraman 28 of 38

The expanded DFD, may have data stores that is new data stores can be created. In fact, if you have absorbed keenly, you could have created certain new files in the 2.1 2.2 and

2.3 and so on, which are over and above what are there in 2. I would like you to kind of go back and look at the tool itself, which is I think, I this 2 and here you had 1 2 files and 3 files. Here I have 1 file, 2 file, 3 file, 4 files, in fact, I created files. So, we have data stores, which I have shown in the expanded DFD, we have shown that new files are there, that is not wrong.

And because the expansion you would like to be able to kind of have the same data, split up into multiple files for convenience, that is effectively what we have done. For convenience in particularly for the process, when you write the process, the data only required by the process only is put up in the file. The file will have all kind of cluttered information, it may not be necessary. In other words, it may not be essential wasting actually space.

Because, when you come to database management system, we will look at some of the issues, in terms of how to kind of break up very large data base into parts, which are all using certain rules and certain methods. We have to make sure that the after we make some alterations in the data base management system, we have, so called relation in that case. And one larger relation can be split up into other relations. So, we look in to that at a later time.

But, at this time I am really looking at not detail of implementation. So, only when you get an implementation, you have to worry about database management system, the kind of database you are going to use in all that. This is the higher level, just to tell them to make sure that datas available in files and you have to make sure itself that later on, when you give it to a programmer, he will be able to appropriately interpret it and clear into a proper database management system.

No new external entity can appear in the expanded DFD, in other words whatever the entities are there, what I mean by external entity is in our case was vendors, mess secretary, mess manager, students all these are external entities. Suddenly, you cannot bring a new external entity into lower level, because all the generation of data is all done by the external entities.

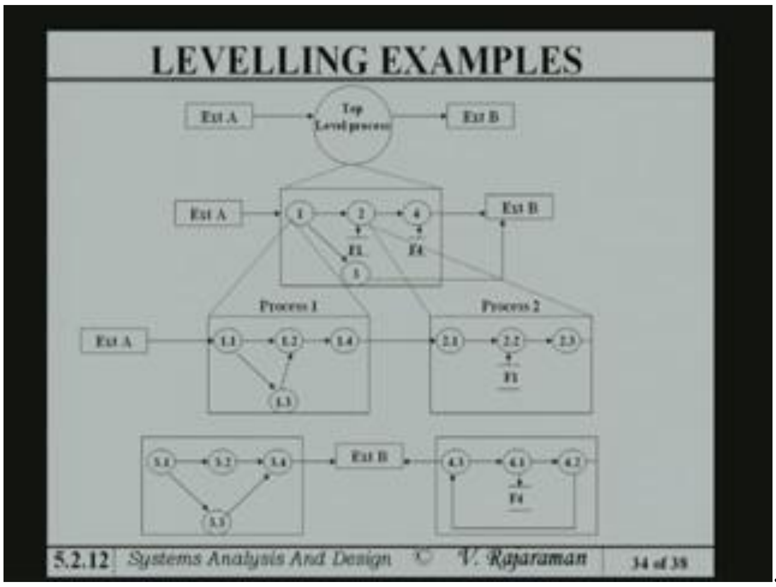
External entity are the generators of data as well as consumers of data, you cannot bring in a new generators and new consumer in an expanded DFD, which is not there in the original that means, you made a mistake in the original. So, you have to back and say may be if you add more to the external entities to be top level DFD and make sure that because of that kind of putting these things, otherwise it should not be there, also the keep the number of process at each level less than 7.

Other words the 7, which is again a magic number, the reason why you try to kind of break up something into smaller parts, breakup into two many smaller parts, you will not able to understand the complex, the each of them and their interrelationship at all very logically. There is a psychological rule which says is the magic number 7 plus or minus 2 that psychological experiment says that, people in general have a certain capacity to remember, like for instance telephone numbers, they have certain capacity average person you know.

There are people who remember 100 digits and so on, but by and large, people have certain capacity to kind of visualize in one visualization, certain number of parts is a number of visualization, with the number of parts becomes too many then the visualization loses its meaning. So, for visualization you have to keep your numbers reasonable, it should not be 2 or 3 it could be 4 or 5 and I have used only 3 or 4, for a simple reason that my screen size is limited.

And I cannot go on and putting of too many of them in my screens. But, if you feel that you want to expand this further, you could do it yourself which you can try next size.

(Refer Slide Time: 21:36)



So, this is the kind of levelling rules, using levelling rules and showing levelling example, an abstract example not an example which is particular is that from particular you go to general that's what I said particularly, you already looked at. Now, we are going to general, so general effectively says that your external entity A, in this case simplify matters and to fit in to my screen, I assume nearly two external entities one which is feeding the input, one which is feeding the output at the top level process.

And to be many more, but as I said this for my convenience of showing and then top level process is divided into number of smaller process 1 2 3 4, each one is certain new file. But external entity you see has remained the same they have not changed only two external entities. ((Refer Time: 22:39)). And now 1 itself is taken into 1.1 1.2 1.3 1.4, 2 is taken into 2.1 2.2 2.3 and so on and similarly for 2 and 3 and 4.

And one could, in fact, go beyond this and make it from 3.1 3.2 ((Refer Time: 23:03)) and so on in fact, I would not recommend on that. Simple reason is that unless it is very, very complex problem. Because you are going to end up with a million line codes something like that, which is true in certain live processes, there what they will probably do is that the top level process itself, will be separated out into many top level process. And each one can be expanded, so if you go too much to the grain size becomes too small then it is not very effective.

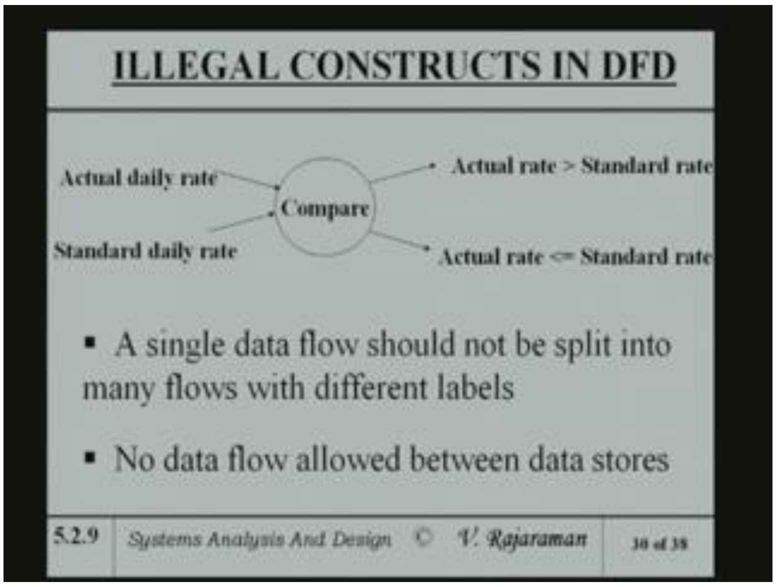
So, there should be reasonable grain size, my suggestion to give about 1 levels, 2 levels and 3 levels in the case, top level middle level and lowest level that exactly what I did in my example. So, let me go back and look at what all we did earlier, in terms of the. We, started with the mess management system is the top level DFD, which is what I have given in the picture.

And then from this, we went to the next level to 2 and 3 and 1 was there on the earlier one and then because 1 2 3 cannot be shown in 1. I showed in 1 and 2 and 3 other and from that I went to 1.2 1.3 broke up 1 into multiple parts. And then I broke up 2 into multiple parts, so effectively I have shown the levelling as an example. So, this is the essentially what is the specific case for the general case, in other words I started up with example, I went to general set of rules, I went back and showed you how I have done it.

So, it is not any magic levelling is a straight forward thing and only promotes certain amount of logical thinking, divide and conquer, dividing a large problem into smaller problems. Now, let us go back and look at some of the illegal constructs in DFDs, no loops are allowed in DFD, what I would mean no loops are allowed in DFD, one output cannot start from particular circle or processes go to other process and return back. So, what is essentially it becomes an endless loop.

Endless loop as you know have no cure, the machine was stopped. So, there should not be any loops a process cannot be a pure decision because a process has got to do a number of things. Normally as I said is not a single instruction, if you have a pure decision, single instructions, so single instruction has no meaning.

(Refer Slide Time: 26:37)



ILLEGAL CONSTRUCTS IN DFD

Actual daily rate
Standard daily rate
Compare
Actual rate > Standard rate
Actual rate <= Standard rate

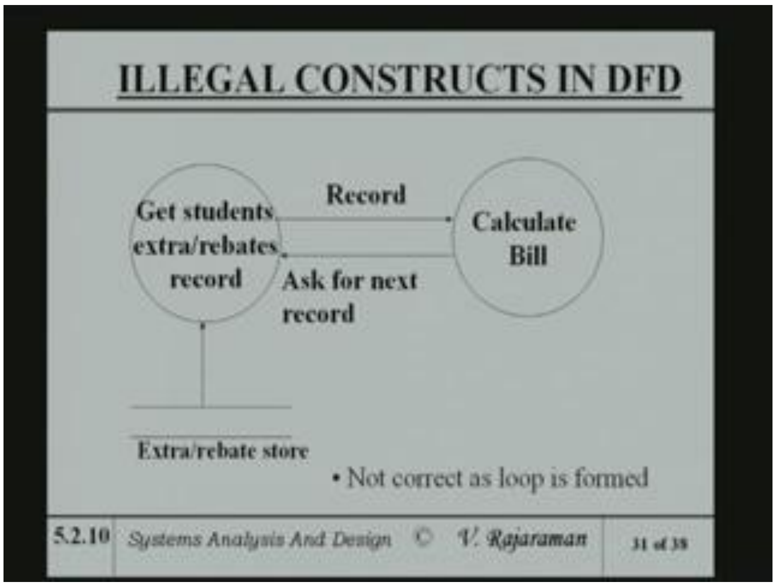
- A single data flow should not be split into many flows with different labels
- No data flow allowed between data stores

5.2.9 Systems Analysis And Design © V. Rajaraman 30 of 38

For instance, the single dataflow cannot be split up into many flows in different labels. Like output I can say here is actual rate greater than standard rate, actual rate less than equal to and also to compare a single comparison instruction. The comparison instruction is a single one, which is not you know, single instruction which is again is not really a process. No data flow can be allowed between stores, because stores are both dumps, dump stores.

Stores only store datas you cannot transfer from one to the other without any processing there is no point in doing it. If, you have to do something like that you might have to merge the two files.

(Refer Slide Time: 27:24)



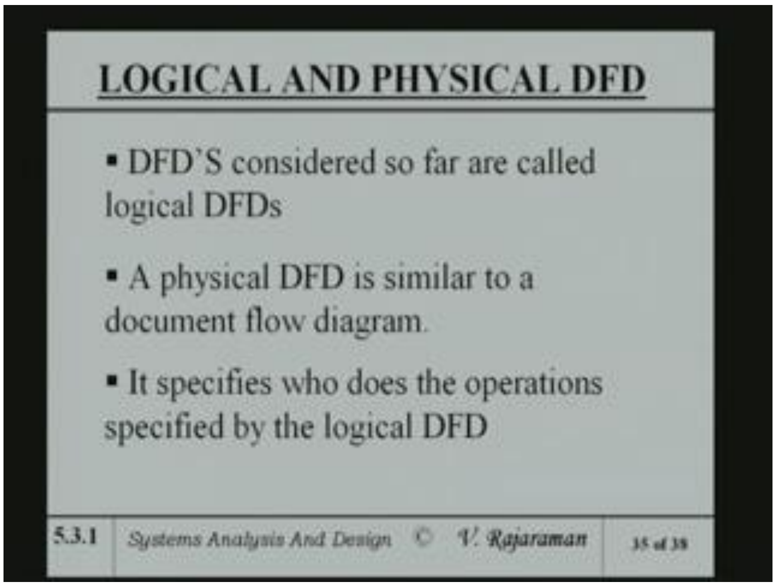
ILLEGAL CONSTRUCTS IN DFD

• Not correct as loop is formed

5.2.10 Systems Analysis And Design © V. Rajaraman 31 of 38

And this is an illegal construction there is a loop, record has taken calculate bill and ask for next record. And then you may end up with a situation, where next record is not there to hang up or there is no proper thing of put for the end of records it may hang up. So, this is a bad construct.

(Refer Slide Time: 27:46)



LOGICAL AND PHYSICAL DFD

- DFD'S considered so far are called logical DFDs
- A physical DFD is similar to a document flow diagram.
- It specifies who does the operations specified by the logical DFD

5.3.1 Systems Analysis And Design © V. Rajaraman 35 of 38

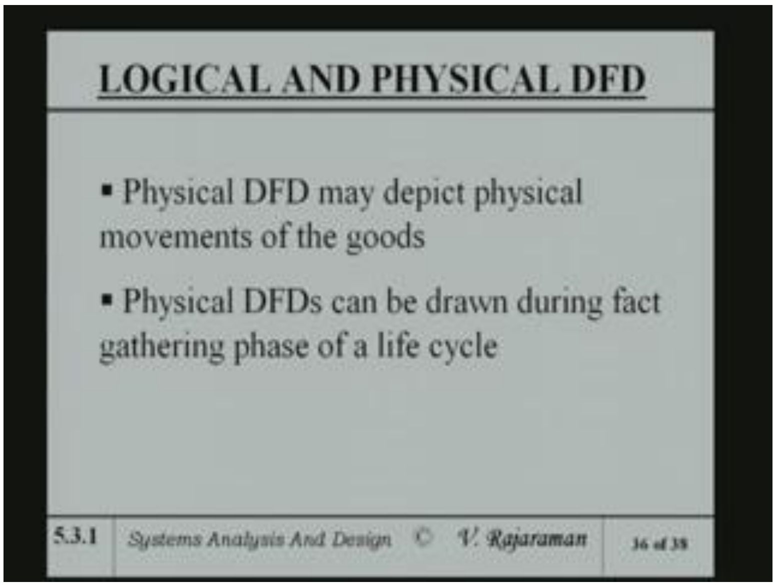
We have been looking at, so far what is known as logical DFDs what I mean by logical DFD is that we are really looked at the logic of the system. In terms of what logical datas

Flow, how the logical datas are processed, how files are created, how the logical datas are taken from the input entity and process results into the output entities and so primarily it is more or less to do with logic, rather than physical documents.

Normally, when you work with the user, the user is lot more comfortable with the physical system, in other words he will tell you who generates what document, whom does it go, what does he do with the document. And what document as he generate, to whom does he send, see the documents need not entirely the logical document, it could be actually physical documents. So, physical DFD is similar to a document flow diagram, but it specifies who does this operation and the operations, which you are specified by the logical DFD.

In other words physical DFD tells who does what, otherwise...

(Refer Slide Time: 29:24)



LOGICAL AND PHYSICAL DFD

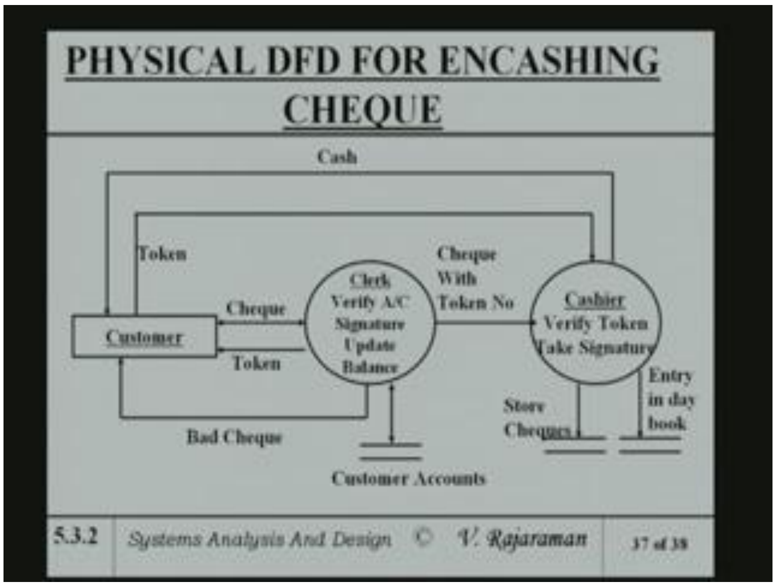
- Physical DFD may depict physical movements of the goods
- Physical DFDs can be drawn during fact gathering phase of a life cycle

5.3.1 Systems Analysis And Design © V. Rajaraman 36 of 38

Let me take an example, we will become clearer physical DFD depicts physical movements of goods, physical DFD can be drawn during fact gathering phase of a life cycle. As I said fact gathering said the users are lot more comfortable with documents flow and things like that, particularly if you are going to a nourish. If you are going to new hotel and trying to kind of a computerize new hotel management system, then they only tell you about, what are the existing method of taking the documents for the customer who comes.

And the document which is the bill is created, payment he makes, so on and so on. So, it is physical thing.

(Refer Slide Time: 30:15)


PHYSICAL DFD FOR ENCASHING CHEQUE
Cash
Token
Cheque With Token No
Cheque
Clerk Verify A/C Signature Update Balance
Cashier Verify Token Take Signature
Customer
Token
Entry in day book
Store Cheques
Bad Cheque
Customer Accounts
5.3.2
Systems Analysis And Design © V. Rajaraman
37 of 38

Let me give an example a daily example, in fact before bank computerization if you remember, may be you are not remember of course, still. Now, many banks which are rural areas are the areas which are not computerized, not bank which are not computerized. Normally small banks, which still use this technique in other words, the customer goes to a, what is known as the clerk or a teller.

And nowadays the term teller is used to have a clerk who gives the cash immediately, but I am essentially telling receiving clerk. And cheque is given by the customer to the clerk, and the clerk checks whether the cheque is valid, signature is valid and is there a balance in the check. And if there is a balance in the account to pay the check and whether any mistakes have been made in the entry of that cheque, other words you have entered the wrong date or you made some corrections then they will always tell you immediately kind of change the date and put your signature again.

So, this physical part even today if, you go and give a cheque because you have to give a physical check except ATMs, you have to give a physical cheque and physical cheque is actually, check by the clerk there except that the ledger is not used. But, a computer is used to look at the ledger to be stored in the computer and signature also stored in the computer.

So, signature which is there is been scanned and stored. So, scan signature and this signature are actually usually compared, most often it is usually compared it is not automatically compared. Because the automatic comparison method, are not still very much matured. I mean of course, they are coming may be in few years, you can just give a check it will be taken by a machine.

And everything will be checked there, even hand written part and then check your signature and so on fortunately that date has not come. We still need people to kind of do all those things. So, verify the signature at balance of course, customer accounts files are available to the clerks and if the bad cheque, the bad cheque is returned for either correction or saying that you do not have any balance. So, the cheque cannot be honored if the cheque is okay then a token is issued.

So that you can use the token, to go the cashier and take your money from the cashier. Now, the cashier also be told that this cheque has been validated and I have given it a token number. So, what that the clerk will do is write of that cheque the token number also and sign there, get the managers signature some places the amount is too large, it is got to be verified by a manager.

Manager also verifies it and then he puts the signature and then it goes to a cashier. Now, in the mean while the customer of course, is waiting for his cash, because this process which could be a manual process takes a few minutes. If it is good bank, it will over in five minutes. But if not such a good bank or rush is too much, first of the month something like that then it will take longer because, the queue will be build up and the clerk will only can do at a certain rate.

So, in any case the token, the whole idea of a token is that you know when to kind of present the token to the cashier and normally in most banks there is a display device on top. And when the token is ready, she will press the button and the token number will appear in the display device. And then you go the cashier present your token and get cash. And the cashier has to in turn stores the cheques, physical cheques because later on may be require for verification and the entry in the day book.

What is meant by day book is, it is issued cash she has to immediately update or cash balance because at the end of the day she is answerable or he is answerable, to how much of money was drawn by her. And how much of money being returned and so they will reconcile, how much of money was given out to customers. And they have reconcile the fact that, this money was taken, this money was given, the balance is so much.

So, that is facilitated by an entry book, it is called a day entry book in terms of the things this is the kind of thing, when you go and computerize the bank for the first time. The physical DFD is the one, which the manager of the bank or any clerk will be comfortable with. And you also understand the process of what goes on of course, this is a very, very simple process, but even that has got a couple of steps.

But it also shows the importance of the documents, documents are not we talk about paperless office, paperless system and so, on, all certain done, paper still play to be important role. Like your cheque book, your cheque is a paper and that has got a very important role to play and the physical cheques have to toured again. Because suppose, if you issued a self cheque and later on you dispute that self check was drawn by somebody for whom, which is not intended then they should go back and be able to retrieve your cheque.

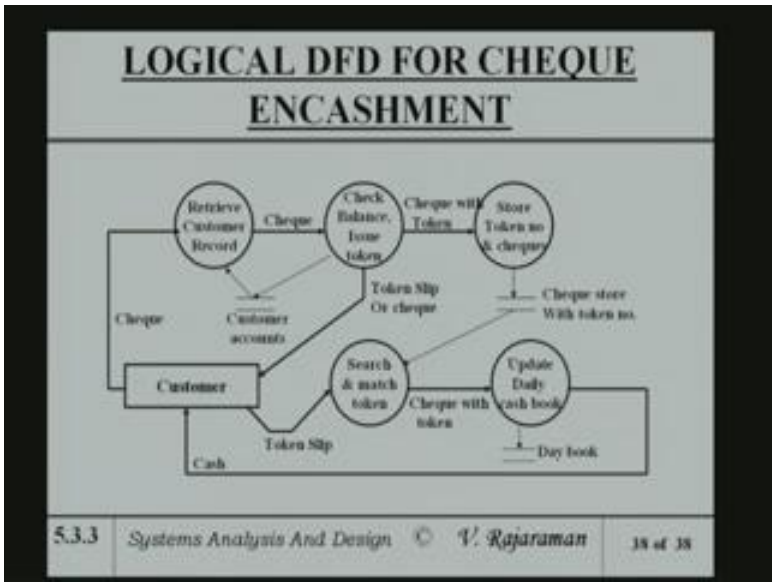
And then see who has drawn the money on your behalf and tell the customer that somebody else has taken it, may you made a mistake or whatever and not or it is a forgery in which case a complaint is lodged. So, the point I am going to make is, you cannot rid of also the paper document and many of very often, the stored cheques are kept stored for a few months. I mean normally, banks they have some kind of a time limit, saying that beyond three months or so, they will destroy all the cheques.

So, if you don’t go back and check your balances and find some cheques, which is issued not issued by you, after three months they will say it is too late, you have to come within a certain period. Three months itself is a very long time and in fact, some banks, modern banks, newer banks and they do not allow for a more than a month I mean, because storing all these paper take up storage space.

And if it is a bank in a central business locality, rentals are very high and you had to have store in store somewhere there, normally they send it out in archive. But then somebody comes and asks, because we cannot reply immediately it has to be taken out from the archives and brought back and so, on. The point I am really want to make is that the paper documents are important, the files important.

And because of the fact that, the paper documents are important, bills are important and storage documents are important, document flow diagrams is also important. It is not ((Refer Time: 38:13)) which you have ignored and given this document flow diagram.

(Refer Slide Time: 38:18)



LOGICAL DFD FOR CHEQUE ENCASHMENT

5.3.3 Systems Analysis And Design © V. Rajaraman 38 of 38

Then you go to logical DFD of same thing except, now I am going to look at what processes will be used in a computerized bank to be able to do with the same thing. So, the computerized bank, the customer present the cheque, so it retrieves the customer record from the customer account. And then the balance is checked and a token is issued. Again if it is teller system, instead of token issues, what the line will go from check balance issue to cash the customer.

And many banks, now have a teller up to a certain limit, that is probably go to the teller and get up to about 2000 rupees or so right away without any further.((Refer Time: 39:22)). Whereas, if it is cheque even if the computerized bank, cheque about 20000 rupees something like that then of course, they will not give naturally they will give immediate cash, teller will not give because teller cannot take the responsibility. So, token will be issued.

And the teller will store the token number and check and also the search and other words they I am breaking up the cashier's thing again. That is a store and token number and cheque is the cheque, either can be stored the physical storage or also in some computerized banks, what they will try to is of course, before you pay cash the token has got to be stored in a physical storage. But after the cashier is paid as I said the documents occupy too much space.

And you cannot keep of too long, some branches are now what they try to do scan it, scan the check and put the scan document in the database. So, that as I said complaint about three months later somebody comes, then you can easily retrieve it, the paper documents may retrieve in a big bundle and they have to take out one by one and so on, it will take a long time, where a retrieve by computer is reasonably fast.

So, if there is a particular bank, which is got a huge number of transactions, with cheques and so on they look put a scanner. Because they actually the scanner, the reason why they may not scanner everywhere is, it occupy space, it cost money and it may not worth scanning. But, some place they do it offline, other words they store it for the whole day, at the end of the day, they will take all those things off line, scan it and store it. So, these are the kinds of things, which you essentially as a systems analyst, you have to tell the bank person.

That is we got too many transactions, maybe you should not scan it online, actually scan it offline. But store physical cheque with token number, which of course can be file in the computer also. And then search and match token, the search and match token is again physical, normally it is physically done, you can also do logically in the sense that if token itself is a number entitle in the computer, token number the token number is on its travels and the token number comes to the search and match and then that displays automatically.

That also can be made somewhat automatic, that is the reason I have shown it in this way. But in this case, a token slip is given to the, for the matching against what comes to the check. And check with token goes to the update daily book, the day book is essentially as I said is an important part to be able to find out how much cash is issued, on a given day. So, the point is that, once you start with physical DFD then you look at the logical processes, which are involved and the logical process probably will make more sense to you.

And then, you can once the logical process makes sense to you, then you can go back and based on the suggestion that some methods of doing things, doing business or better computerization to the company for which you are doing it. This case I took very simple example for bank, same thing I could try for say hotel management system or something like that, where you look at maybe, you go to hotel and find out what do physically. And then you can essentially see how, it can be done logically.

That give you an example of this so, this module is kind of concerned with dataflow diagrams, the major issues are the major points we covered was that data flow diagram and important tool in analyzing systems. And you start certain rules are there, in terms of how to create dataflow diagram, data flow diagram actually graphical lane, the graphical lane the components consist of the process, files, external entity, data flows, input flows and output flows and so on.

And these are all how they are all interconnected and then we looked at the how, to start the top level data flow diagram and then we into kind of up or levelling and we looked into some of the levelling rules. And also we looked into some of the rules, which you should follow in creating the DFD, other words DFD itself you know, at the end of making a DFD to check you made mistakes or not, you back and look at the rules.

And if you are logically done the work, those rules are you could have automatically followed. But, sometimes you are careless and then something will be there, which like a loop, putting in to processes, then you can recognize that then immediately remove it. In fact, some of these things, there also tools available, tools are available doing number of things, tools are available for drawing the data flow diagram.

Drawing of course, consist of an nice looking drawing, you want to draw of course, you use the drawing tool in the PC itself, but DFD drawing tools are there, which give you the label and labelling and so on, levelling itself is of course, something little bit more, why I would say human. Automatic leveling will not be done by any computerized program. But, the checking of incorrect DFDs, like loops, like two files been connected. All those can be detected by a computer program.

So, the computer program available, to detect the errors, to kind of help you draw help you kind of store the thing, so that you can retrieve it. And we got so multiple diagrams show a larger diagram print it out, show larger diagram. But, nobody likes a large diagram. But nobody likes the very large diagram, you really have to kind of, always start with top level and then go the smaller levels and each one by one you should show.

So, these are all some tools, which are there and some tools which are in the open domain.

But, as you know open domain tools, normally gives a one month license or something like that and they expires. But, in any case they learn want to learn, you are always go the open domain tools and look through that and see what it does. And at least when you go and work for the company later on, you got professional tool you have not been completely unaware of the system tools.

That is the reason why should always look at open domain software, particularly as a students because most colleges cannot afford to buy some of these very expensive tools and so open domain of course, is itself, for one month free trail will be there. And you can try it out and you will not take more than a month, buy out some DFDs and so on. So, you can try it out for a few days and then of course it expires, it expires, and that is what I would suggest. In the web part of this course, there is web support material you have some site names and so on for your convenience.

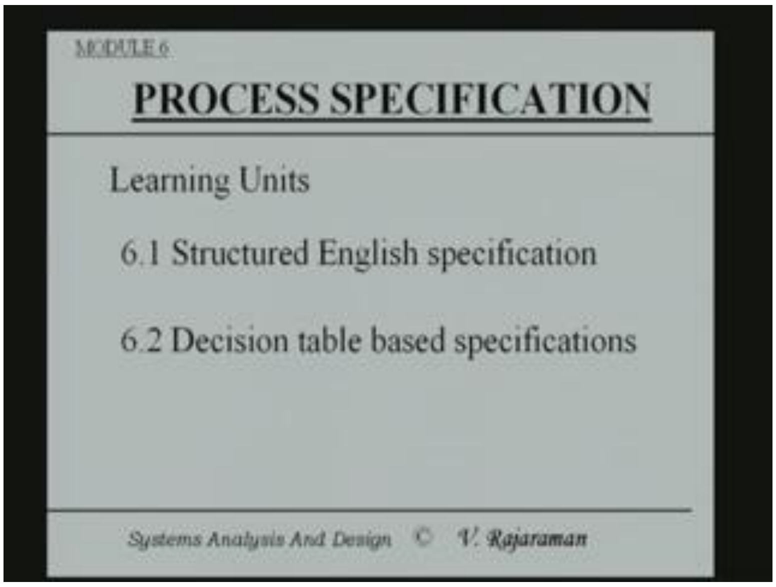
So, some URL s are been given, so you can look at those URLs, for some of their tools. And so, you can play around with that URL, see the whole complained about our education is that is impractical, other words we only talked about theory, we don’t do any practice. So, that is the reason why, the practice part unless you know how to use tools yourself, a practice cannot be taught in a lectures see, it can be taught to some extend in lab

But even in lab, I think things like searching for particular URL, dataflow diagram and so on it is anybody with a reasonable common sense and google search can do these things. So, what I would suggest is that in most of the things, I am going to talk about later on also apart from what I said lecture, our support material in the web which I have created. Apart from that also URLs, google search can do and get a lot of information.

In fact, sometime you call the information over killed, the whole idea I am putting a set of transparencies in the web support material for this course, without having to go through an ocean of information, you can find concisely. And also each module have to given the URLs, which are relevant for that module also given a comparison of other books, where you can read this and what books give a reasonable treatment of this.

So, other words the text book is important I have, but along with text book you also can look at some reference books. And this is the whole idea of combining lectures, also some web support material to go along with the lectures. So, this module is now over and I start with next module, which is a process specification.

(Refer Slide Time: 50:18)



MODULE 6

PROCESS SPECIFICATION

Learning Units

6.1 Structured English specification

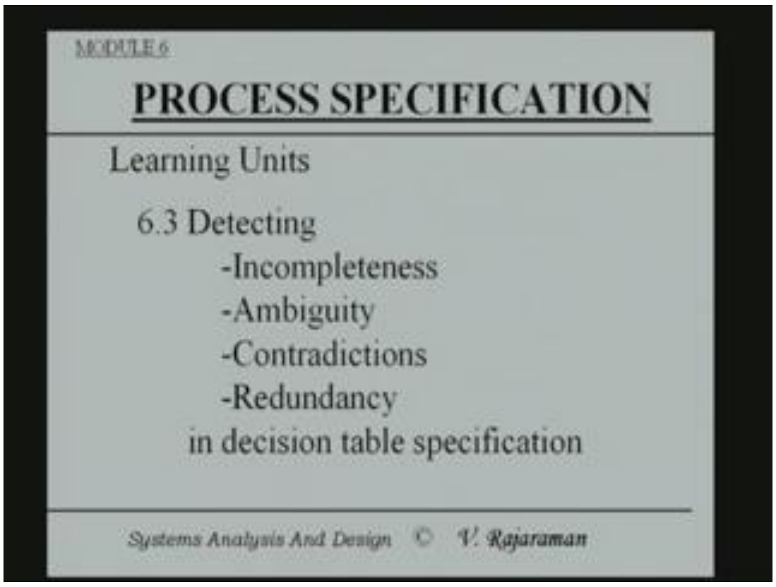
6.2 Decision table based specifications

Systems Analysis And Design © V. Rajaraman

The process specification is primarily what goes in to each of the circles, the DFD each of the circles which are actually a process. So, how to clearly initiate the specification, specification can be written down in playing sentences or a page that does not make much sense, that is what given by the manager to you before you computerize, you have to convert that word statement or see a half a page of description into a process.

So, the structured english is what I am going to use for process specification. And there are certain types of business rules they call it. It is very complex business rules, where the whole lot of if conditions, then, else and so on. And they can become very confusing in a structure english thing, whereas in a decision table they get concise as in a tabular structure, just like tooth table were able to see at a glance. What conditions are applied, what rule apply, what actions you have to take. So, these are the two major ones I am going to talking about in this module.

(Refer Slide Time: 51:48)


MODULE 6

PROCESS SPECIFICATION

Learning Units

6.3 Detecting
- -Incompleteness
- -Ambiguity
- -Contradictions
- -Redundancy

in decision table specification

Systems Analysis And Design © V. Rajaraman

So, when you come up with the process specification, you must be sure the specification is not incomplete, completely is whatever is given and in the write-up that is been followed, and nothing has been left out. And there is no ambiguity, sometimes written english will have some contradictory specifications and those got to be checked before it goes into computer, ambiguity because in fact, human language unfortunately is not completely unambiguous.

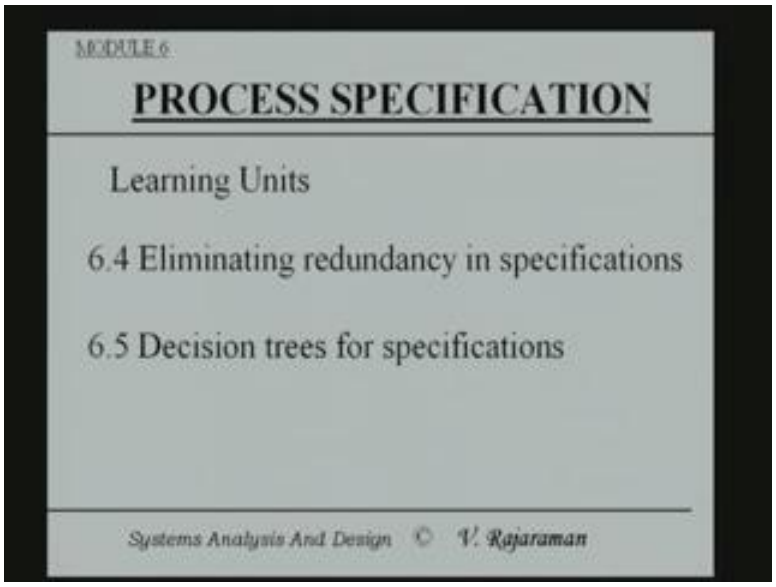
If the human language unambiguous and it can entertain only one particular way, you can owe me any lawyers at all. So, human languages or by nature certain amount of ambiguity. But the ambiguities got to be clear, no ambiguity can occur on the process specification and contradictions, any contradiction requirement you put there, which you should be able to detect and tell the manager if I do, I can't do the other, redundancy unnecessary rules are given, the rules can be much more concise.

Because what happens in real practices, particularly things like insurance rules and many other rules in fact, even government surveys rules and so on the rules are all initiated. And then they find out some problem, in naturally implementing it and then they put add more rules, without really seeing whether new rules or contradictory to the older rules are ambiguous and so on.

And so new rules go on building up and you got very complex set of business rules, which you have all these difficulties and even then it can be incomplete because in a specific case. And it could happen in that because we went on adding as an afterthought, you could have put certain same rules again, because you get remember the rule already existed. So, this got to be done automatically to some extent, the tool will you use should be able to check this also.

So, decision table specification it is reasonably easy, though the checking of incompleteness, the ambiguity, contradictions and redundancy that is not all that easy in the structured english statement. So, the basic rule is to use decision tables, whenever you have a whole set of complex rules and use the structured english, when you have lot of loops, when you have something like programming like structures, do while repeat until things like that, in which case the specification in english is much neater.

(Refer Slide Time: 54:54)



MODULE 6

PROCESS SPECIFICATION

Learning Units

6.4 Eliminating redundancy in specifications

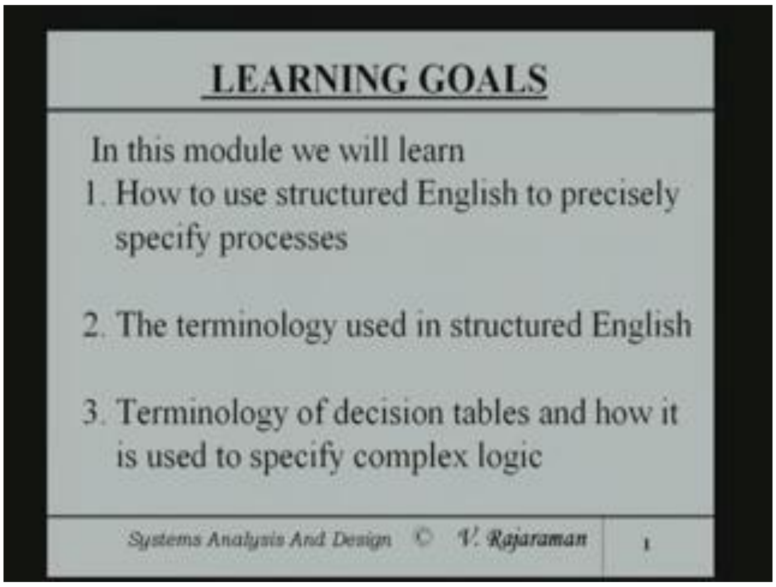
6.5 Decision trees for specifications

Systems Analysis And Design © V. Rajaraman

Eliminating redundancy specification, it is also something called decision trees, which is descended of decision table and we will find out later on, the decision table in a certain strict order, there is no strict order in following conditions. Otherwise, conditions in a table can be carried out in many order, any random order, whereas decision tree in one where, the condition has to be tested in a specific order.

So, tree is something like a, you all know about tree structure, except that the notes of the trees or decisions the arrows are actions. And so gain show of the glance, a tree starting with certain set of decisions. And how those decisions will got broken up into further decisions and how ultimately actions are taken, I will give an example as I go along.

(Refer Slide Time: 55:56)



LEARNING GOALS

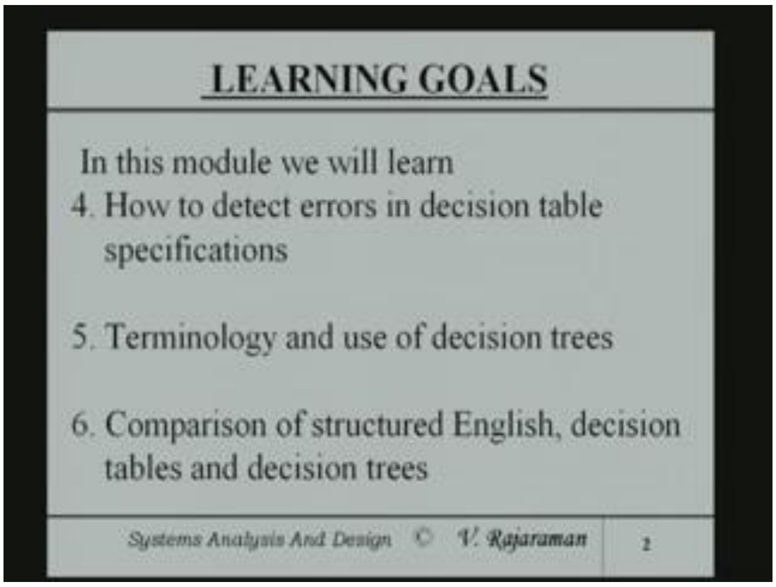
In this module we will learn

1. How to use structured English to precisely specify processes
2. The terminology used in structured English
3. Terminology of decision tables and how it is used to specify complex logic

Systems Analysis And Design © V. Rajaraman 1

How do you use structured english precisely, specify processes terminology you used, terminology used in decision table.

(Refer Slide Time: 56:05)



LEARNING GOALS

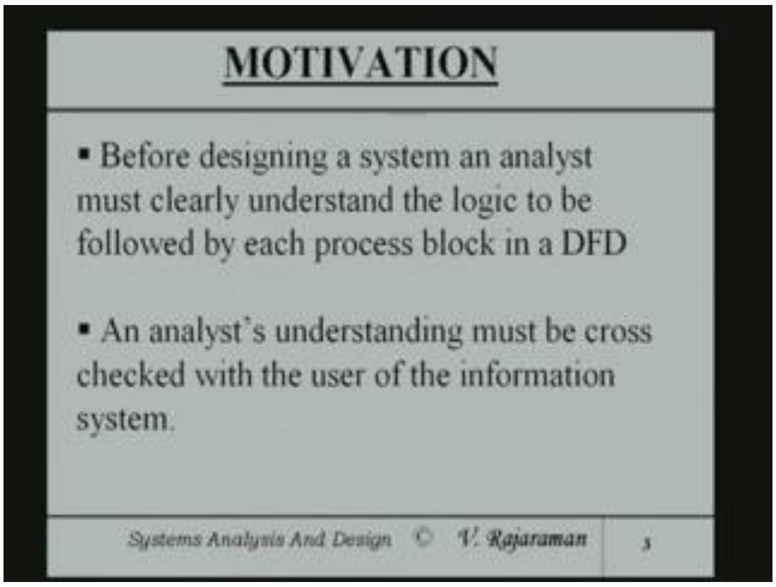
In this module we will learn

4. How to detect errors in decision table specifications
5. Terminology and use of decision trees
6. Comparison of structured English, decision tables and decision trees

Systems Analysis And Design © V. Rajaraman 2

How to detect the errors, decision trees, comparison of structured english, decision table and decision trees, because I am giving three different tools, I should compare all three tools and that’s what I will do at the end.

(Refer Slide Time: 56:23)



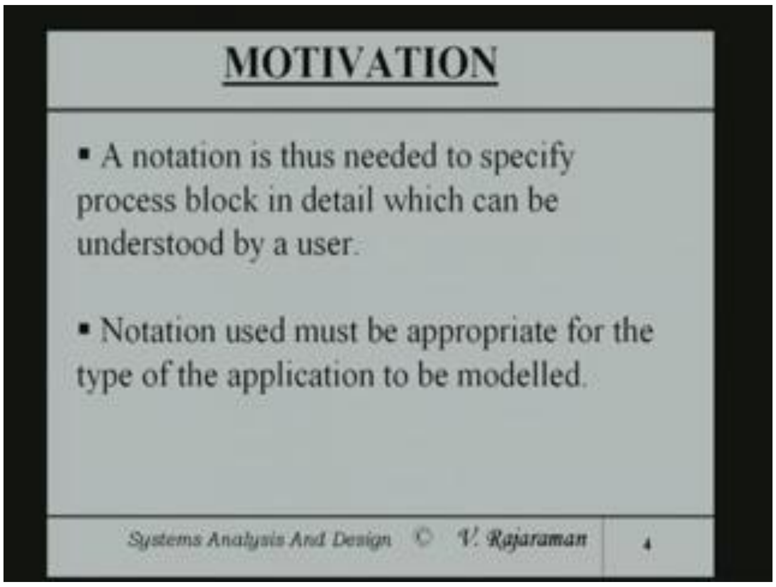
MOTIVATION

- Before designing a system an analyst must clearly understand the logic to be followed by each process block in a DFD
- An analyst's understanding must be cross checked with the user of the information system.

Systems Analysis And Design © V. Rajaraman 3

So, before designing system, the main motivation is that you must clearly understand that the logic I already said that and for each process block of the DFD has got to be expanded. And an analyst understanding must be cross checked with the user of the information system.

(Refer Slide Time: 56:40)



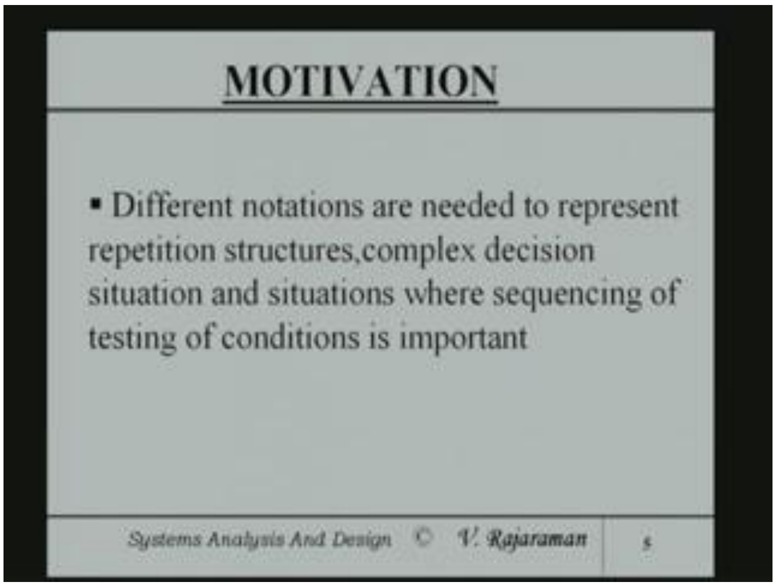
MOTIVATION

- A notation is thus needed to specify process block in detail which can be understood by a user.
- Notation used must be appropriate for the type of the application to be modelled.

Systems Analysis And Design © V. Rajaraman 4

A notation is thus needed to specify process blocks in detail, notation used must be appropriate for the type of application to be modelled.

(Refer Slide Time: 56:49)



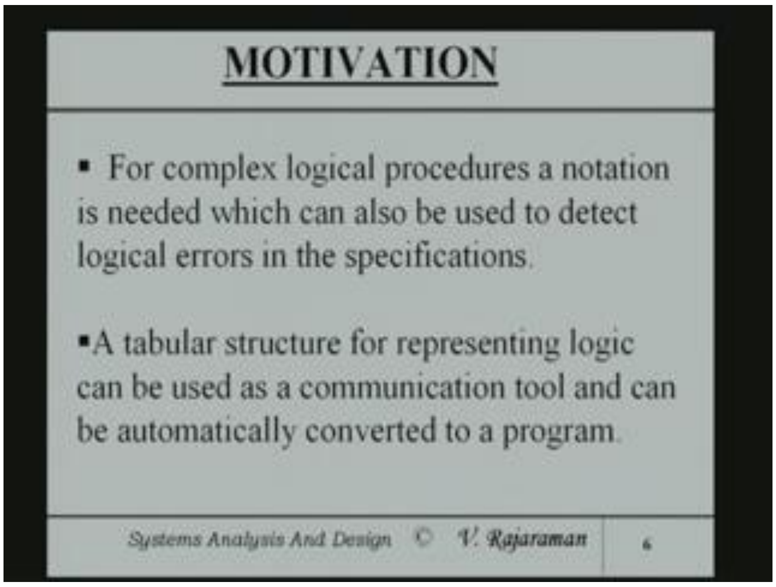
MOTIVATION

- Different notations are needed to represent repetition structures,complex decision situation and situations where sequencing of testing of conditions is important

Systems Analysis And Design © V. Rajaraman 5

Different notations are needed to represent the repetition structures, complex decision situation and situations where sequencing of testing of conditions is important, that's how there are three different kind of tools.

(Refer Slide Time: 57:05)



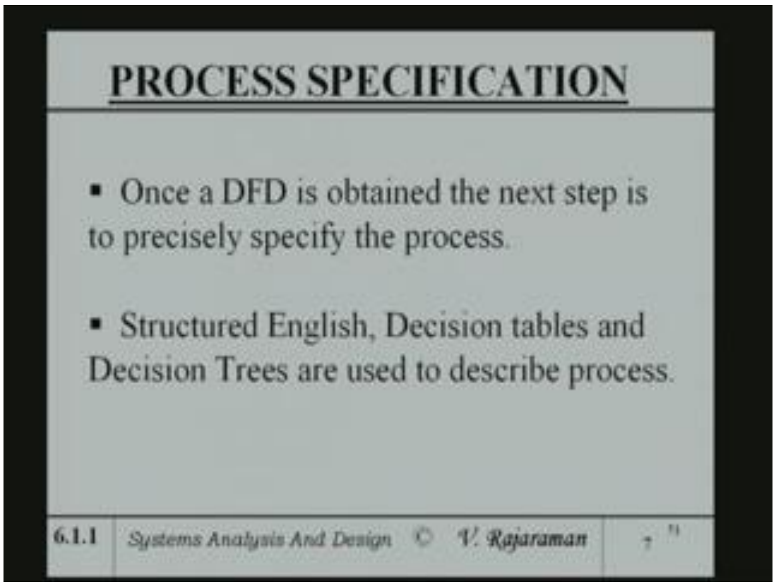
MOTIVATION

- For complex logical procedures a notation is needed which can also be used to detect logical errors in the specifications.
- A tabular structure for representing logic can be used as a communication tool and can be automatically converted to a program.

Systems Analysis And Design © V. Rajaraman 6

As I said for complex decision processes and detect logical errors and so a tabular structure is useful.

(Refer Slide Time: 57:18)



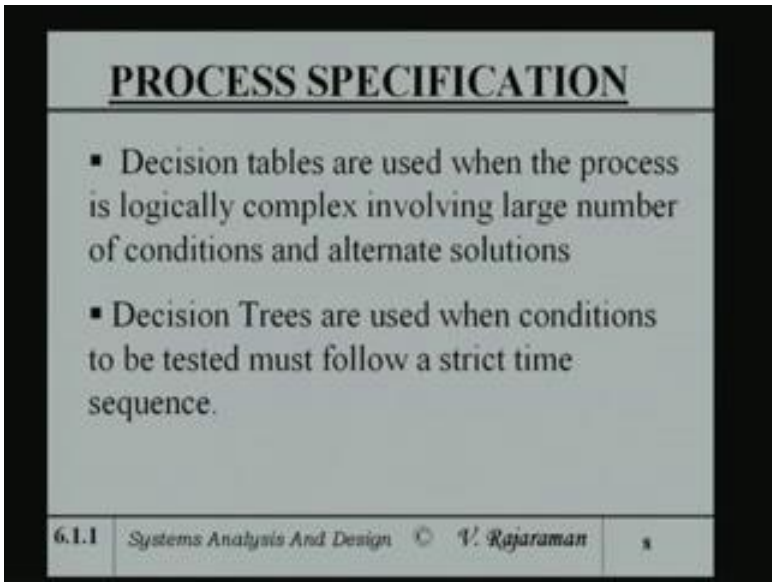
PROCESS SPECIFICATION

- Once a DFD is obtained the next step is to precisely specify the process.
- Structured English, Decision tables and Decision Trees are used to describe process.

6.1.1 Systems Analysis And Design © V. Rajaraman 7

And let us start with the actual process specification once the DFDs are obtained, the next step is to precisely specify the process. Either structured english decision tables or decision trees.

(Refer Slide Time: 57:34)



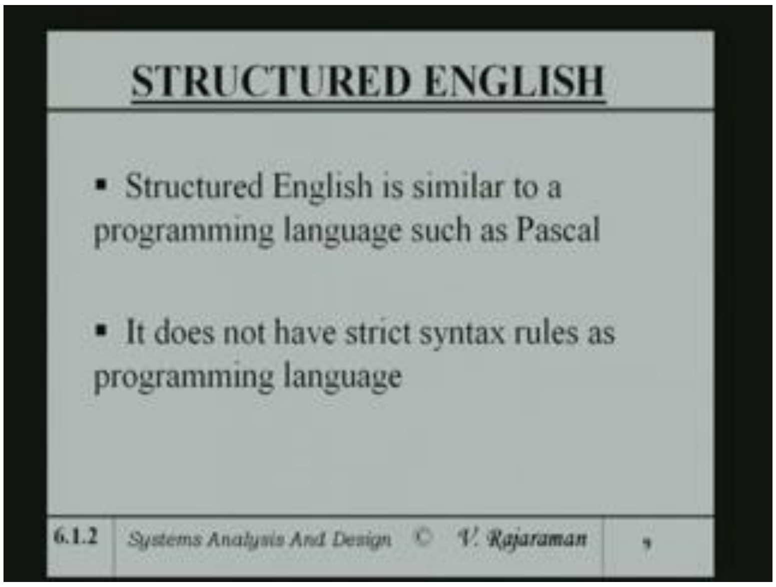
PROCESS SPECIFICATION

- Decision tables are used when the process is logically complex involving large number of conditions and alternate solutions
- Decision Trees are used when conditions to be tested must follow a strict time sequence.

6.1.1 Systems Analysis And Design © V. Rajaraman 8

Again I am saying the same thing, decision tables are used, when the process is logically complex involving, where used and conditions followed certain strict order.

(Refer Slide Time: 57:45)



STRUCTURED ENGLISH

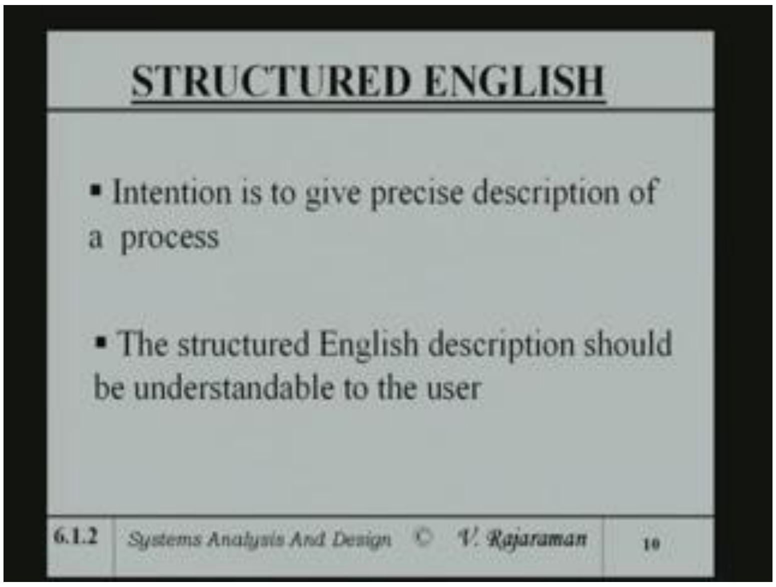
- Structured English is similar to a programming language such as Pascal
- It does not have strict syntax rules as programming language

6.1.2 Systems Analysis And Design © V. Rajaraman 9

Structured english is similar to programming language, like pascal. In fact, it is closer to pascal tendency because C is too concise and it does not have our strict syntax dramatic rules. Other words there is no comma, full stop is got to be compulsory, like you know any programming language, you have to interpreted by a computer, it is very important to have a of proper commas, full stop, dashes, dots and so on at the right place, you don't have it then it is a syntax error.

So, does not have any syntax rules, because it does not have any syntax rules, it cannot be automatically check by computer. It is got to be converted into a program and the program will be checked by computer. But at the time you create the process specification in structured english, you can also create what, test data can be used later on, when you write the program that will be easy.

(Refer Slide Time: 58:53)



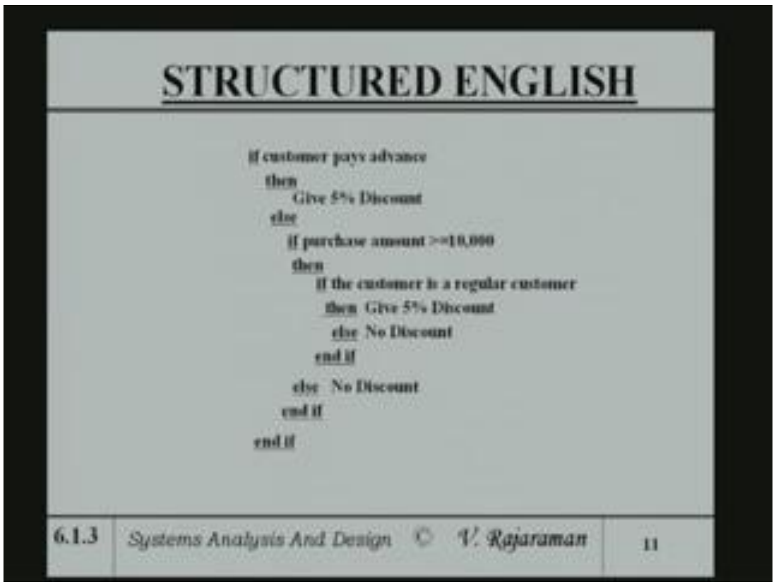
STRUCTURED ENGLISH

- Intention is to give precise description of a process
- The structured English description should be understandable to the user

6.1.2 Systems Analysis And Design © V. Rajaraman 10

Structured english the intention is to give precise description of process. The structured english description should be understandable to the user, you have to give a precise instruction description and should be usable.

(Refer Slide Time: 59:09)



STRUCTURED ENGLISH

```
if customer pays advance
  then
     Give 5% Discount
  else
    if purchase amount >=10,000
     then
        if the customer is a regular customer
          then Give 5% Discount
          else No Discount
        end if
     else  No Discount
    end if
end if
```

6.1.3 Systems Analysis And Design © V. Rajaraman 11

I will give an example, I will give very quickly give an example, customer pays advance then give 5 percent discount, else if purchase amount is greater than 10,000. Then if the customer is regular customer, then give 5 percent discount else no discount and so there is whole lot nested, ifs and else the whole point is that it shows that it gives the rules, a set of rules for giving discount, when a customer buys something from you.

So, based on purchase order, so on and do that. So, next time I will look at this in greater detail, explain this and I will essentially stop at this point and continue in next lecture.