Management Information System
Prof. Biswajit Mahanty
Department of Industrial Engineering & Management
Indian Institute of Technology, Kharagpur

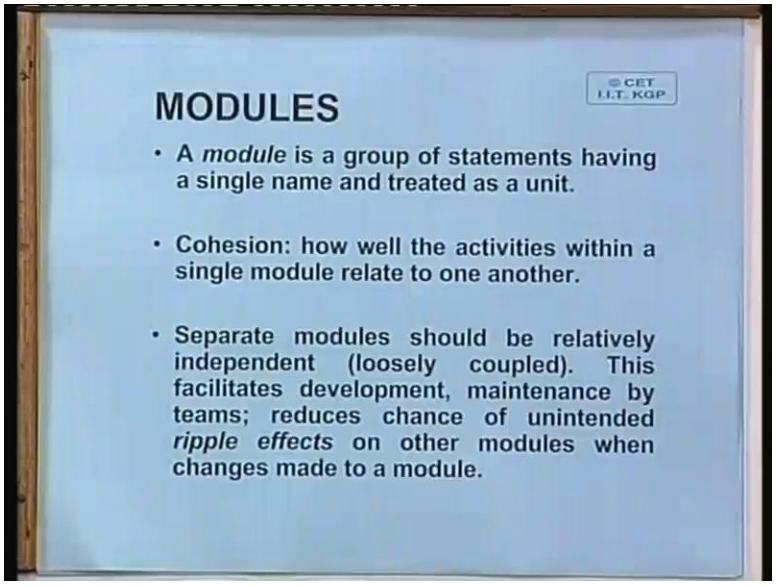
Lecture No. # 21
System Design – II

(Refer Slide Time: 00:49)

System Design - II

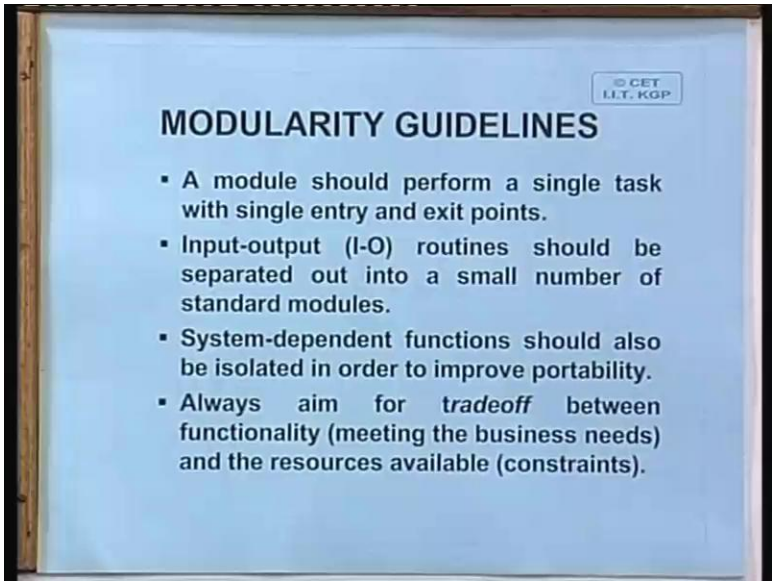
So today let us begin a new chapter on the coupling and cohesion right. So yesterday we had discussed the system design initial things. The basic ideas about what are the basic components of system design. Usually there are three things one is that of user procedures, DBMS – Data Base Management Systems and finally the automated systems. Now in the automated system design basically we are discussing about how to divide the automated systems into well-defined modules. So we are in the middle of this particular slide.

(Refer Slide Time: 01:46)



That is a module is a group of statements having a single name and treated as a unit. Some problem. (( )) (01:57) Okay, anyway you can just read it out I think someone go and tell. So cohesion is how well the activities within a single module relate to one another right. Separate modules should be relatively independent that is loosely coupled. This facilitates development maintenance by teams, reduces chance of unintended ripple effects on other modules while changes are made to a module. This we had already discussed in the previous class right. Now there as certain guidelines which actually decides what should be there in a module?

(Refer Slide Time: 02:40)



So some of them are a module should perform a single task with single entry and exit points, right. That is the first 1 the module should perform a single task with single entry and exit points, right. That means there should not be multiple entry and multiple exits. That means all the inputs and all the outputs should be coupled together and they should be entered at a single point of time. The input and output routines should be separated out into a small number of standard modules. So a basic idea here is that whenever it comes to the, your various input and output handling. Because you see for every software development there comes certain number of modules which will be common to all the modules.

You know something like if there are numbers of subsystems, there will be certain things which will be common to many. So this should be defined in some standard functions or routines or classes or objects or whatever and there should be basically some standardized things. Then system dependent functions should also be isolated in order to improve portability. I have discussed portability in my previous class. Basic idea is that due to many reason you may have to do some platform changes. And this platform changes let us say from may be windows base system to unique base system or even in unique base system. You may have some 1 kind of machine like HP 9000 you may like to go to something like Solaris. So between the sun Solaris

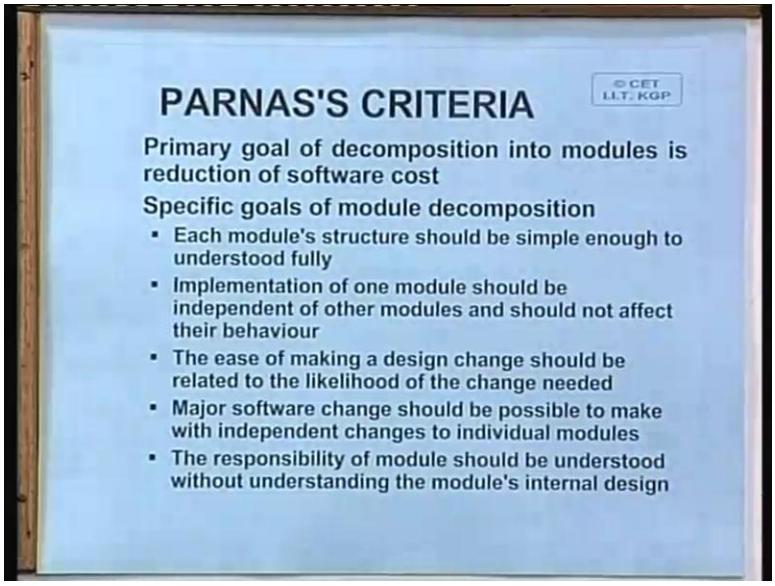
and let us say the HP 9000, although 99.99 percent things should be the same in UNIX and whatever is your programming language may be C plus plus or C.

But there will be at least 1 percent difference. I mean not really 1 percent say 0.01 percent a small percentage would be different and because of these differences those are may be called your system dependent functions. So you should take special care that if you are going to use some system dependent things you put it in a different module. So that when you go to the new system all that may be required is change that particular module and not the entire thing. Always aim for tradeoff between functionality and the resources available. So usually what does it mean that meeting the business needs and the constraints. That means you may not have resource available.

Suppose there are not lot of the hardware that is the required hardware is not available. And you basically require certain things. For example you require a very high speed computation. May be there is an online simulation and you want the simulation to proceed at a very fast manner. Now you may ask that how important is for me to have a very fast simulation is it more important or it is more important to make this simulation available to more number of people. You see many a time what happens that whenever you are running a simulation the simulation is a back end activity all right. It should be it may be run by a group of people at the computer center itself. Whereas the output of this simulation may be the simulation or manufacturing or some other things might be important to a large number of people all right.

So suppose in the name of speed if you buy a very high end machine with very high speed etcetera. Simulation will be carried out very fast definitely. But because of lack of budget or whatever you cannot buy adequate terminals and give it to all the concerned people who actually require the simulation output. Now you have to say what is important to us is this. Simulation running it very fast is more important or giving it to all these people at more important. If you thing giving it to all these people are important from the companies point of view and you can always schedule as simulation runs towards the evening or something like that. You should go for multiple terminals rather than high speed. So this is the basic idea that tradeoff between the functionality and the resources available. And according you decide your modules right which module to develop and which are not to be developed.

(Refer Slide Time: 07:54)



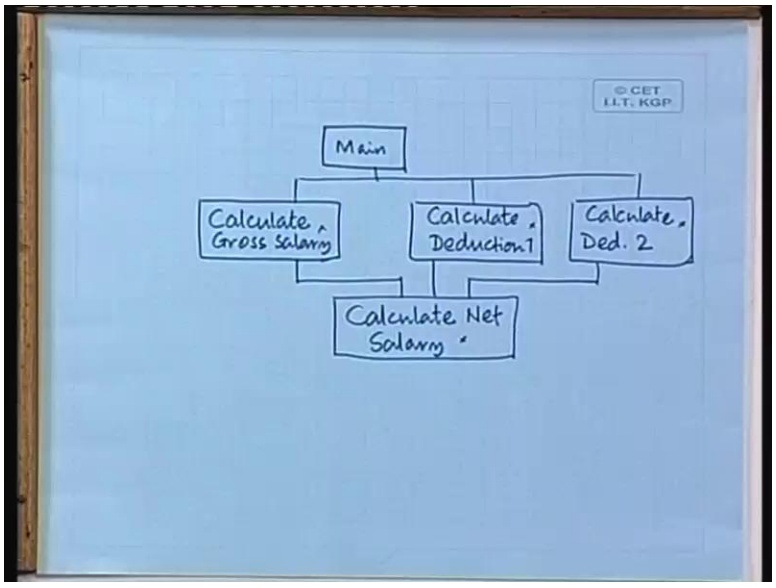
Then there are a set of criteria known as Parnas's criteria. So basic idea about Parnas's criteria, there are 5 basic points for modularization. The first thing is definitely primary goal of decomposition into modules is a reduction of software costs. So one needs to reduce the cost of the software to be developed. But there are some specific goals the first one, each module structure should be simple enough to be understood fully it should be simple. That is the first thing each module should be simple. Then implementation of one module should be independent of other modules and should not affect their behavior right.

So a sometimes what happen that if you modularize in that manner, suppose you want the salary, let us take a simple example of salary computation. In salary computation there are essentially 2, 3 major components. The first 1 is calculating the gross salary. Basically the basic of a person then DNA's allowance then your other pays HRA's and so on this constitutes the gross salary. Then there are the deductions. Now usually what happens there are large numbers of deductions right, large number of deductions say provident fund deductions family pension deductions then there are 100s of things like loan repayments.

If a person has taken a number of loans then repaying those loans or sometimes what happened a person may be member of some clubs or something. Now please try to understand what actually sometime can happen is that you, suppose you make the module like this that 1 module does just the gross salary calculations right. Another module calculates all the details of the individual deductions. say we have small, small modules which calculate all these individual deductions. Then what happens you also after seek? Now you have a large number of modules which are basically utilized for gross computation and the individual details computations right. But individual results are stored in each of them, each of them.

Now suppose you have to define a module which will calculate the net salary. If you do not think about a something like a database where each module would put their data and basically you would read from the database. Suppose you do not want, that is sort of a design and you have a design where a person has to read through all the modules and collect output from each of these modules. All the modules go to a single module and these final modules calculates net salary. I do not know whether you are clear about this or not basically what I am trying to say is something like this.

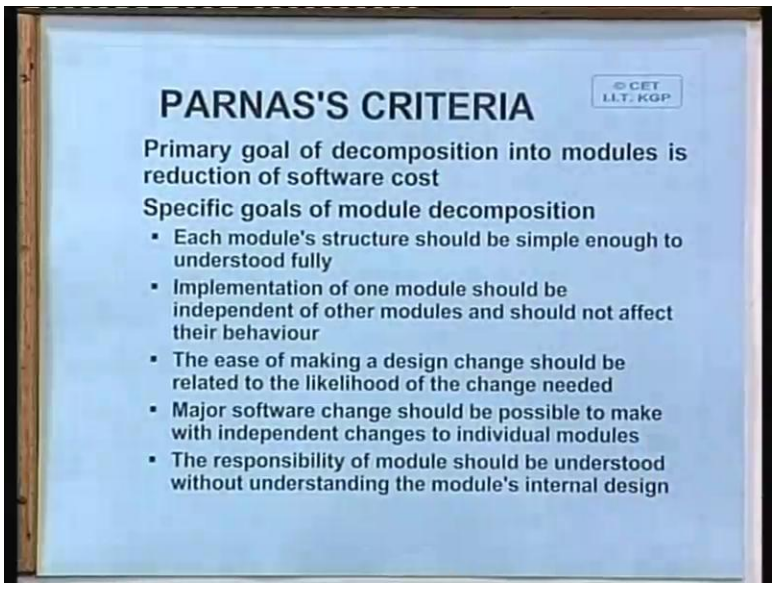
(Refer Slide Time: 11:37)



You see suppose you have a structure of this type. Now see this calculate net salary if you have to do this the way it is shown here then you have the gross salary coming from here. You have the deductions coming 1 coming from here deduction two coming from here, deduction three coming from another, then suppose due to whatever reason the net salary is wrong. Now the error is need could be here please understand error could be here or in any of these modules the problem may not be here problem may be in any of these modules. So it will be very difficult to understand where the problem is not only that the person who has to write this module has to understand each of these top modules all right.

So this is the difficulty a much better design would be that individual calculations of gross salary or individual deductions should actually go to the database all right. Then from the database you can obtain all the relevant details and calculate the net salary. So what actually happens is that when you are calculating your net salary you need not understand what is wrong in the others. You simply know that this is my data store I collect the data from this data store and put it back to the data store right. So in that sense the modules are independent of one another.

(Refer Slide Time: 14:51)



Then the ease of making a design change should be related to the likelihood of the changes needed. So basically what it basically says again a something to do about functionality. It usually what happens that ah whenever you are going for the design change all design changes are not equally important all right. So there is a word if you recall, I have said write in the beginning that is requirements gold plating gold plating the basic idea is that see there is nothing like 100 percent requirements fulfillment. You cannot do there is you can maximum do 99.99 percent not 100 percent there will be always some more requirements which you could not meet. So it is necessary for the requirements of that of the design changes to be hierarchically sorted out according to their importance, according to their need and try to focus on those design changes which are required the most.

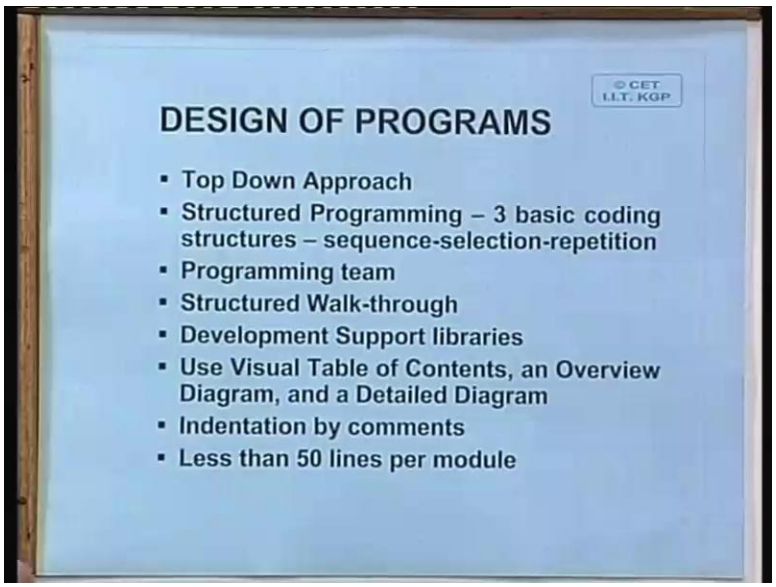
Then major software changes should be possible to make with independent changes to individual modules. So this is important. See ultimately the your complete software. Suppose you have to make major changes. Now conceptually whatever you have done you have done but end of the day the changes should be coming out of individual module changes. Again the idea is the one and the same that how do make use of more number of people? How to improve productivity? The responsibility of a module should be understood without understanding the modules internal design. I think you understand the difference between responsibility and internal design what is the responsibility here has a very specific meaning the basically the meaning of responsibility is that what the module is supposed to do? What are the messages that the module need to pass to other modules all right.

So suppose 1 module is connected to two three other modules. Then what messages or data should be passed to those other modules and why this message or data is to be passed so that the other modules can perform something. So in that sense this is the responsibility of this module is it clear. So this is the idea. So basic meaning of this particular word is what a particular module should you know send data or pass messages to other module should not be suppose you want to understand this. It should be possible without going into the internal program logic of that particular module. Is it not? See in today's world we talk about ah see basic idea it can be understood this way. Suppose you have a TV remote all right. Now this TV remote the technology of this TV remote most often has nothing to do with the design of the TV you can

change the Television you can put another Television which internally may work totally differently.

The technology is totally different even how it handles that remote signal is totally different the earlier Television where handing the remote signal in 1 way right. The new Television is again handling the same remote signal; but in a different logic by using a different technology. But you do not have to worry about this. Even if the remote the technology within the remote control, may also change all right. How it generates signal and when you press a button 1 what happens inside you need not know. All you need to know is I press 1 the TV comes on all right. So this is the responsibility part it sends a message that message has something to encode as 1 is it clear. So this is how it can be done. This concept will be again, we shall come back to this when we go back to the object oriented programming an object oriented analysis and design.

(Refer Slide Time: 20:00)



Now before we move over to coupling and cohesion one or two simple things first thing. There are certain basic guidelines of whenever we design a program which are to be followed. The first 1 is it should be a top down approach top down approach top down approach means basically there are two approaches. Whenever you are designing a program basically it could be top down or it could be bottom up right. So bottom up means you try to look at the program from just look at the individual tasks all right. But if you do like that you develop lot of individual programs.

But when you have to combine them all you find that it is not possible to put them together. It is difficult to combine the programs into a single cohesive application.

Why because the interfaces will not match because changes in 1 program might affect another program in an adverse way all right. It will not be an integrated development. Then structured programming. Basically what is known as structured programming is there are three basic coding structures sequence selection repetition. I think you know that we have also discussed sequence is that the statements are executed 1 after another selection if then else kind of logic or switch kind of logic and repetition looping, do while, etcetera. While for then there has to be a programming team there has to be a programming team usually there are we have discussed I think there are basically chief programmer team and democratic setup egoless programming etcetera.

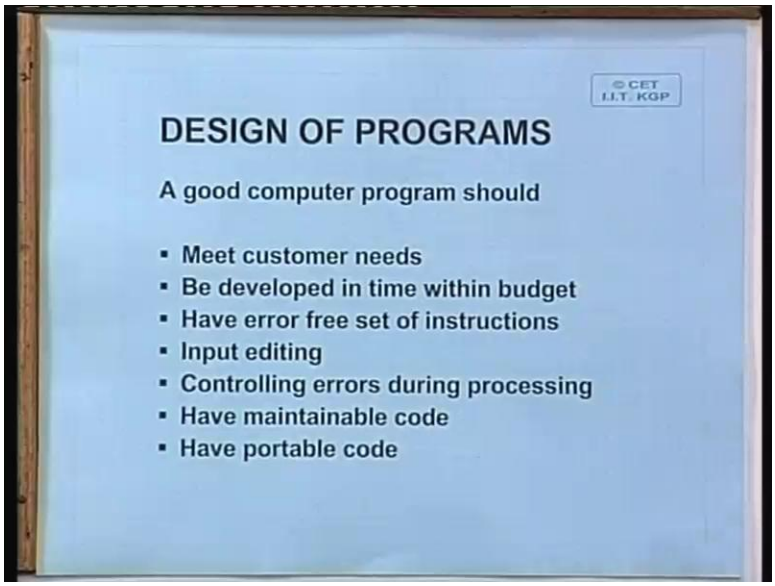
Then we should have a very important thing a facility for a structured walk through a structured walk through is a very important quality tool for any program development. What happens whenever you have designed a module or a program there should be regular quality inspections and usually who should form these quality inspection teams? It should be the developers themselves a group of quality experts, the project leaders, etcetera. They should go through the code and try to understand whether there is some errors right. So sometimes what happen, the error may not be within the program or looking at it from that point of view really you do not understand where is the problem. See basically a work can be done in 100 ways, all right.

Now it is only a given way which is going to be amenable to all the other work right. So if you are doing in 1 of those 100 ways you feel you are doing it correctly is it not. But problem is when you go to the suppose you know you are handling some 100 papers all right. Which has numbered 1 to 100? Now you can put it serially you may not put it serially or you may put it serially but backwards. So when you put it backwards you have you have sorted it is not but backwards but the next operation requires it to be sorted from 1 to 100. What will happen the next operation will suffer because it does not know how to handle backwardly sorted numbers. It expects the numbers to be coming in 1 to 100 sequences, so it is a mistake in a sense.

See it is not really an error as far as you are concerned you have sorted it. But the next program will face difficulty. So these kinds of errors will also be seen in the structured walk through basic. This is basically kind of inspection only but is an inward one all right. Then there should be development support libraries there should be two types of libraries. One is basically for enhancing the knowledge of the developers and the second 1 basically where the standard libraries or standard functions standard classes objects which you can borrow to your application. Then there should be a visual table of contents an overview diagram and a detailed diagram. These are actually what is known as hypo hierarchically hierarchical input process output charts hypo charts.

I will not go into detail of hypo charts basic idea about hypo chart is that it shows the input process output. What are the inputs? What is being processed? What are the outputs in detail? Right. So when you show them in a summarized form you can call it an overview diagram. When you show them in great detail you can call it a detail diagram then there should be indentation by comments very important without proper indentation the program is unreadable and finally there should be less than fifty lines per module. There are some specific requirements about a good computer program. A good computer program it should meet customer needs.

(Refer Slide Time: 26:16)



It should be developed in time and within budget. It should have error free set of instructions right. It should have a facility of input editing. There should be control of errors during processing errors and it should have maintainable code. And finally it should have portable code right so good computer program should have the following things. Now certain important things. I mean it should meet customer needs and developed in time and within budget should have error free set of instructions. Now what is input editing what is input editing input editing is like called input validation? I think I have used the word validation a number of times. The basic idea of validation is that you see every field every field suppose a particular object like a student a student has got a number of fields is it not or data items attributes key elements.

So suppose it has a roll number it has a name and hostel and so on and so forth. Now each of these particular field or data items has got a domain of values what is meant by domain of values. Say for example your roll number. Any number cannot be your roll number all right. The roll number should be now how many digits 7, 8? 8 digits. So it should be an 8 digit number right and these 8 digit number out of that the first two digits are supposed to be the year of joining. Now since the present batch of B-Tech students should not have joined I think reasonably beyond 97, 98 right. So if someone's roll number who is supposed to be a B-Tech student begins with something like 96, 95 or 85, it should be a wrong one right. So it could be a kind of validation right. It should be a kind of validation. That means we can always specify that the roll numbers should not begin with no lesser than 95.

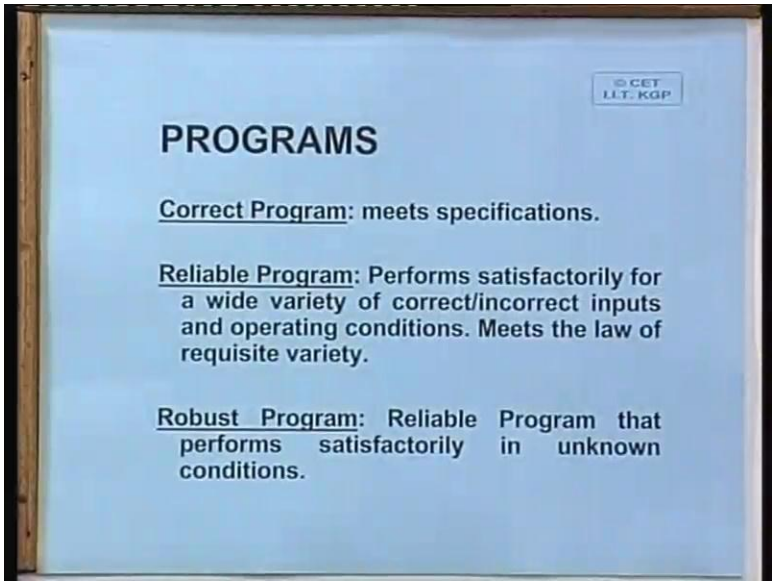
Let us say or no greater than 0 3. So the number should be 0 basically 0 4 to 95 is ruled out the first two digits should not be between 0 4 to 95 is it not. So 96, 97, 98, 99, 0 0 0 1 0 2 0 3 are the allowed numbers. Now we can again do some more checks for example. If the student is in second year then we can again make some more validation all right. So these all are part and parcel of input editing so whenever you are receiving some input, the input should be edited. Now what happens the input editing was a very important thing in the earlier day cobalt programs or FORTRAN programs or even C programs. But with the advent of database management systems, this has been tremendously reduced. What has happened, nowadays most of these input editing is should be taken care of by the DBMS itself right.

But however even if you have a DBMS you have to enter the data into the DBMS all right. The data the DBMS, the data should actually first you have to fill it up only then you can be used right. So at that time again input editing is going to be very important. Then sometimes during processing also you can control errors. You can control errors. See many at this is why it is always better to include little bit of redundancy little bit of redundancy right. Suppose you are giving your bills say TA DA bills, suppose you have gone out for some trip for let us say 1 week and you have given a bill. Usually those bills are given first of all day wise and then item wise all right. So day 1 travel then local travel food etcetera. Day two travel local travel food etcetera, etcetera.

So what may happen you can always compute a day wise total and you can also compute item wise total right. So how much you have spent on Monday? How much you have spent on Tuesday, Wednesday, Thursday like that or otherwise how much you have spent on travel how much you have spent on local travel how much you have spent on food etcetera. Now if you sum it over the days or if you sum it over the item that is you get the grand total that is the final amount you are claiming. It should be same both ways all right. Now if it is not same that means 1 of the calculations have gone wrong is it okay. So these are called true figures or basically these kinds of calculations can check what is known as processing errors. Is it okay? And I have discussed about portability and maintainability also should be very important how to improve maintainability by improving visibility.

What is visibility so that the program is very easily readable anybody reading the program should understand the code all right. How to do this? By first of all proper indentation; indentation, gaping should be properly oriented all if then block should begin with similar positions all right and comments and very important thing naming of variables. Most of us when we write a program we do not give much importance to the variables naming of variables. But is an extremely important thing. The variable name should be exactly what the variable is and not any fantasy name that you may think of at that point of time. Then there are three important categories in which we can classify program. One can be called a correct program, reliable program and a robust program.

(Refer Slide Time: 34:14)



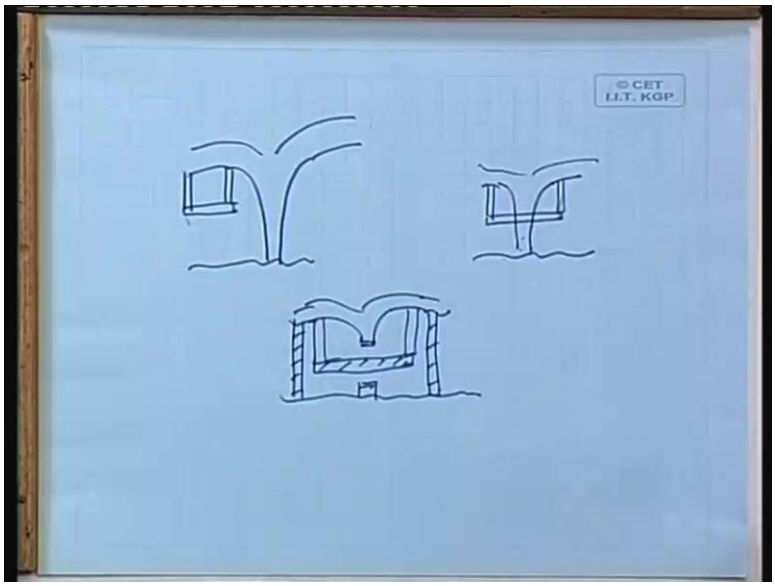
See these concepts of correctness reliability and robustness is universal is not only for programs. It could be for all most any other concept. Say you may say the design correct design reliable design or a robust design all right. Usually the word correct is used whenever the particular thing meets the specifications all right something that meets the specifications is a correct program all right. The reliable program on the other side it should perform satisfactorily for a wide variety of correct or incorrect inputs and operating conditions. So what is a reliable program reliable program it should perform over a wide range of inputs and operating conditions? But please understand it is still within the system within the system you can vary the input you see something like suppose you have an electric blub all right.

Now these electric bulb is most often until unless some very peculiar thing happens is burning at may be 200 to 20, 260 volts all right. And with normal operating conditions all right. There are not much of other external disturbances the voltage is steady and not too much of fluctuations all right. But you might have rated that particular bulb for 140 to 260 or 140 to 300. So whether the bulb is reliable or not it has to be tested at 140 at 300 and at all possible voltages all right. The concept of robustness is reliable program that performs satisfactorily in unknown conditions. See this is called a robust design usually what happens suppose a human being a human being.

Suppose we say that human being can withstand temperature between let us say 0 degree to 50 degree. But suppose you have been put in 55 degree. What will happen? You will not melt. You will be you will be uncomfortable you will feel very bad. But at the same time you will still exist. But suppose your bulb which is rated at 140 to 300. Suppose you give 301 or 305. And if the bulb immediately goes out you cannot blame the company because the bulb was not supposed to run at 305. It is supposed to run up to 300. But it does not mean movement it becomes a little higher than 305. It should immediately go out is it all right. So that is usually this is a Japanese concept that even in situations which is beyond your operating conditions it should not behave in a peculiar manner.

I mean it should not simply go out just because you have been out of range it should still show a reasonable behavior like what happens to human being. So that is the kind of expectation all right. Now let me give an example. I think I am late today. But still these example should be given. I feel this is about the difference between correctness and reliability and robustness. Now I think I have discussed little bit on what is reliability and what is robustness but not much what is actually correct; the correctness. You see this is about a very simple example about the design of a swing all right. So specifications given was that suppose I have a tree here.

(Refer Slide Time: 39:04)



I have a tree here and basically. This is what was expected this is what was expected that in this tree you put a swing all right. But it was it was given in a different way. See this is a diagram, so you all understand what it is but when the actual the specification document was prepared it was said like this the swing should be made of wood all right. It should have supports supporting the wood in such a way that it should not fall it should be put on a tree branch which is sufficiently strong and finally it should swing all right. So what happens the company made the swing all right they made the swing and everything was done very nicely. But it was a little bigger it was a little bigger so what they did they put it on the tree in this way actually the tree was not very big so they could not put.

Because when they try to put it on this side of the branch then they find that branch is not strong. So it could not be done so they did it like this all right. So this is how they put the swing. Now see they were claiming this is a correct swing why it is correct because it is put on branch which is strong enough this is made of wood and these supports are strong enough to hold a person who would like the swing. Then the people said no. It is not correct swing because it is not swinging. While I tried to swing it hits the tree they said okay. I will do fix that also but you may have to spend little more. Okay no problem some more money is released so a very igneous way they have done this.

You know what they did they cut the tree into two parts all right from the then they have to build two more supports so that the top portion of the tree still remains and the swing is mounted in between it is swinging. You see it is swinging and the tree is cut and these supports are required so that it can be supported. So this is an example of a correct swing all right. The swing is definitely correct, definitely correct. Because it serves all the specifications all right. But it definitely not robust or reliable although see these looks like an extreme example. But unfortunately many of the programming or many of the application software solutions that we give in real life may be like this, right. Unfortunately I cannot begin the coupling and cohesion part. We shall do it on the next class.