Software Engineering

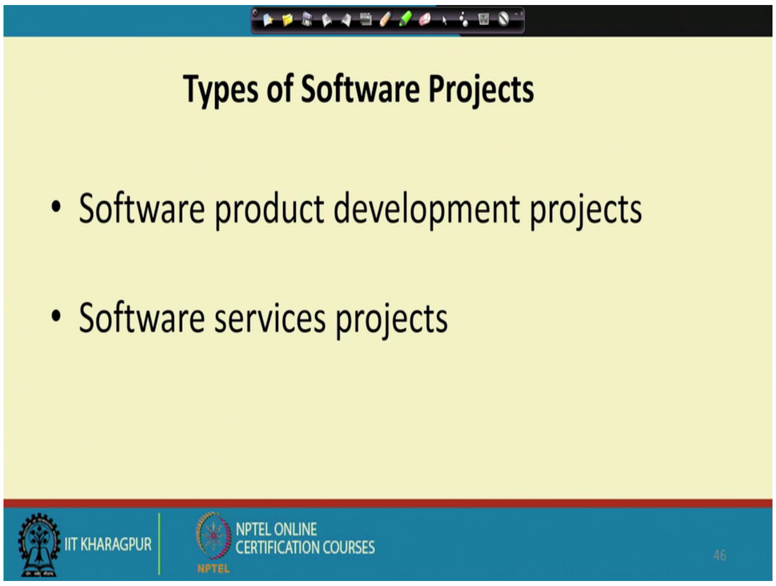
Prof. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 04
Introduction- IV

Welcome to this lecture. And in the last lecture we were discussing about the types of software projects. Before we look at the software engineering principles we want to form a broad idea about the types of software projects that are been undertaken in the industry, so that we will be able to appreciate the different principles and the principles which are used for which type of software development and so on.

We were saying that there are two main types of software projects.

(Refer Slide Time: 00:58)



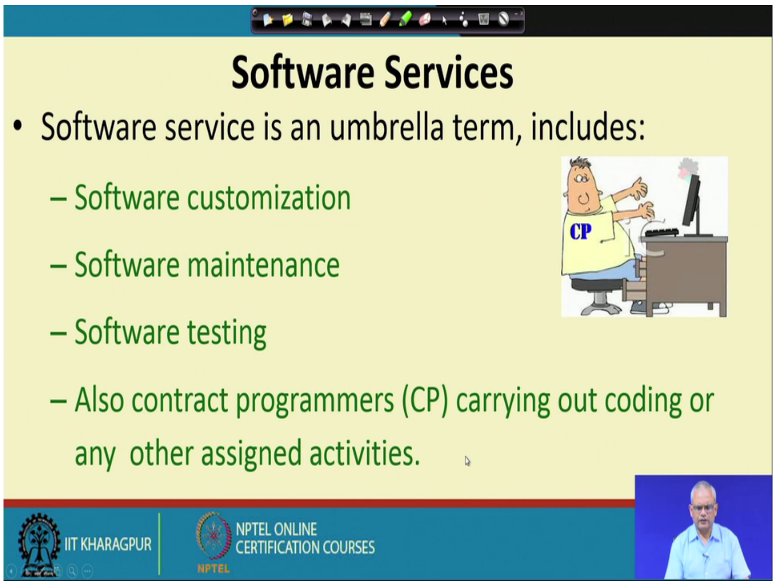
Types of Software Projects

- Software product development projects
- Software services projects

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL
46

One is the product development projects and the other are services projects. These are the two broad category of software projects.

(Refer Slide Time: 01:08)



Software Services

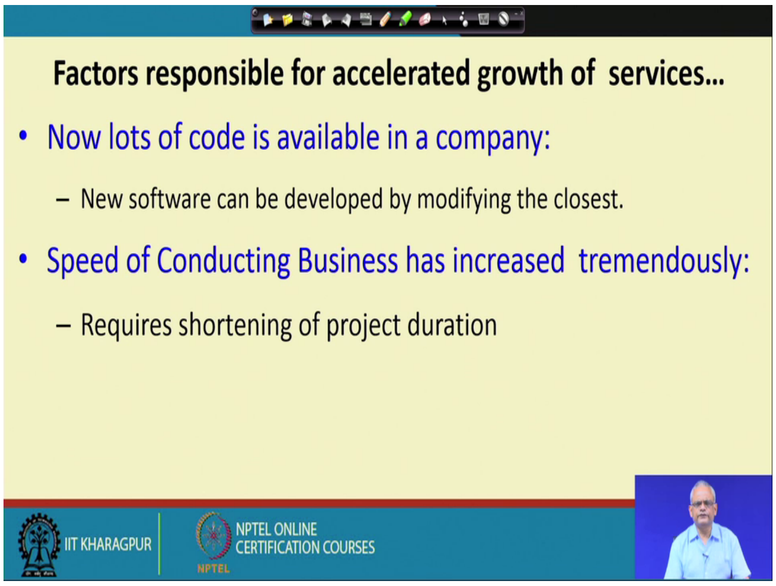
- Software service is an umbrella term, includes:
  - Software customization
  - Software maintenance
  - Software testing
  - Also contract programmers (CP) carrying out coding or any other assigned activities.

CP
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

First let us look at software services. We said that software services is growing very fast. Right now, the market size for software services and the products are evenly balanced, but then the software services projects are growing much more faster rate. The software services project is basically an umbrella term, in the sense, that a various types of software services projects. For example, we have software customization, some software needs to be changed little bit to fit the requirements of some customer. Software maintenance: so this is also a services some software has developed, but then an organization has a project to maintain this, a team is dedicated for maintenance for some project some software. And this is another type of software services a software maintenance

Software testing: may be some organization developed a software, but then the testing part has outsourced where testing to an organization which will do only the testing. So, here the software testing is software service. In another extreme form we might have a company which just supplies the contract programmers. And a development company will need some of these for on temporary basis for carrying out some work and this is also a service.

(Refer Slide Time: 03:07)



Factors responsible for accelerated growth of services...

- Now lots of code is available in a company:
  - New software can be developed by modifying the closest.
- Speed of Conducting Business has increased tremendously:
  - Requires shortening of project duration

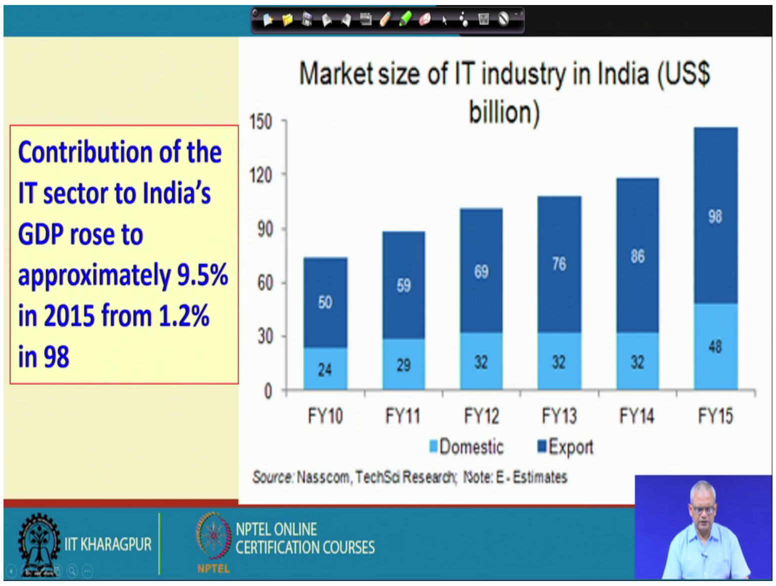
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Before we proceed let us see little bit about why is that the services segment is growing very fast. The number of services projects is large as compared to product development projects. Earlier 50-60 years back we had very little code written. So, for meeting the requirements of any customer needed to develop software from scraps. But now, every company and even otherwise there are large number of projects that have been completed lot of code is available. So, to develop a new software lot of reuse made from the existing software. Not only that lot of software is available, but also right now the development work needs to complete very quickly.

Just to give an example let us say an educations institute decides to automate various book keeping activities. If it was in the 1950s or 60s it would our project to a company and that will take a year or two to develop the product: to develop the software and install at the educational institute. But right now, a month is an average time in which such a work can be completed and even 15 days. So, the project duration has shortened and in 15 days somebody cannot really develop a million line code, has to basically tailor some existing software.

The speed of conducting business has increased tremendously not only education institute but all companies they want software to be quickly developed and installed. And this can only be done by tailoring some existing software. And that is another reason that the services projects have really increased.

(Refer Slide Time: 05:38)


Contribution of the IT sector to India's GDP rose to approximately 9.5% in 2015 from 1.2% in 98

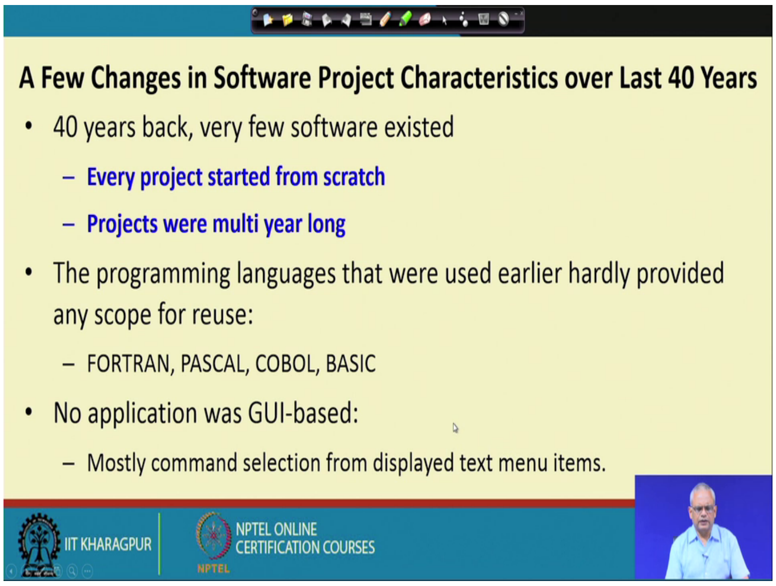
Market size of IT industry in India (US$ billion)

	FY10	FY11	FY12	FY13	FY14	FY15
Export	50	59	69	76	86	98
Domestic	24	29	32	32	32	48

Source: Nasscom, TechSci Research; Note: E - Estimates

If you look at the market size of the IT industry in India it is grown rapidly over the last few years this is a repeat from the NASSCOM. And if you look at here that the export segment is larger than the domestic segment, and right now almost 10 percent of the GDP comes from the IT sector and it was only 1.2 percent in 98 and before that it was near to 0 percent. So, the IT industry in India has made rapid progress, large number of projects are existing, large number of projects have been completed, and this is a growth area, and already 10 percent of the GDP is from the IT industry.

(Refer Slide Time: 06:49)


A Few Changes in Software Project Characteristics over Last 40 Years

- 40 years back, very few software existed
  - Every project started from scratch
  - Projects were multi year long
- The programming languages that were used earlier hardly provided any scope for reuse:
  - FORTRAN, PASCAL, COBOL, BASIC
- No application was GUI-based:
  - Mostly command selection from displayed text menu items.

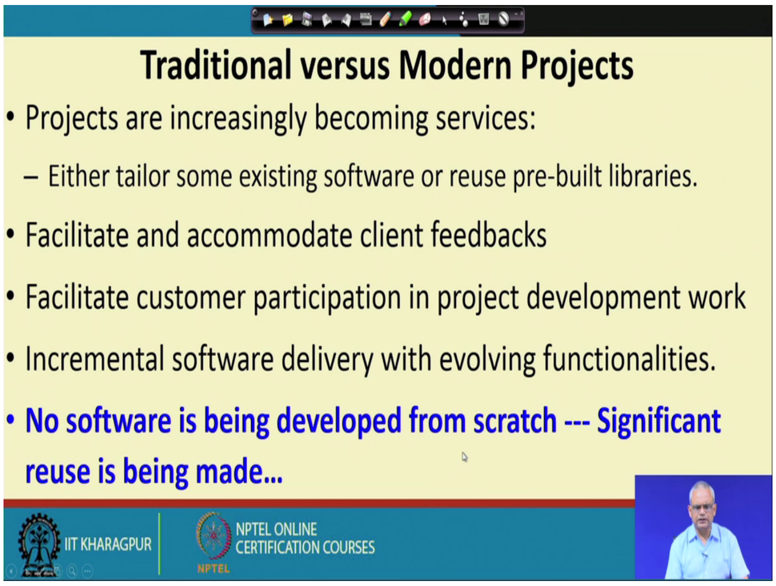
But then before we proceed looking at the software engineering issues let us just have a small idea about the scenario of the Indian Software Companies. One thing that is easily noticeable is that the Indian Software Companies had excelled in the services segment almost every project is a services type of project. But then, would like to ask you this question that why is it so why there are very few product development projects. Because, we said that even now worldwide 50 percent of the development projects are product development and other 50 percent is services, but in India almost hundred percent is services project, what can be the reason.

The reason can be that: in product development system there is a large gestation and risk you develop a product, and then it may do well in the market or it may just flop; lot of investment just may vanish. But then if a product development project succeeds then it keeps on giving return year after a year, whereas the services project is typically paid only once. But then, the Indian companies possibly a risk hours and they concentrate only on the services projects. There are few product development projects and few success stories about products in the Indian context.

Now, let us see how the projects are changed over the last 40 years because, if we understand the characteristics of the projects, how they have changed; we will be able to appreciate what are the techniques that were applicable earlier in the early states like it is in 1950s then 60s. And, at the present time what are the principles and techniques that are applicable and how these are changed.

Long back, in the initial years of software development very few software existed. Almost every project started from scratch and projects were multiyear long 2 year, 3 year and even 5 year projects existed. And the programming languages that were used were FORTRAN, PASCAL, COBOL, BASIC, etcetera; and these programming languages are very little facility for reuse of code. And almost no application was GUI-based, but now almost every application has a GUI and heavy reuse is made. The project duration has really certain from multiyear to just couple of months, and the programming languages at the present time support reuse of the code.

(Refer Slide Time: 10:23)



Traditional versus Modern Projects

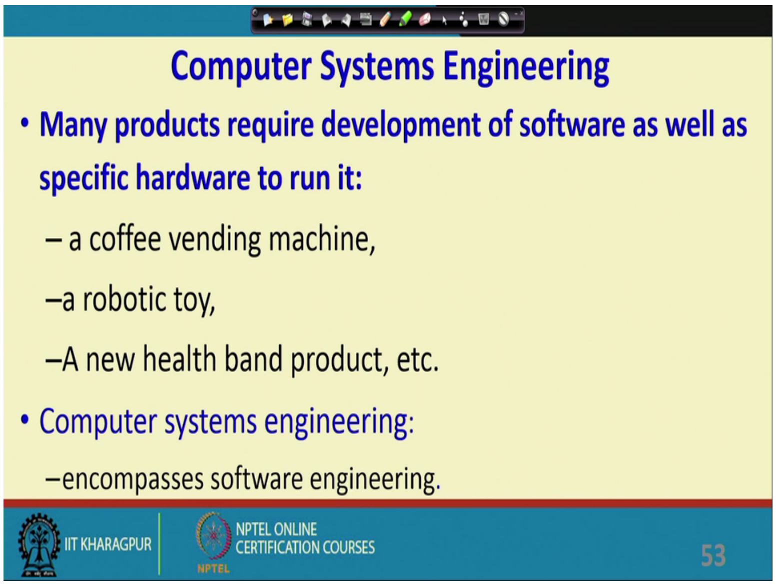
- Projects are increasingly becoming services:
  - Either tailor some existing software or reuse pre-built libraries.
- Facilitate and accommodate client feedbacks
- Facilitate customer participation in project development work
- Incremental software delivery with evolving functionalities.
- **No software is being developed from scratch --- Significant reuse is being made...**

Now, let us look at the traditional versus modern projects. Services segment is becoming having a large growth rate, here we tailor some existing software or reuse prebuilt library. In the modern projects the client is a part of the project, the client gives feedback and the project and the project they try to accommodate the client feedback, but earlier that is the traditional projects. There were product development projects largely and there once the development starts no feedback is taken, they freeze the requirements and start according to that.

Right now, the customer is part of the project development. Even for very large software the software is developed incrementally. That means, that every 2 week or so the software is installed at the client and then it evolves new functionalities are added every 2 weeks or so.

No software is being developed from scratch and significant reuse is made.

(Refer Slide Time: 11:57)



Computer Systems Engineering

- Many products require development of software as well as specific hardware to run it:
  - a coffee vending machine,
  - a robotic toy,
  - A new health band product, etc.
- Computer systems engineering:
  - encompasses software engineering.

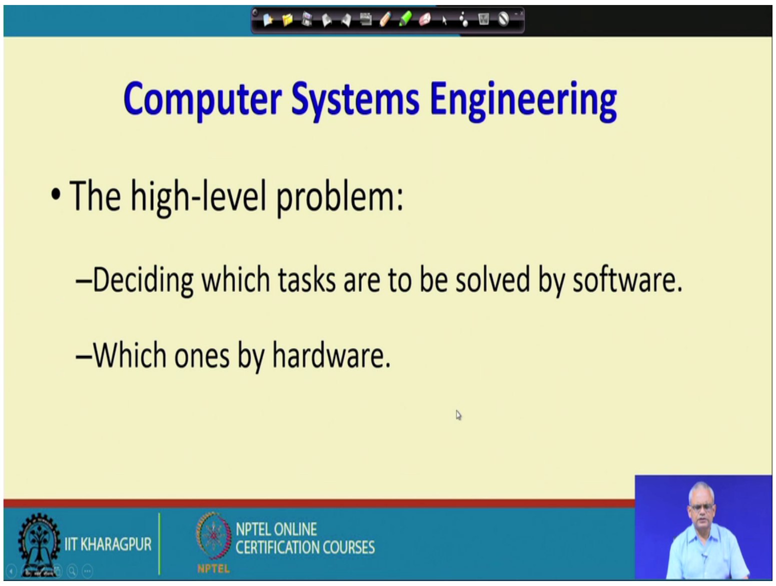
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES 53

Before we look at the software engineering principles, we are look we are concentrating on few basic aspects. Now let us try to understand about what is computer systems engineering.

In many development work the hardware and software are developed together, it is not that we always develop a software and then make it run on a computer like a desktop or a server. No, there are many systems where the hardware also has to be designed from fresh and the software will run on that hardware. Just to give some examples: let us say a coffee wending machine or a robotic toy. So, here the robotic toy has a special processor, it has mechanical parts and so on. And the software will has to run on that specific processor; these are not general type of processor.

So, here we cannot just write a program on the desktop and then we will just get done with it, because the software has ultimately to run on that processor and this robotic system or let us say a new health band product. So, there is a processor and the health band which monitors the health and displays various suggestions and feedback. For this type of system where there is a hardware that needs to be developed and also the software the software has to work on that hardware we call it as System Engineering. It is actually a superset of the software engineering.

(Refer Slide Time: 13:54)



Computer Systems Engineering

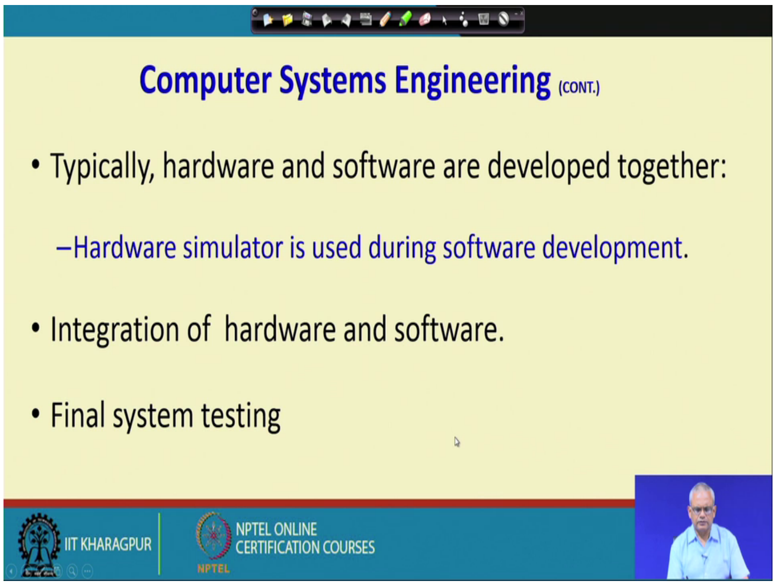
- The high-level problem:
  - –Deciding which tasks are to be solved by software.
  - –Which ones by hardware.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

But how is the computer systems engineering problem solved.

Here first, a high level understanding is of the problem is made, then decision has to be taken which part to be solved by software and which part to be solved by hardware, and then the software and hardware development start parallelly. But the only problem here is that how does the software will be tested, because hardware is also under development; possibly by using a simulator or something the software can be tested and developed.

(Refer Slide Time: 14:35)



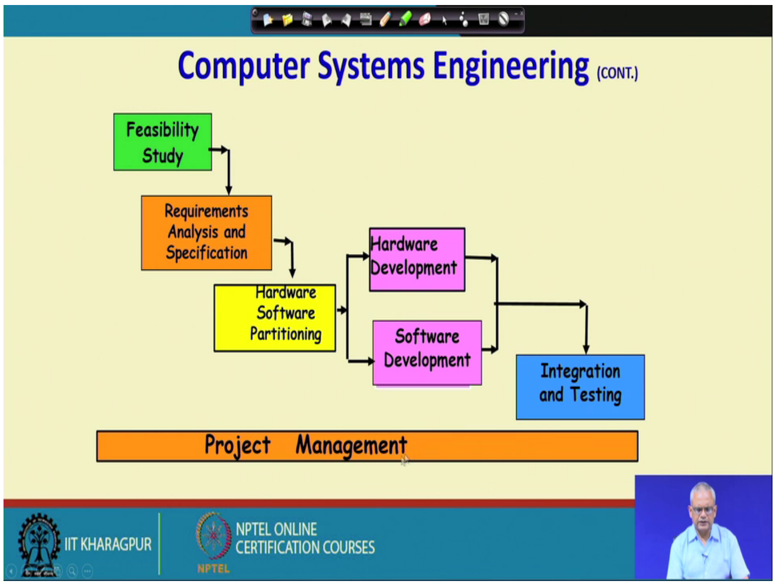
Computer Systems Engineering (CONT.)

- Typically, hardware and software are developed together:
  - –Hardware simulator is used during software development.
- Integration of hardware and software.
- Final system testing

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, hardware simulator is used during software development and then once the hardware and software development is complete they are integrated and the full system is tested.

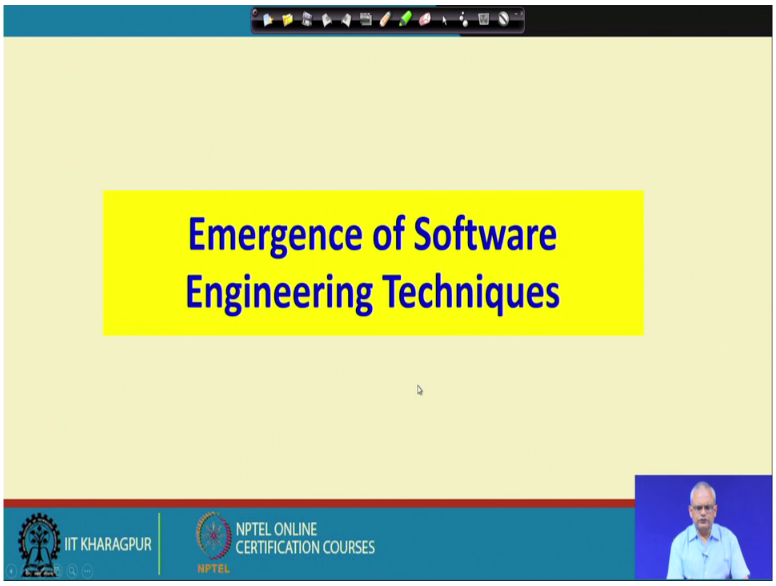
(Refer Slide Time: 14:56)



Computer Systems Engineering (CONT.)

So, this is a model or a schematic of the computer systems engineering, initially check whether we can do it we need to do it that is a feasibility study. The requirements analysis and specification, understand the requirements and document, then decide which one to be done by hardware which part to be done by software, then hardware development software development proceeding parallel, and after they complete integrated and the final system testing is done. And of course, through the entire project duration the project management activity is carried out to manage the various activities in the system engineering project.

(Refer Slide Time: 15:49)

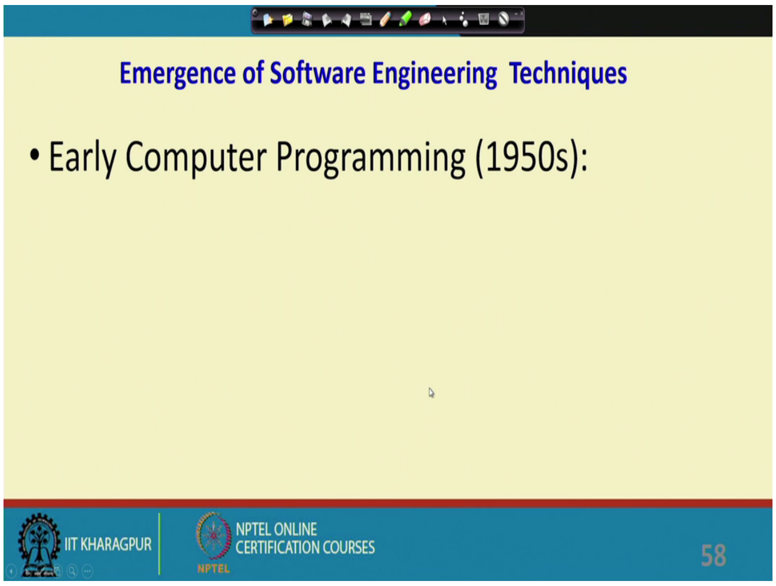


Emergence of Software Engineering Techniques

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Now, let us look at the different software engineering techniques; how they have developed. We are not concentrating on the software engineering techniques themselves which we will do a little later, but we will see that how they have evolved in time. This will give us an idea to appreciate that how starting with simple techniques, the software techniques have become more and more sophisticated.

(Refer Slide Time: 16:25)



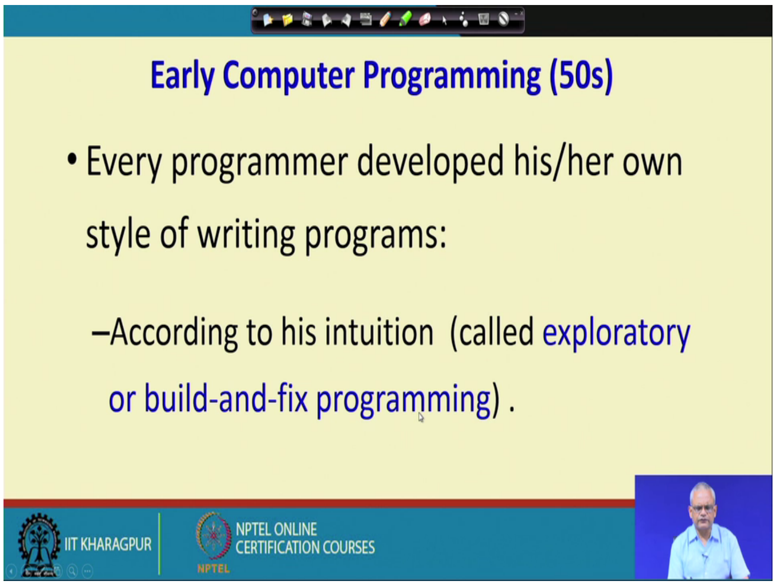
Emergence of Software Engineering Techniques

- Early Computer Programming (1950s):

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL
58

In the 1950s was the early computer programming, the programs were small.

(Refer Slide Time: 16:37)



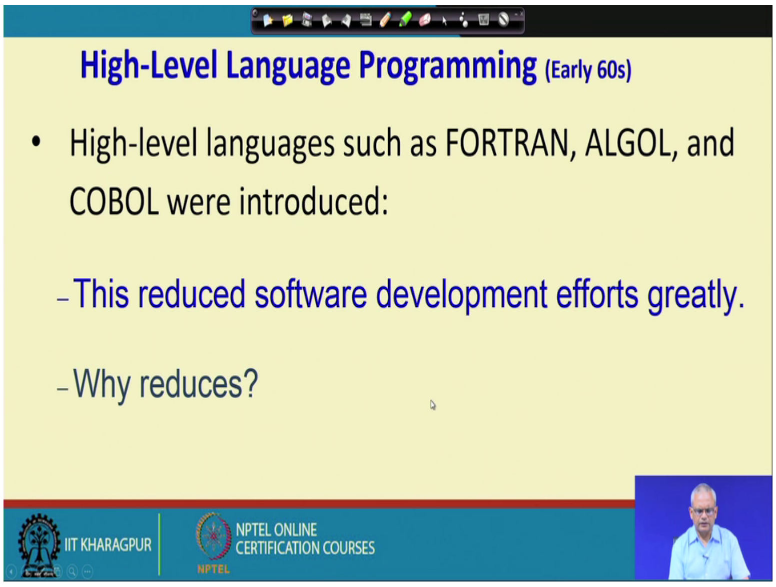
Early Computer Programming (50s)

- Every programmer developed his/her own style of writing programs:
  - According to his intuition (called exploratory or build-and-fix programming).

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

And they were used written using language like FORTRAN. Every programmer was having his own style that is basically the build and fix programming, every program was developed according to the intuition of the programmer what suits a programmer.

(Refer Slide Time: 17:00)



High-Level Language Programming (Early 60s)

- High-level languages such as FORTRAN, ALGOL, and COBOL were introduced:
  - This reduced software development efforts greatly.
  - Why reduces?

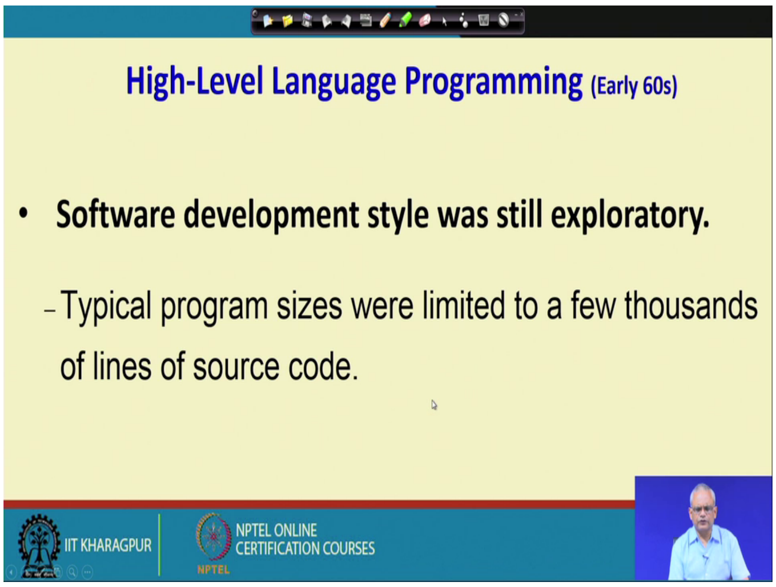
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Languages like FORTRAN, ALGOL, COBOL, were used later that was in 1960s. And the assembly language used was restricted. But then, the productivity improves significantly with high level language programming. Can you try to answer that; why is the productivity much higher using higher level programming languages as compared to assembly language programming.

The answer is that there are two parts: one is that one single line of high level program code, we will have to write many assembly code and the high level code is easily understood written machine language much more involved, because machine language is basically you have to understand the machine architecture, the registers; how that is transferred from one register to another and so on load to specific register. But then here, we will deal with a obstruction that would deal with only the variables.

So, high level language programming is much more simple. It abstracts out the details of hardware organization architecture.

(Refer Slide Time: 18:51)



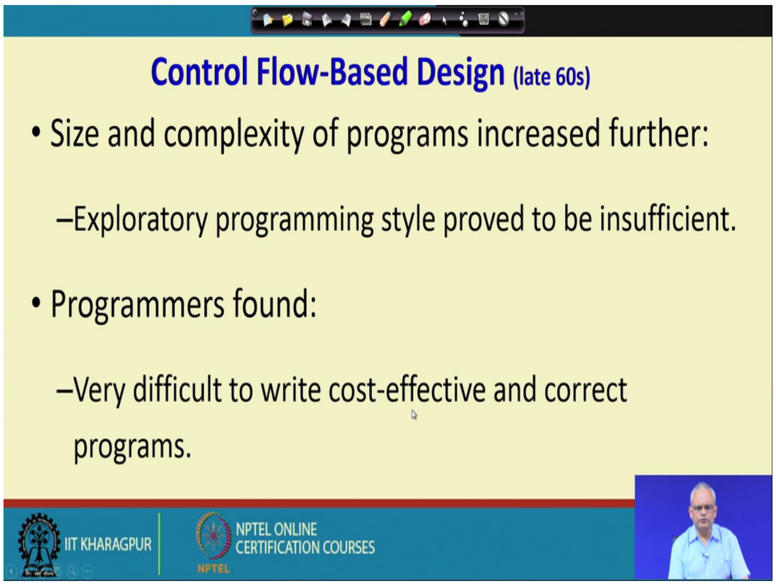
High-Level Language Programming (Early 60s)

- Software development style was still exploratory.
  - Typical program sizes were limited to a few thousands of lines of source code.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

But then, even though high level programming languages were used the development was largely exploratory, build and fix model, and typical program size was only about thousand line code.

(Refer Slide Time: 19:10)



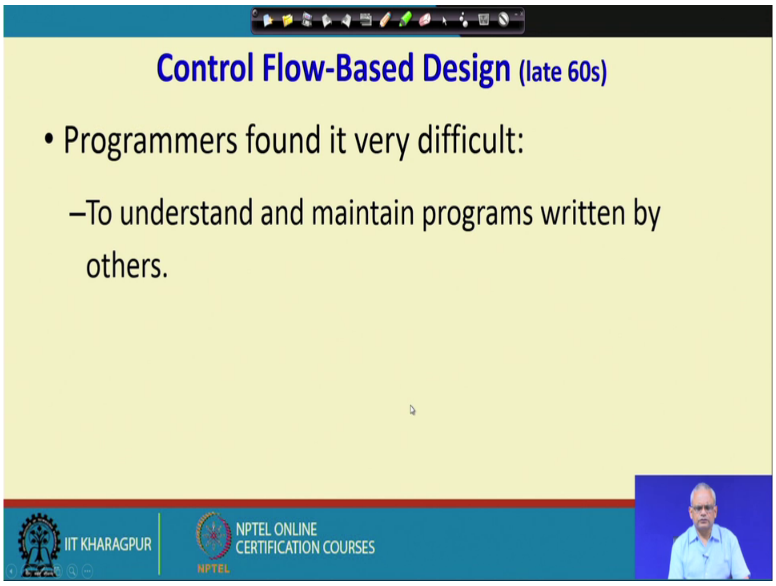
Control Flow-Based Design (late 60s)

- Size and complexity of programs increased further:
  - –Exploratory programming style proved to be insufficient.
- Programmers found:
  - –Very difficult to write cost-effective and correct programs.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

But the sizes of the program increased and the exploratory style became insufficient. It was taking long to develop code they are having too many bugs. But then there were good programmers.

(Refer Slide Time: 19:41)



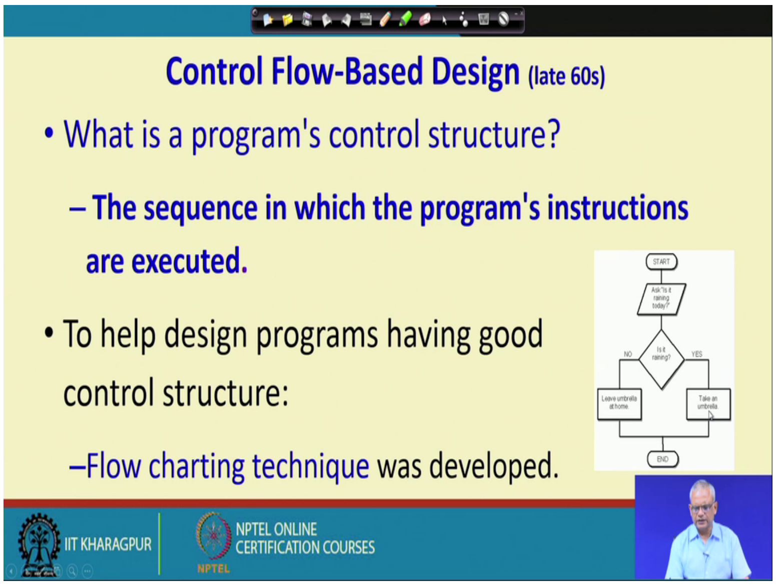
Control Flow-Based Design (late 60s)

- Programmers found it very difficult:
  - –To understand and maintain programs written by others.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Who advised that to be able to develop a good program you have to pay attention to the control flow.

(Refer Slide Time: 19:49)



Control Flow-Based Design (late 60s)

- What is a program's control structure?
  - **The sequence in which the program's instructions are executed.**
- To help design programs having good control structure:
  - Flow charting technique was developed.

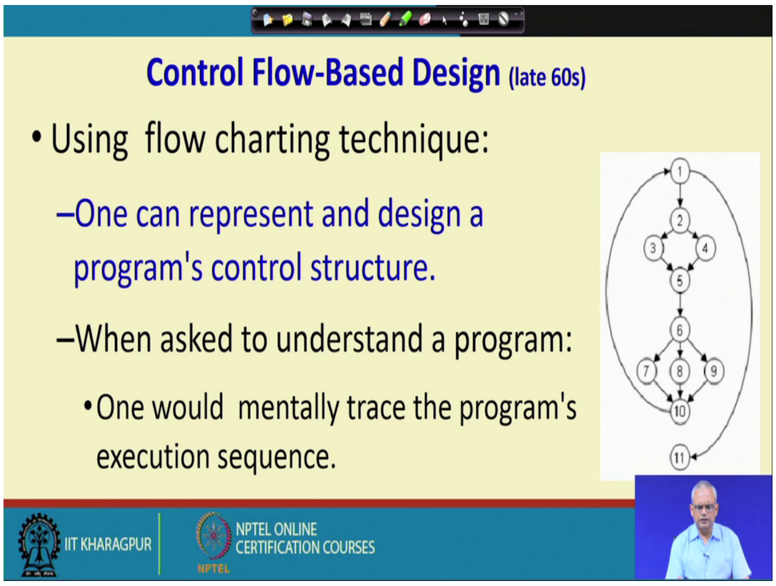
START
Ask "Is it raining today?"
Is it raining?
NO
YES
Leave umbrella at home
Take an umbrella
END
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

So, the good programmers who were experienced and who were writing programs well they advised that; please pay attention to the programs control structure. But what exactly is a control structure of a program? The control structure of a program is the sequence in which the programs instructions are executed. The sequence in which the programs instruction are executed may get altered when there are decisions, loops and so on.

So, we need to represent a control structure, we need to represent the sequence statements the decisions the loop structure and so on. And by looking at the control structure we can determine in what sequence the program will get executed. So maybe this statement and after that the decision statement will get executed depending on the outcome either this statement like get executed, if the outcome is false this statement will get executed and so on.

But then, the programmers had the advice from good programmers that pay attention to the control structure. You must have a good control structure for writing a good program. And the flow charting technique was developed, the flow charts were used to represent the programs control structure and best on these the code was written.

(Refer Slide Time: 21:36)

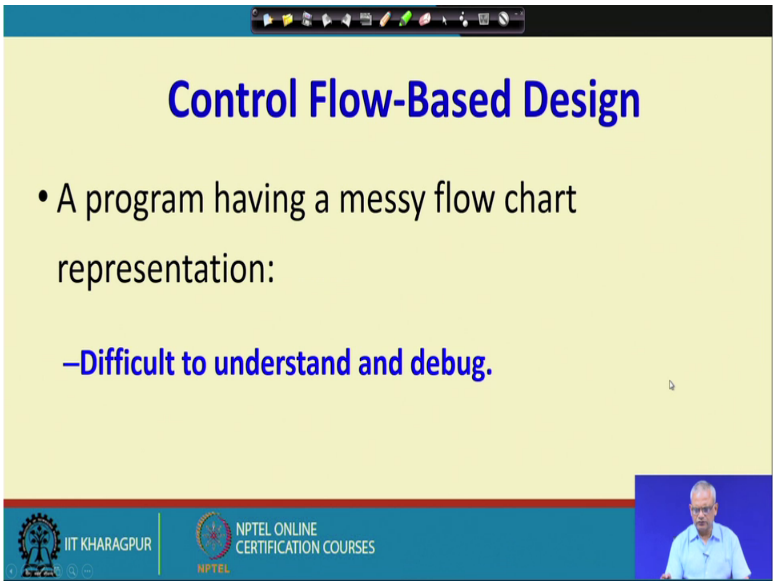


Control Flow-Based Design (late 60s)

- Using flow charting technique:
  - One can represent and design a program's control structure.
  - When asked to understand a program:
    - One would mentally trace the program's execution sequence.

But then, you must know that the control structure helps us to understand the program a good control structure. Because, if we want to understand a code having this control structure we will like to understand the different paths through it; that is starting from the first statement, which are the statement that are executed, where the output is generated and so on.

(Refer Slide Time: 22:07)

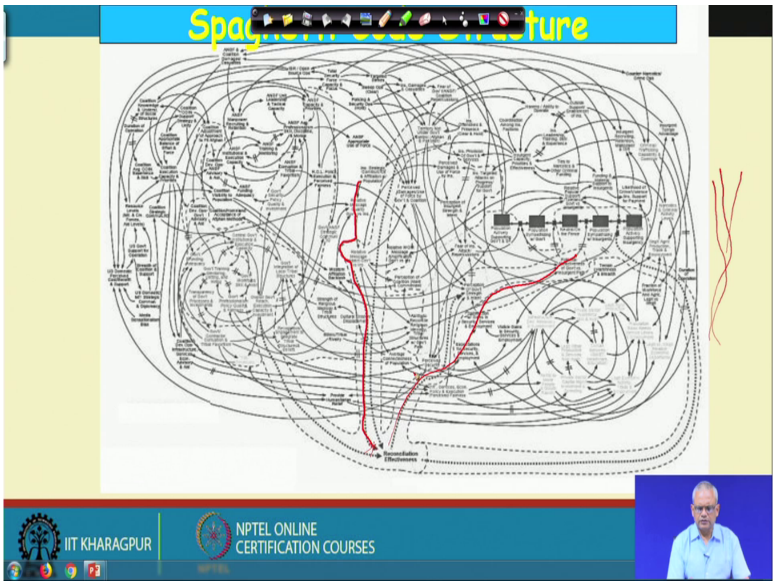


Control Flow-Based Design

- A program having a messy flow chart representation:
  - **Difficult to understand and debug.**

A program that has a bad control flow representation, that is a flow chart representation or control flow representation that will be very difficult to understand it will very difficult to debug and it will have lot of errors in it. But then why is it so, can you answer that if the control flow representation is bad let us say something like this.

(Refer Slide Time: 22:43)

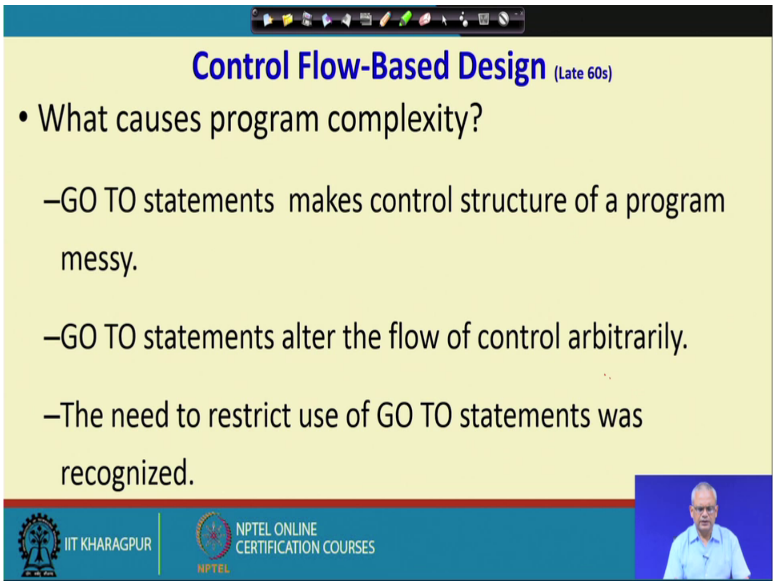


It is a very bad control flow representation. Why, becomes very difficult to understand the program having such a control structure,

The answer is that, if we want to understand this program then we will have to trace the different paths through it. Let us say we observe the output here and then we look at what is the path that this has taken, what are the statements that have got executed before it, and what might have got executed before that and so on, and from where it has got the input. And we need to do that for every path to be able to understand this program and imagine that there are thousands millions of paths here.

So, be able to trace the execution through all the paths in this is extremely tedious it will take long time. But if it had a good control structure and there are only few paths we can trace the paths; just 3-4 paths and then we are able to understand compared to a very complex programs control structure.

(Refer Slide Time: 24:21)



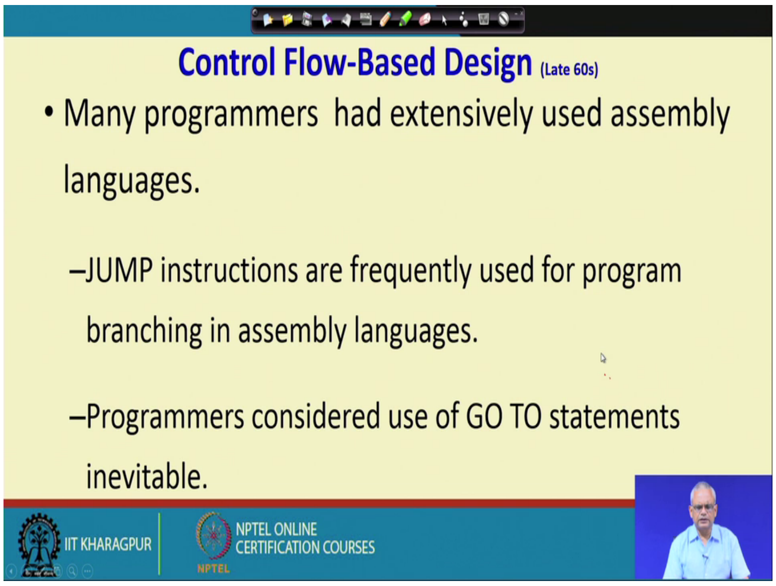
Control Flow-Based Design (Late 60s)

- What causes program complexity?
  - GO TO statements makes control structure of a program messy.
  - GO TO statements alter the flow of control arbitrarily.
  - The need to restrict use of GO TO statements was recognized.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And that gave rise to the control flow base design. The control flow base design said that please have a good control structure for the program. But how does one have a good control structure for the program? They said that, see it is GO TO statements they are the culprits actually, they make the control structure bad. Of course, the modern programming languages they hardly support GO TO statements, but the earlier programming languages like FORTRAN etcetera they had this GO TO statements and these were used heavily.

(Refer Slide Time: 25:09)



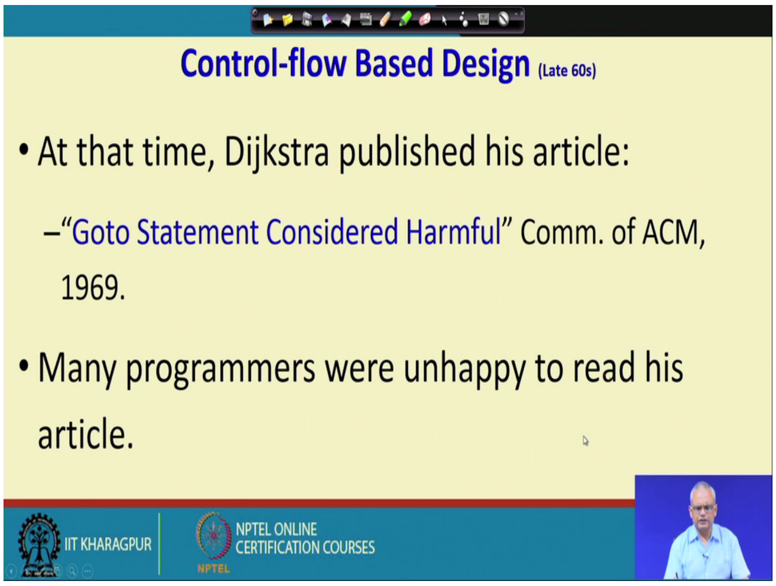
Control Flow-Based Design (Late 60s)

- Many programmers had extensively used assembly languages.
  - JUMP instructions are frequently used for program branching in assembly languages.
  - Programmers considered use of GO TO statements inevitable.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But look at this in the historical prospective. The assembly programmers, they said that see without jump instruction you cannot write a program basically. So, GO TO statements are inevitable without using GO TO statement you cannot write sophisticated programs.

(Refer Slide Time: 25:38)

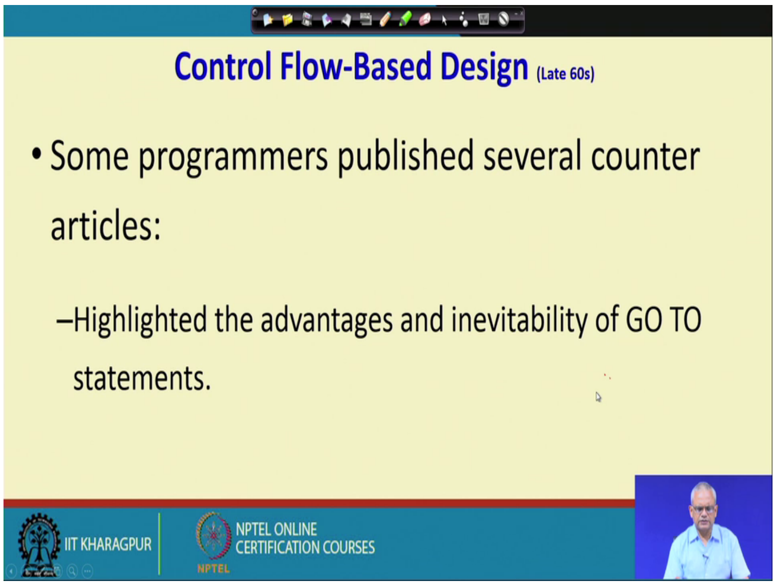


Control-flow Based Design (Late 60s)

- At that time, Dijkstra published his article:
  - "Goto Statement Considered Harmful" Comm. of ACM, 1969.
- Many programmers were unhappy to read his article.

At that juncture Dijkstra he published an article in the communication of ACM 1969 a landmark article called as 'Goto Statement Considered Harmful'. And he wrote about the problems that a GO TO statement creates and obviously, many programmers who were basically having assembly programming background they were very unhappy.

(Refer Slide Time: 26:04)



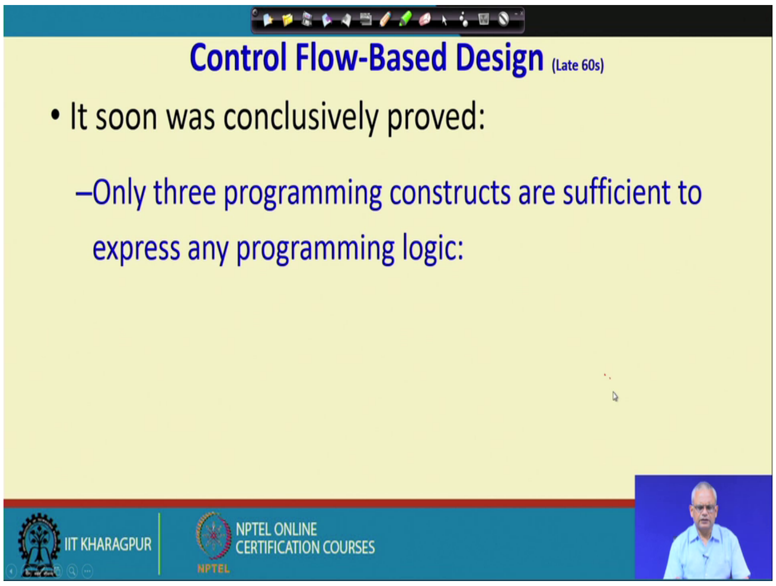
Control Flow-Based Design (Late 60s)

- Some programmers published several counter articles:
  - Highlighted the advantages and inevitability of GO TO statements.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

They wrote counter articles they said that see without GO TO statement you cannot really write a large program.

(Refer Slide Time: 26:14)



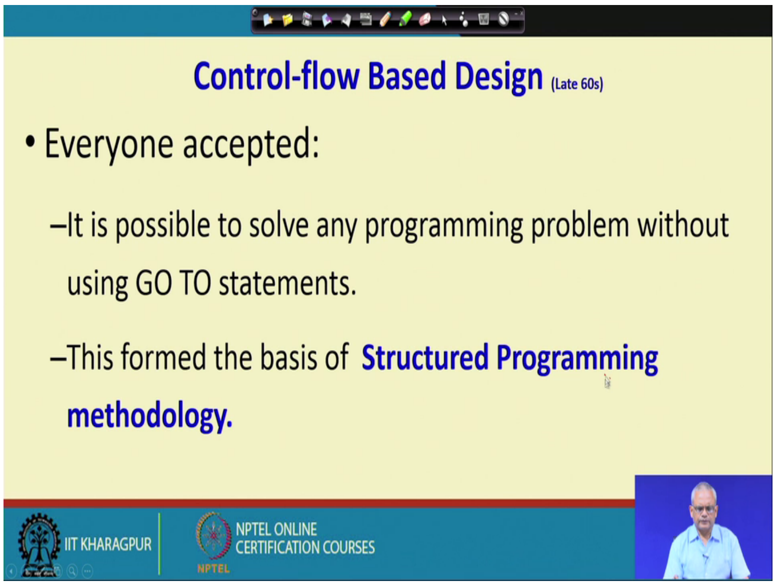
Control Flow-Based Design (Late 60s)

- It soon was conclusively proved:
  - Only three programming constructs are sufficient to express any programming logic:

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

But then, it was proved that to be able to solve any programming problem that includes the large complex problems you need only three types of constructs: the sequence, selection, and the alteration constructs.

(Refer Slide Time: 26:36)



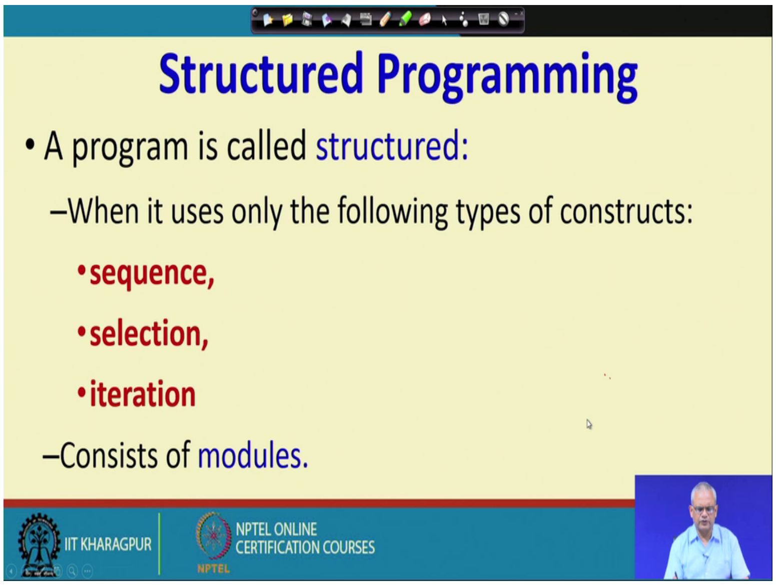
Control-flow Based Design (Late 60s)

- Everyone accepted:
  - It is possible to solve any programming problem without using GO TO statements.
  - This formed the basis of **Structured Programming methodology.**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And slowly everybody accepted that it is possible to write large programs without using GO TO statements and this earn the basis of the structured programming methodology.

(Refer Slide Time: 26:48)



Structured Programming

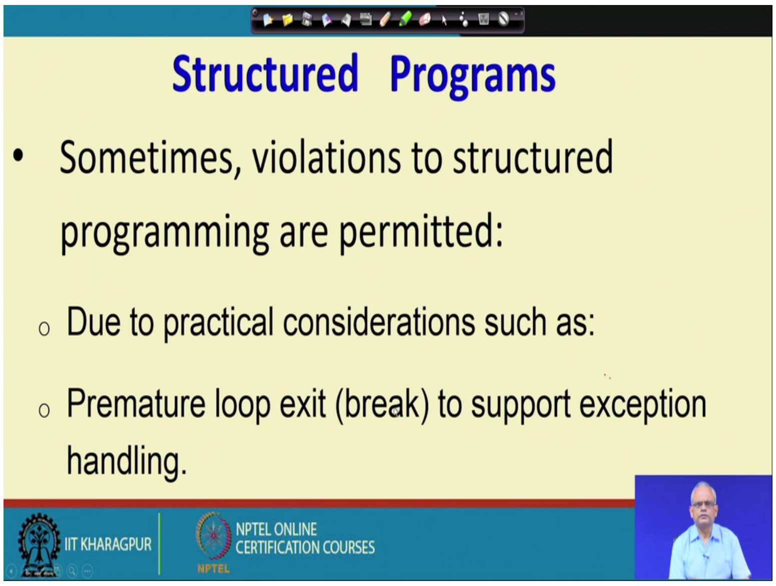
- A program is called structured:
  - When it uses only the following types of constructs:
    - **sequence,**
    - **selection,**
    - **iteration**
  - Consists of modules.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us understand what exactly structured programming. A program is called structure when it uses only three types of constructs: the sequence, selection; sequence is one statement after the other statement like one arithmetic statement followed by another arithmetic statement. Selection, like if then else, switch and so on. Iteration, like for loop while loop, do loop and so on. But just see here there is no place for a go to here.

So, a program is structured when it uses only sequence selection and iteration does not use GO TO statements, and also it is a modular program. This is the basic features of a structured program is that it uses only sequence selection and iteration type of statements and also it consists of modulus.

(Refer Slide Time: 27:58)



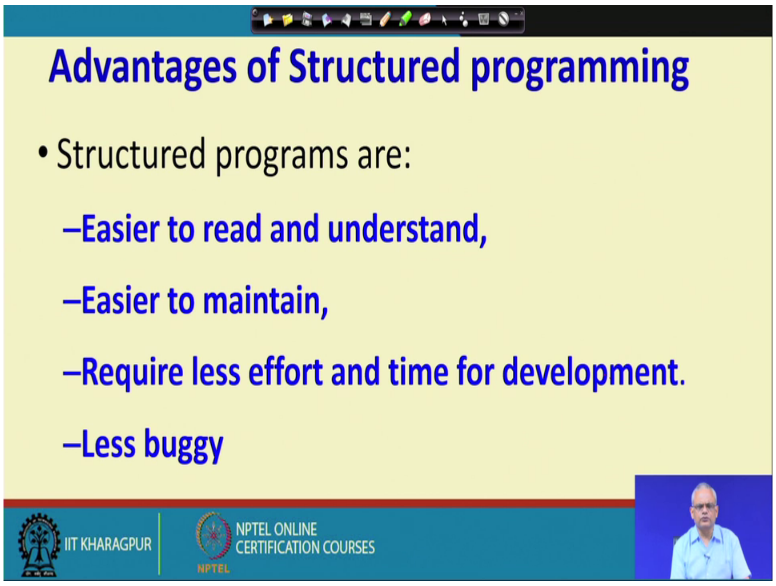
Structured Programs

- Sometimes, violations to structured programming are permitted:
  - Due to practical considerations such as:
  - Premature loop exit (break) to support exception handling.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Of course, sometimes may have to use GO TO statement in the modern languages do not really use GO TO statement, but then they have statements like break for premature loop exit, exceptions and so on. So, these are basically not structured constructs, but then occasionally it is an allowed due to practical considerations.

(Refer Slide Time: 28:32)



Advantages of Structured programming

- Structured programs are:
  - –Easier to read and understand,
  - –Easier to maintain,
  - –Require less effort and time for development.
  - –Less buggy

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

But let us understand that what is the advantage of structured program. There are many advantages: one is that as we were discussing a structured program has a good control structure, because it does not use GO TO statements. It has few paths, a control structure is simple and therefore, it is easy to read and understand.

And since it is easy to read and understand it is also easy to maintain, because for maintenance we need to first read and understand the program and then decide what to change. It also requires less effort and time for development. Why is it; so that we need less effort and time for development. The answer is that: if you are developing in on structured way even a small mistake that you have committed will take long time for you to correct, but a structured program can easily correct understand under the grasp.

And therefore, it requires less effort and time for development. And also it is less buggy because you understand it well and therefore, chances of bugs being there is less, whereas non structured program you do not really understand how the program works, what are all the paths that exist, there maybe thousands of paths. And the bug may exist in one of the paths which you never thought existed.

So, the structured programming principle has been accepted. It is still very popular technique in almost every program that we write we expect it to be structured program. And of course, as you are saying that the modern programming languages facilitate writing structured programs, because they do not have constructs like go to and so on.

So, we will stop at this point of time and continue in the next lecture.

Thank you.