

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

Lecture – 06
Life Cycle Model

Welcome to this lecture, we will continue what we were discussing last time.

(Refer Slide Time: 00:25)

- Use of Life Cycle Models
- Software is developed through several well-defined stages:
 - Requirements analysis and specification,
 - Design,
 - Coding,
 - Testing, etc.

Differences between the exploratory style and modern software development practices

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 94

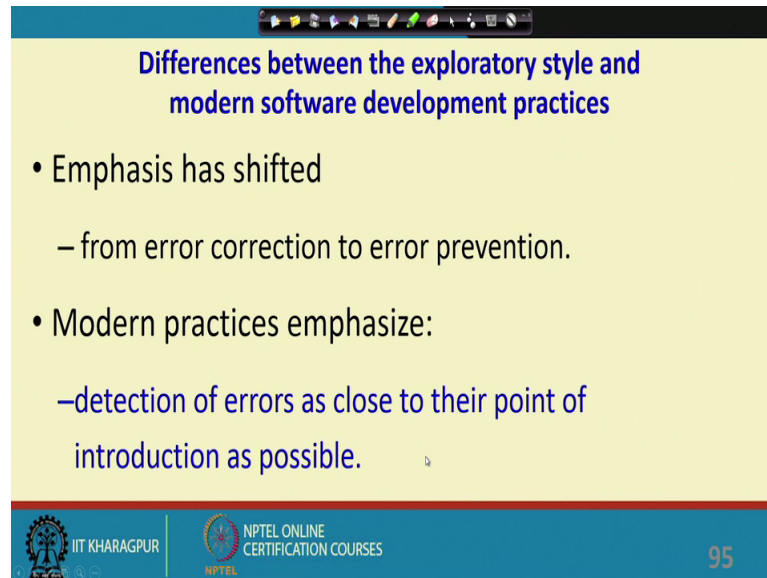
Last time we are discussing that the software engineering practices have really evolved over the years have become much more sophisticated. Now, let us look at some major differences, in the way software is being written now one is that the software engineering principles. Themselves have evolved and the other is the software itself has evolved.

The software writing problem is not the same as the problem that now we are solving. The, problems have become much more complex, but at the same time these are only incrementally different from the problem that has been solved.

Let us let us look at these issues now; the differences between exploratory style and the modern software development practices. Compared to 1950s or 60s where the exploratory style was used, now the life cycle models are used every program is developed according to an adopted life cycle model, where there are a set of stages and the development occurs through those stages.

The stages typically the requirements analysis specification design, coding, testing, and so, on.

(Refer Slide Time: 01:58)



Differences between the exploratory style and modern software development practices

- Emphasis has shifted
 - from error correction to error prevention.
- Modern practices emphasize:
 - detection of errors as close to their point of introduction as possible.

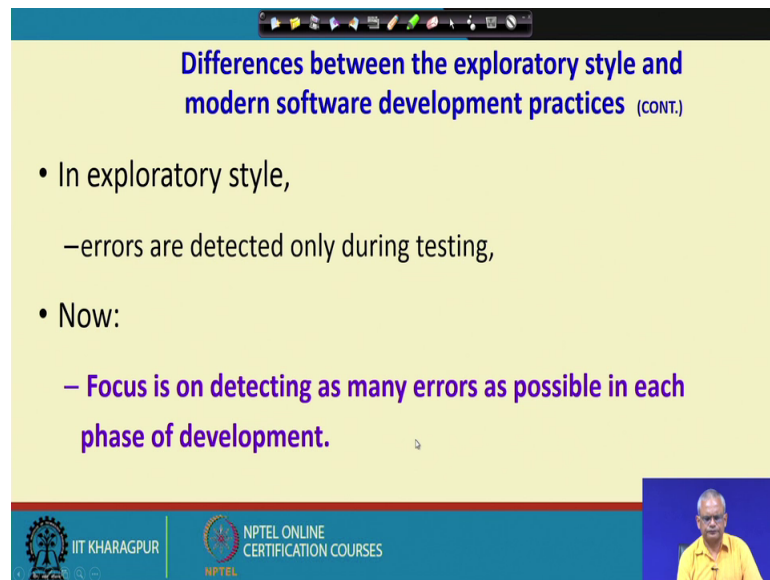
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 95

One major difference between the way program was written 1950s and 60s to now is that the emphasis has shifted from error correction to error prevention. The earlier technique the exploratory style was to first write the program complete somehow and then start correcting the errors, there will be hundreds of errors keep on eliminating one by one until all errors are eliminated.

But, now the emphasis is error prevention as the program is written we track the errors and remove it from there itself. So, that during testing we have very less number of errors to correct, or in other words as soon as we have a mistake committed by the programmer in program development, we try to detect it as close to the time of commitment of the error as possible the main advantage of this is that it is very cost effective.

If, you correct the error after testing then you will have to first find out where exactly the error has occurred make changes to the code again test and so on. Here, before testing we just identify the place where the error is there we do not have to look at we just look through or review the code and so on, review the specification review the design identify the error corrected there itself and do not wait for error to be discussed detected and determined during the testing phase.

(Refer Slide Time: 04:05)



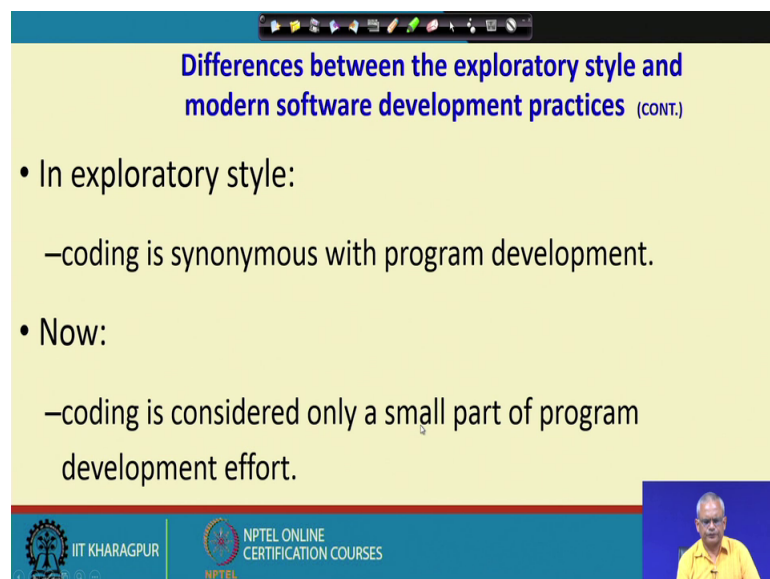
Differences between the exploratory style and modern software development practices (CONT.)

- In exploratory style,
 - errors are detected only during testing,
- Now:
 - **Focus is on detecting as many errors as possible in each phase of development.**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The software is being developed through number of phases and in each phase, wherever a program commits a mistake. It is detected and corrected in that same phase as far as possible.

(Refer Slide Time: 04:26)



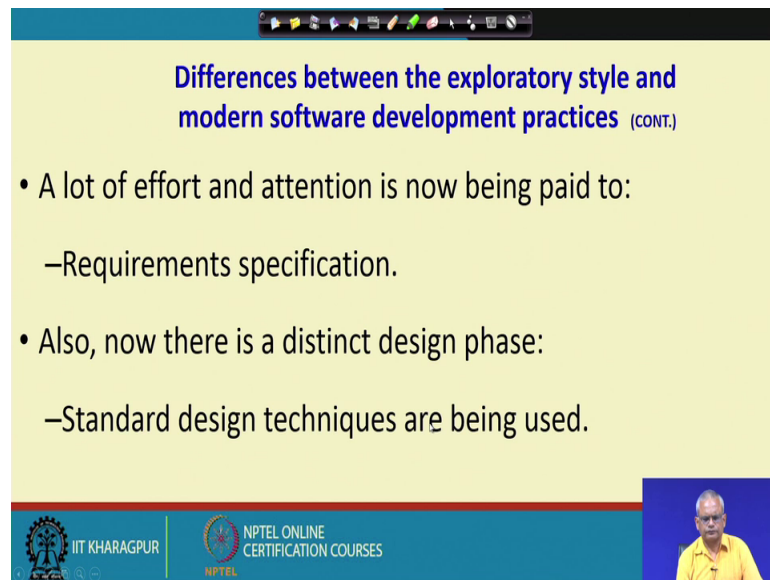
Differences between the exploratory style and modern software development practices (CONT.)

- In exploratory style:
 - coding is synonymous with program development.
- Now:
 - coding is considered only a small part of program development effort.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The exploratory style program development was basically coding, start writing the code as soon as the problem is given, but right now in the modern development practice coding is actually a small part of the development activities. The major activities are specification design then coding and then testing.

(Refer Slide Time: 05:01)



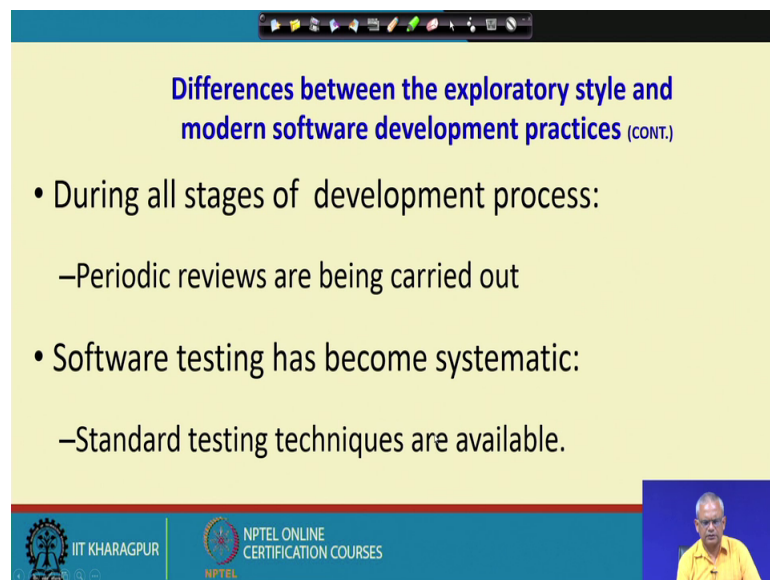
Differences between the exploratory style and modern software development practices (CONT.)

- A lot of effort and attention is now being paid to:
 - Requirements specification.
- Also, now there is a distinct design phase:
 - Standard design techniques are being used.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Of course, lot of attention now is given to requirement specification then the design has to be done that, standard design techniques that are available.

(Refer Slide Time: 05:16)



Differences between the exploratory style and modern software development practices (CONT.)

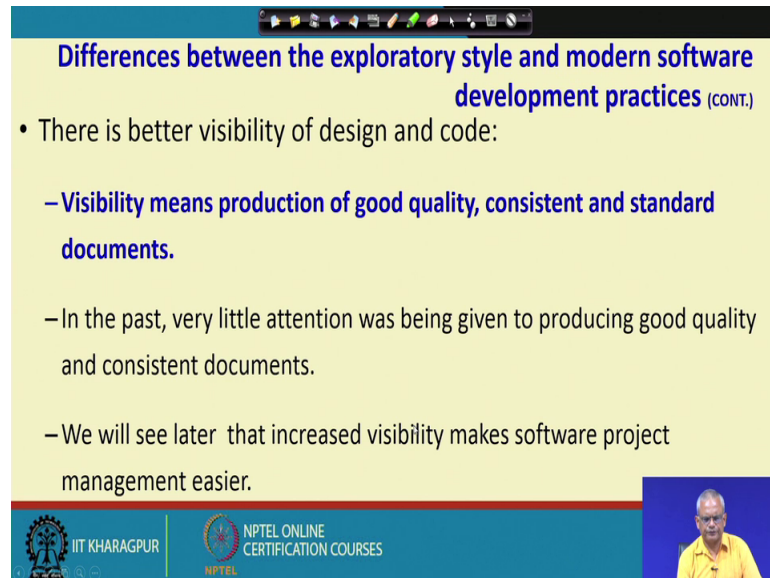
- During all stages of development process:
 - Periodic reviews are being carried out
- Software testing has become systematic:
 - Standard testing techniques are available.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And, at the end of every phase reviews are conducted the main idea behind review is to detect errors as soon as these are committed by the programmer, do not wait for the error to be detected during testing. Reviews help us to detect the errors much earlier, and this is an efficient way to detect errors and therefore, reduces cost of development and the time of development.

The testing techniques have become really sophisticated we will look at them later in this course, and there are many standard testing techniques and tools are available.

(Refer Slide Time: 06:03)



Differences between the exploratory style and modern software development practices (CONT.)

- There is better visibility of design and code:
 - **Visibility means production of good quality, consistent and standard documents.**
 - In the past, very little attention was being given to producing good quality and consistent documents.
 - We will see later that increased visibility makes software project management easier.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Earlier the programmer just wrote the code and then never know when it will get completed you will just have to ask the programmer, how far he has done in the coding? But, even if he say that I am nearly complete completing the code, but then he might have too many bugs and he may take years to correct them.

But, right now there is a visibility of the development somebody can see that the design is done, code is done or let us say the requirements are done. So, these documents improve the visibility, earlier there are no visibility you just have to ask the programmer that how far he has done? Right now you can see the documents and determine with what stages of development it is.

The increased visibility makes software project management much easier, earlier the project manager has to just ask the programmer how he is doing, how long he is? And, naturally the projects were going hey we are one year project taking 5 years was not uncommon, but now due to the increased visibility the project manager can very accurately determine when the project at what stage it is and how long it will take to complete the work?.

(Refer Slide Time: 07:49)

Differences between the exploratory style and modern software development practices (CONT.)

- Because of good documentation:
 - fault diagnosis and maintenance are smoother now.
- Several metrics are being used:
 - help in software project management, quality assurance, etc.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Because, good documents now are produced later maintenance becomes easier several metrics are being used, which are used to determine the quality of the software, software project management and so on.

(Refer Slide Time: 08:09)

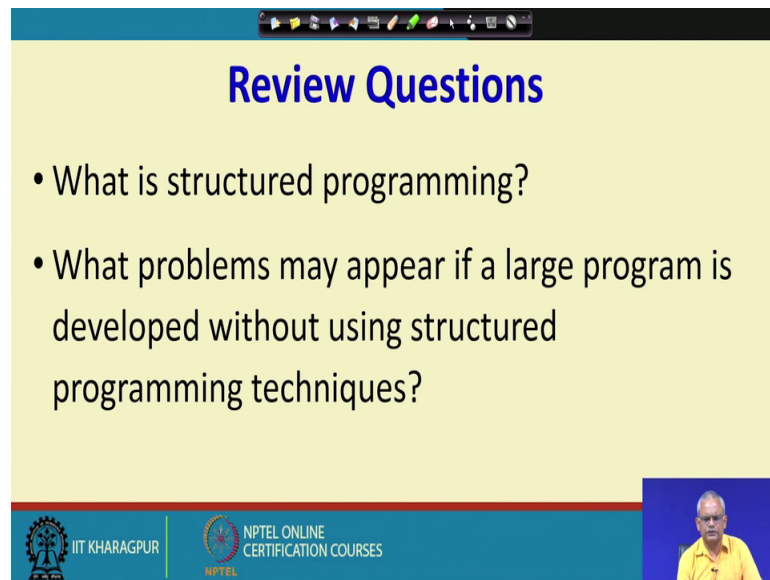
Differences between the exploratory style and modern software development practices (CONT.)

- Projects are being properly planned:
 - estimation,
 - scheduling,
 - monitoring mechanisms.
- Use of CASE tools.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

There are many project management techniques that are being used like estimation scheduling monitoring mechanisms and also case tools are being used extensively.

(Refer Slide Time: 08:24)



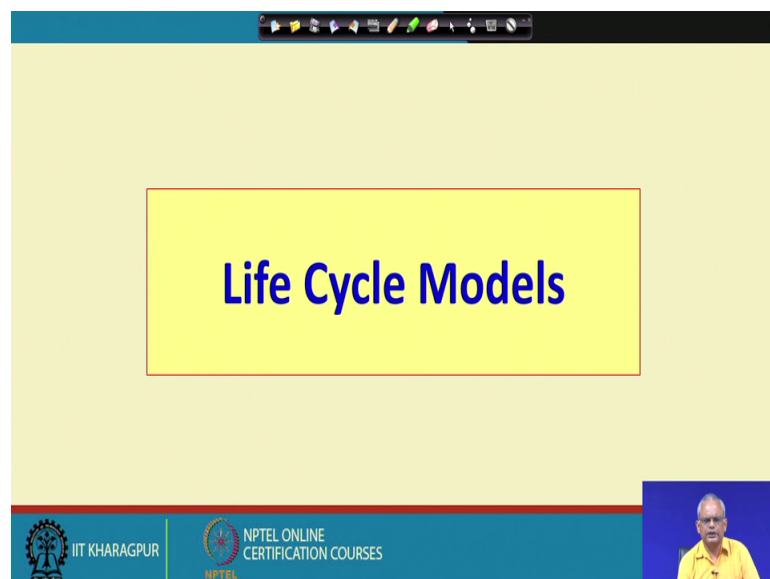
Review Questions

- What is structured programming?
- What problems may appear if a large program is developed without using structured programming techniques?

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us look at a couple of review questions before we proceed further; one is what do we mean by structured programming. As, we had discussed structured programming is writing programs that do not use go tos, but only use sequence selection and iteration type of constructs and their modular. What problems may appear if a large program is developed without using structured programming techniques? Ok. The program will take long time longer than what it should be it would cost much more, it will have lot of bugs difficult to maintain difficult to understand the code and so on.

(Refer Slide Time: 09:29)

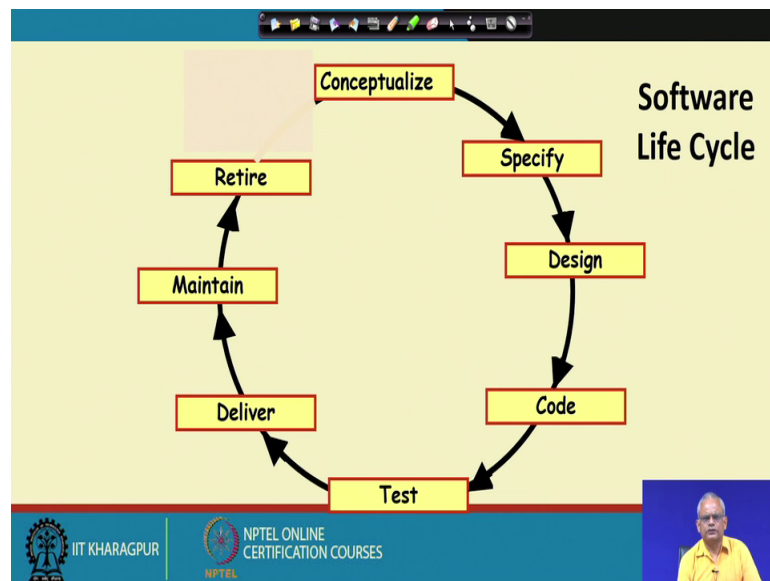


Life Cycle Models

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, we have completed the introductory topics that we wanted to discuss before we look at the specification design testing and so on. We will just spend a couple of lectures on the lifecycle models, because these are one of the basic principles that we must know and before a project starts we must know what are the life cycle models that are available? Which one will suit this project and we should be able to use this project or if a project is using certain lifecycle model, we must be able to know that why this is being used and what is involved in the lifecycle model that is being used.

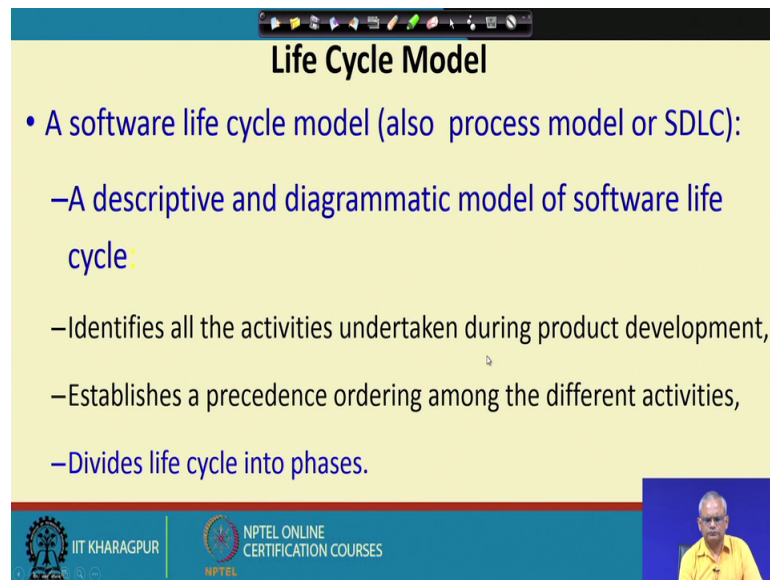
(Refer Slide Time: 10:19)



At a very conceptual model we have some notion of a life cycle. A life cycle is the set of the stages through which something evolves. For example, a human being life cycle involves from child, adult, old ages, and retired and so on. Similar is the software life cycle conceptually you can think that somebody thought of the software.

So, that is conceptualization then it finds out what exactly is needed specification, then design the software code based on the design test it, install it, and deliver may then later problems are maintained and then after some time the software may not be needed anymore maybe, because the business has changed maybe different software have become available which are much better, or maybe the hardware changed. So, no more used for whatever reason slowly after many years it is no more used and it is retired.

(Refer Slide Time: 11:48)



Life Cycle Model

- A software life cycle model (also process model or SDLC):
 - A descriptive and diagrammatic model of software life cycle:
 - Identifies all the activities undertaken during product development,
 - Establishes a precedence ordering among the different activities,
 - Divides life cycle into phases.

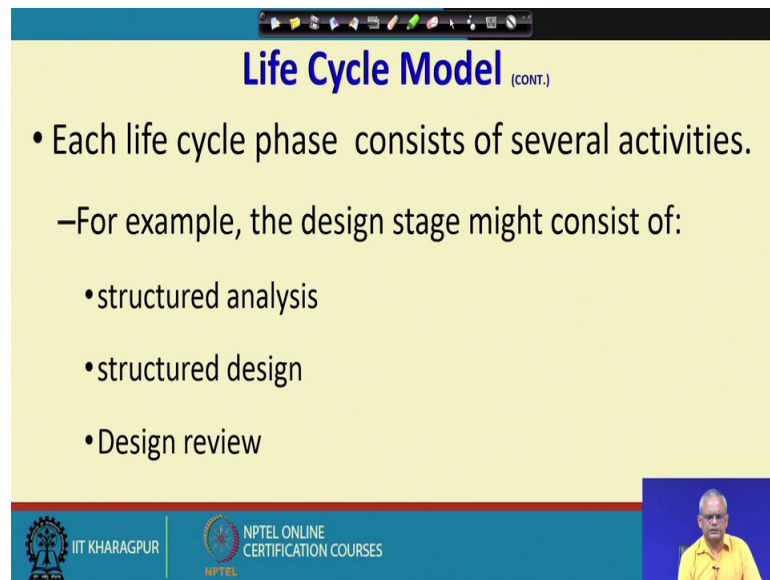
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Based on this intuitive idea of a lifecycle model let us look at the what exactly is a software lifecycle model which is also called as a process model, the development process model or the software development life cycle SDLC. Normally we represent software lifecycle model or the process model in the form of a diagrammatic representation just like, we had represented in the earlier slide, but just a diagrammatic representation is not enough even though it helps us to get a quick glance of what happens? What are the stages? How does the software evolve starting from concepts conception to the final code, but it lacks details.

And therefore, with every lifecycle model will not only have a diagrammatic model, but also a descriptive description of what exactly is involved. Every lifecycle model should identify what are the activities through, which software is developed. May be the requirements gathering, requirement analysis, specification, high level design, detail design and so on; Every software is developed through different types of activities and lifecycle model represents all these activities.

And, not only it represents the activities, but also it establishes a precedence ordering that which activity should be done after which one and also divides all these activities into phases, that is which activities should take place in which phase of development.

(Refer Slide Time: 13:59)



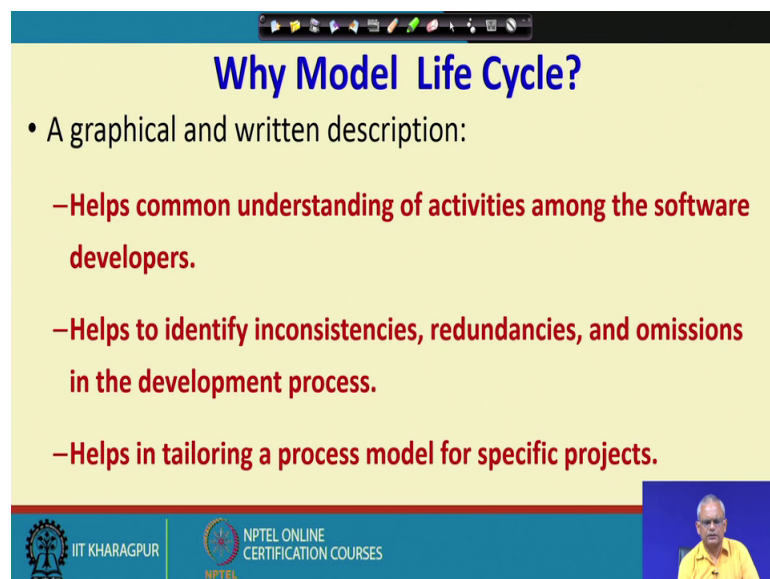
Life Cycle Model (CONT.)

- Each life cycle phase consists of several activities.
 - For example, the design stage might consist of:
 - structured analysis
 - structured design
 - Design review

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

In other words every phase of development consists of many activities. For example, the design stage activity may be structured analysis, structure design, design review and so on.

(Refer Slide Time: 14:18)



Why Model Life Cycle?

- A graphical and written description:
 - Helps common understanding of activities among the software developers.
 - Helps to identify inconsistencies, redundancies, and omissions in the development process.
 - Helps in tailoring a process model for specific projects.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, let us answer this basic question, that why model lifecycle? If as long as we understand that what are the activities to be done and so on, why do we need a lifecycle model?

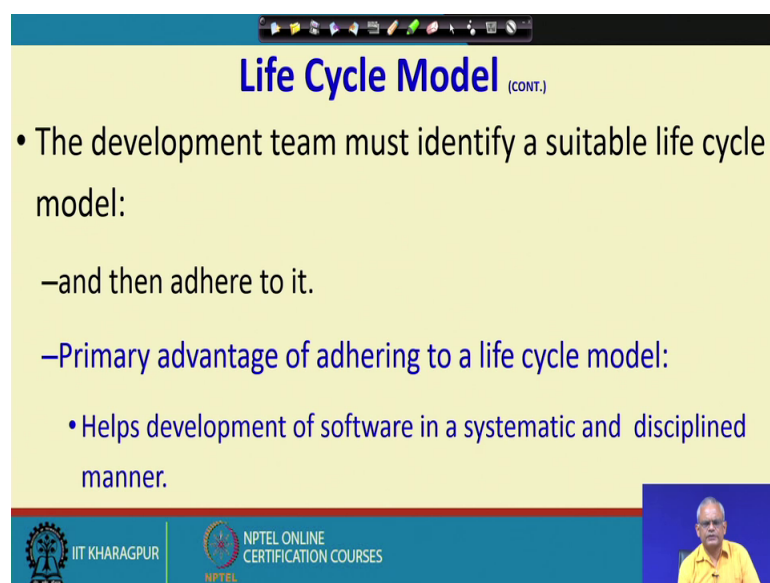
Now, every development organization they have their own life cycle model, which is a written description graphical and also written available in the form of books to be referred by existing programmers new programmers and so on. The main advantage of having such a graphical and written description, that is having a documented life cycle model is that all developers will have a common understanding of activities that will be done, the sequence of activities and so on.

Therefore, the development becomes much more disciplined, it helps to identify inconsistencies in the activities, redundancies some activity not required omissions some activities required, but not there let us say review is not mentioned, but that is really required.

So, we have to mention review in the lifecycle model. And, not only that as long as we have a written lifecycle model for a specific project, we can look at the lifecycle model and tailor it little bit that these are the activities which are not required or these are some new activities that will be required for this specific project.

No wonder that most of the standards now quality standard they require every organization, which would qualify for that quality standard would to have a written lifecycle model or the process model. Quality standards like ISO ACI etcetera. They all require a documented lifecycle model.

(Refer Slide Time: 16:50)



The slide is titled "Life Cycle Model (CONT.)" and contains the following text:

- The development team must identify a suitable life cycle model:
 - and then adhere to it.
 - Primary advantage of adhering to a life cycle model:
 - Helps development of software in a systematic and disciplined manner.

The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom, and a small video inset of a speaker in the bottom right corner.

Once, written life cycle model is available an organization they have identified a suitable lifecycle model for the type of the projects that do, the programmers the developers are asked to understand it fully and they adhere they use it rigorously.

(Refer Slide Time: 17:11)

Life Cycle Model (CONT.)

- When a program is developed by a single programmer ---
 - The problem is within the grasp of an individual.
 - He has the freedom to decide his exact steps and still succeed --- called Exploratory model--- One can use it in many ways
 - Code → Test → Design
 - Code → Design → Test → Change Code →
 - Specify → code → Design → Test → etc.

The diagram illustrates a flowchart for the Exploratory model. It starts with a box labeled 'Initial Coding'. An arrow points to a box labeled 'Test'. From 'Test', an arrow points to a box labeled 'Fix'. From 'Fix', an arrow points back to 'Test'. A red arrow points from 'Test' to the right, labeled 'Do Until Done'.

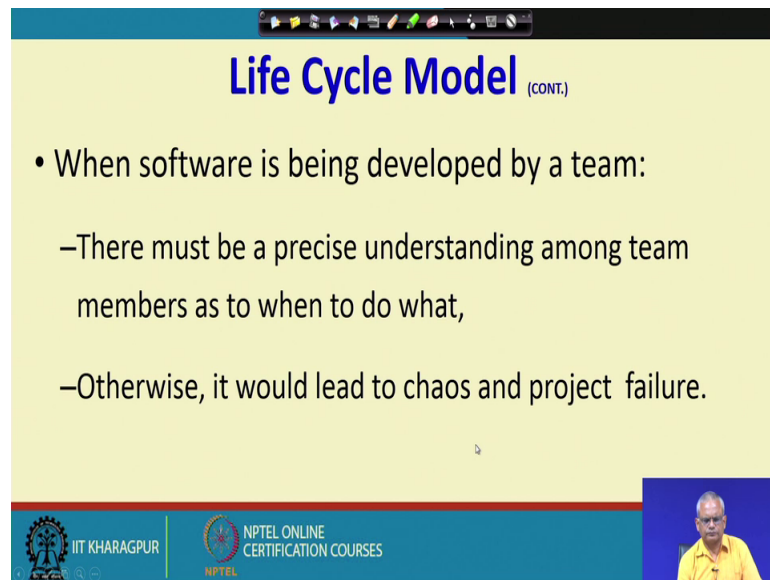
The main advantage is that the lifecycle models help him help to solve a big problem, that as long as the prog problem is small it is done by one programmer or one developer. And, he does by himself whatever he needs to do? But, when there is a team of developers working to solve a large problem, then the lifecycle model has to be used because every developer has to have a precise understanding of what exactly needs to be done? What the other developers are doing? What in they are expecting from the development?

The, earlier life cycle the earlier informal development technique that was used was the build and fix tile. This can be used only when the problem is very small and is within the grasp of an individual, and he has the liberty to do as interpret this style, this is a very informal style interpret maybe he just writes the requirements first very preliminary ways does coding test fix and so on.

But, another designer may interpret this and say that first needs to write the code, then design then test then change code another designer might say that need to specify code design test.

So, the developer said flexibility in using whatever style that suited to them, but when there is a team development this kind of informal style of development would create real problem and the project will fail.

(Refer Slide Time: 19:19)

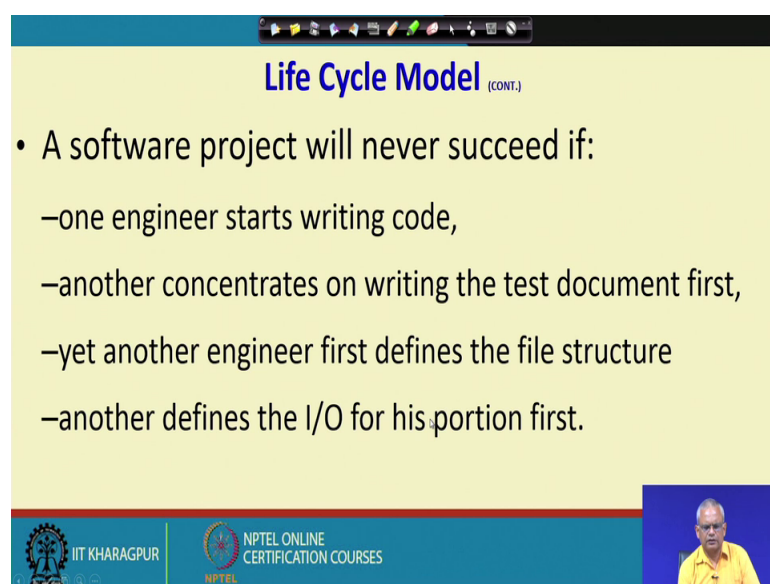


The slide is titled "Life Cycle Model (CONT.)" in blue text. It contains a bulleted list with two main points. The first point is "When software is being developed by a team:", followed by two sub-points: "–There must be a precise understanding among team members as to when to do what," and "–Otherwise, it would lead to chaos and project failure." The slide footer includes the IIT Kharagpur logo, the text "IIT KHARAGPUR", the NPTEL logo, and "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a speaker is visible in the bottom right corner.

- When software is being developed by a team:
 - There must be a precise understanding among team members as to when to do what,
 - Otherwise, it would lead to chaos and project failure.

Team development there must be precise understanding of the team members, that who will do what, when, what is expected from a team member, who will supply that and so on. Without such a disciplined life cycle model there can be chaos and project failure.

(Refer Slide Time: 19:46)



The slide is titled "Life Cycle Model (CONT.)" in blue text. It contains a bulleted list with one main point: "A software project will never succeed if:", followed by four sub-points: "–one engineer starts writing code,", "–another concentrates on writing the test document first,", "–yet another engineer first defines the file structure", and "–another defines the I/O for his portion first." The slide footer includes the IIT Kharagpur logo, the text "IIT KHARAGPUR", the NPTEL logo, and "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a speaker is visible in the bottom right corner.

- A software project will never succeed if:
 - one engineer starts writing code,
 - another concentrates on writing the test document first,
 - yet another engineer first defines the file structure
 - another defines the I/O for his portion first.

For example, let us say life cycle is not followed one programmer writes code for some part, another is there he wants to do the test document design first, another one is doing the file structure, another one is writing specification and so on.

Then, this type of project is destined to failure, because they are doing very different activities and when the specification is done by one programmer the other programmer has already written the code and so on.

(Refer Slide Time: 20:31)

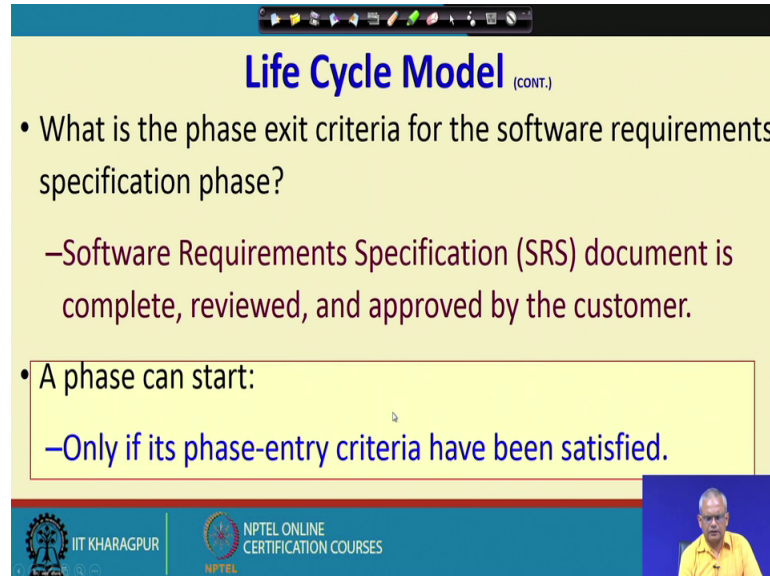
The slide is titled "Phase Entry and Exit Criteria" in blue text. It features a diagram of a life cycle model consisting of three yellow squares connected by blue arrows, with an arrow pointing to the right from the last square. Below the diagram are three bullet points: "• A life cycle model:", "–defines entry and exit criteria for every phase.", and "–A phase is considered to be complete:", followed by a sub-bullet point "• only when all its exit criteria are satisfied." The slide footer includes the IIT KHARAGPUR logo, the NPTEL ONLINE CERTIFICATION COURSES logo, and a small video inset of a man in a yellow shirt.

Every lifecycle model as you are saying consists of a set of phases, but then before a phase starts certain conditions must be satisfied, before the work can start. And, similarly a phase let us say requirement phase to complete, you must know what are the conditions, that must be satisfied for the stage to complete.

For example, the requirement document is completely written and has been reviewed by the team members internally and also by the customer. Without a phase entry and exit criterion clearly mentioned in the life cycle model, it will create a lot of confusion. For example, if we do not say that requirements document is complete only after the document has been written completely and has been reviewed, different teams may interpret it differently and some might just write it and then proceed to the next phase, another team may say that we need to review and so on.

So, for on the ambiguous interpretation of the life cycle model every phase must have entry and exit criteria, that when does the phase can start and when can it end.

(Refer Slide Time: 22:22)



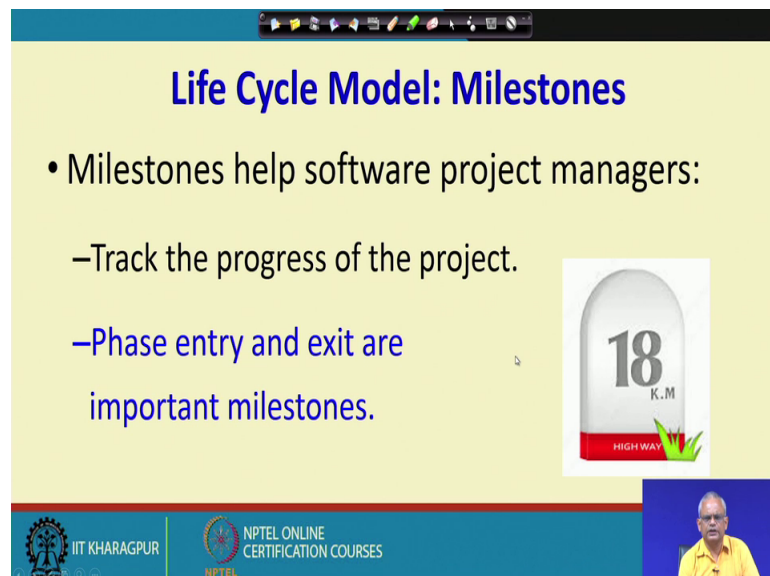
Life Cycle Model (CONT.)

- What is the phase exit criteria for the software requirements specification phase?
 - Software Requirements Specification (SRS) document is complete, reviewed, and approved by the customer.
- A phase can start:
 - Only if its phase-entry criteria have been satisfied.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES


The exit criteria for the requirement specification phase is that requirements should have been gathered should have been analyzed, should have been documented, internally reviewed by the team members, all errors in the requirements must have been corrected and finally, the customer should have approved the requirements. The phase can start only after the phase entry criteria have been satisfied.

(Refer Slide Time: 23:03)



Life Cycle Model: Milestones

- Milestones help software project managers:
 - Track the progress of the project.
 - Phase entry and exit are important milestones.

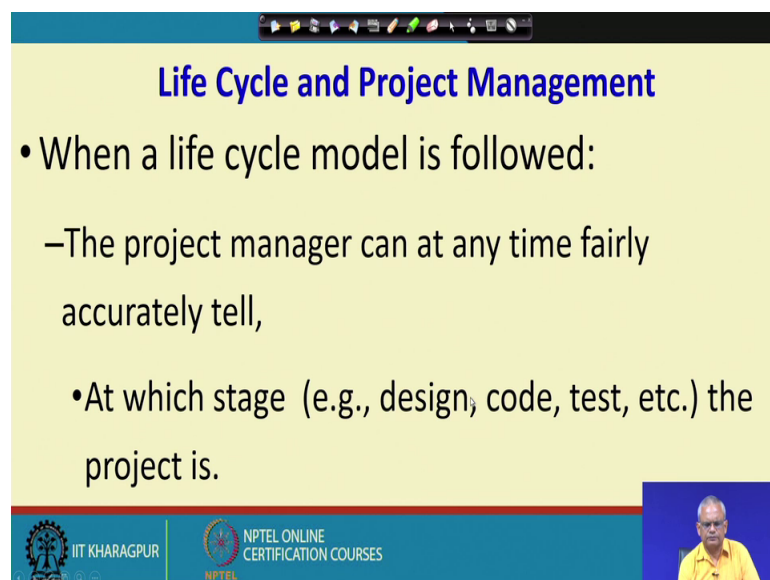


IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

One of the main advantages of having the phase entry and exit criteria is that it helps us define milestones. A milestone is an important event during the development of software. For example, if the phase exit criteria is satisfied then an important milestone is reached. The advantage of having milestones is that the project manager knows how far the project has progressed and it becomes easy for him to plan, monitor the project, whether it will be able to meet the promise that was given to the customer about the delivery or he has to put more manpower or outsource some part and so on.

The phase start and end are important milestones, that is, entry and exit are important milestones and it helps to track the project effectively.

(Refer Slide Time: 24:26)



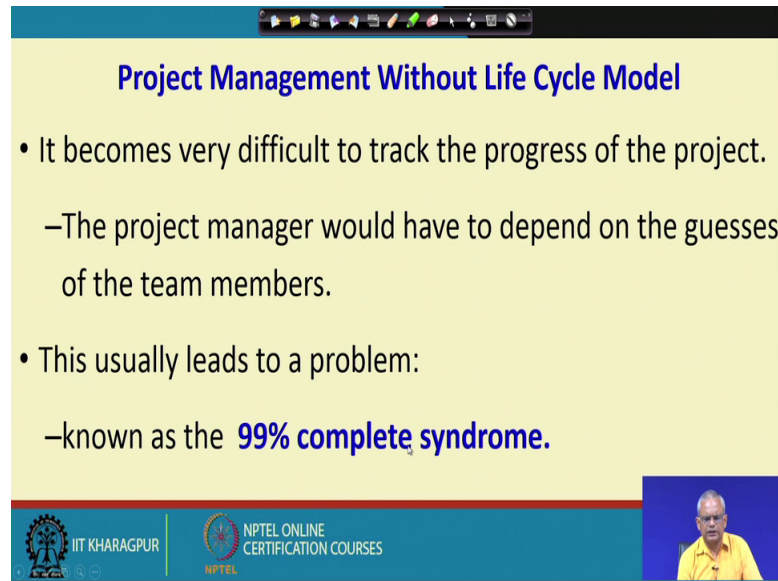
Life Cycle and Project Management

- When a life cycle model is followed:
 - The project manager can at any time fairly accurately tell,
 - At which stage (e.g., design, code, test, etc.) the project is.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

If we do not have such milestones then the project manager is at a loss, does not know how far the project has progressed, he can just ask the team members that how long do you think it will take? How far you have completed?

(Refer Slide Time: 24:49)



Project Management Without Life Cycle Model

- It becomes very difficult to track the progress of the project.
 - The project manager would have to depend on the guesses of the team members.
- This usually leads to a problem:
 - known as the **99% complete syndrome.**

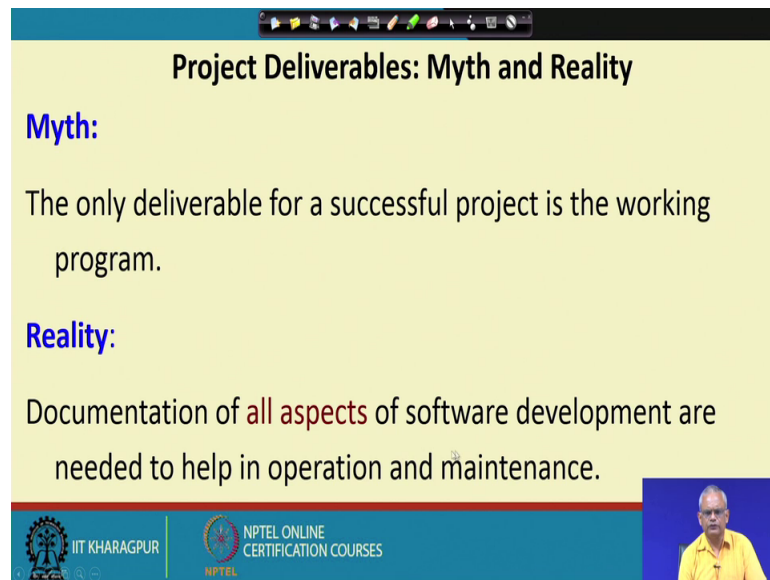
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, the problem is that in such guess of the team members where the project manager has to rely on the guess of the team members. The programmers are normally very optimistic and the programmers typically say that see almost done.

This used to be a big problem when the lifecycle model was not being followed and this problem is known as the 99 percent complete syndrome. That is whenever the project manager asked to the team member, they say that almost done 99 percent done, because they are highly optimistic. But then it just keeps on taking more and more time the project manager becomes impatient asks why it is taking time they say that some bug, some reason, they give something has to be changed and so on.

So, the project management without a lifecycle model is very difficult and suffers from the 99 percent complete syndrome.

(Refer Slide Time: 25:58)



Project Deliverables: Myth and Reality

Myth:

The only deliverable for a successful project is the working program.

Reality:

Documentation of **all aspects** of software development are needed to help in operation and maintenance.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

NPTEL

The slide features a yellow background with a blue header and footer. A small video feed of a man in a yellow shirt is visible in the bottom right corner. The footer contains logos for IIT Khargapur and NPTEL.

But, at the as the development precedes what are the documents that are produced? Is it that the code is the most important document the most important active artifact and the only code is to be given to the customer, not really.

It is not the working program that is given to the customer it is all the documents all documentation needs to be developed. For example, the specification document, design document, the test document, the maintenance document the user's manual and so on. All these documents have to be developed during the development process, it is not just the code, code is just one part of the deliverables.

(Refer Slide Time: 27:08)

The slide is titled "Life Cycle Model (CONT.)" in a yellow box. It contains the following text:

- Many life cycle models have been proposed.
- We confine our attention to only a few commonly used models.

Below these points, a list of models is shown, grouped by a red bracket labeled "Traditional models":

- Waterfall
- V model,
- Evolutionary,
- Prototyping
- Spiral model,

The slide footer includes the IIT Kharagpur logo and the NPTEL Online Certification Courses logo. A small video inset of the presenter is visible in the bottom right corner.

So, far we have seen some very basic concepts on the life cycle the phases phase entry and exit criteria milestones and so on. Now, we will start looking at the important life cycle models, the traditional models, like waterfall V model, evolutionary prototyping and spiral model and also the modern agile models. But right now, we are just nearing completion of this lecture we will break here and we will start from this point in the next lecture.

Thank you.