

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

Lecture - 06
Life Cycle Models – II

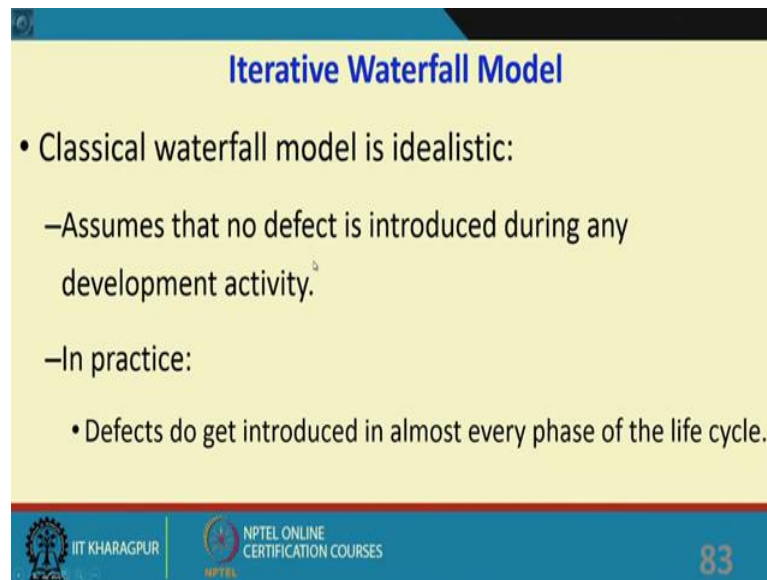
Welcome to this lecture. In the last lecture we had discussed about the Life Cycle Model and especially the classical waterfall model and as part of the life cycle model we had looked at various activities that are undertaken during the development process. And we had spent some time on the feasibility study which is a important activity undertaken by the project manager with the help of a case study we had identified what are the activities that are taken up by the project manager during feasibility study and at the end of the feasibility study the business case is written. This is one of the initiating processes writing the business case and we had seen that how an effective business case can be written.

(Refer Slide Time: 01:24)



Now, let us proceed further in this lecture based on the discussions that we had the last lecture we will discuss about the iterative waterfall model we will look at the V model, evolutionary model and prototyping model that is the plan for this lecture.

(Refer Slide Time: 01:42)



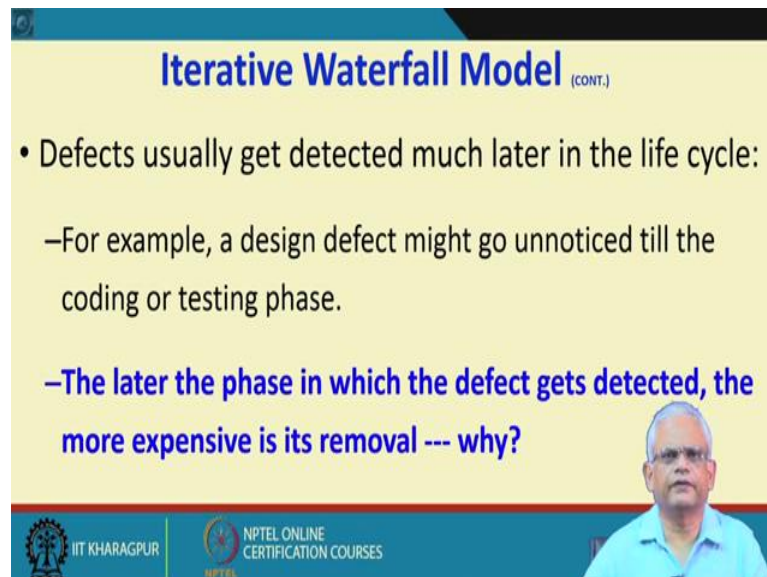
Iterative Waterfall Model

- Classical waterfall model is idealistic:
 - Assumes that no defect is introduced during any development activity.
 - In practice:
 - Defects do get introduced in almost every phase of the life cycle.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 83

The classical waterfall model is an idealistic model, the main problem is that it assumes no defect is introduced during any phase each phase is completed and the next phase starts and so on until the project completes, but in effect there are lot of defects that are introduced in almost every phase of the lifecycle.


(Refer Slide Time: 02:16)



Iterative Waterfall Model (CONT.)

- Defects usually get detected much later in the life cycle:
 - For example, a design defect might go unnoticed till the coding or testing phase.
 - The later the phase in which the defect gets detected, the more expensive is its removal --- why?**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

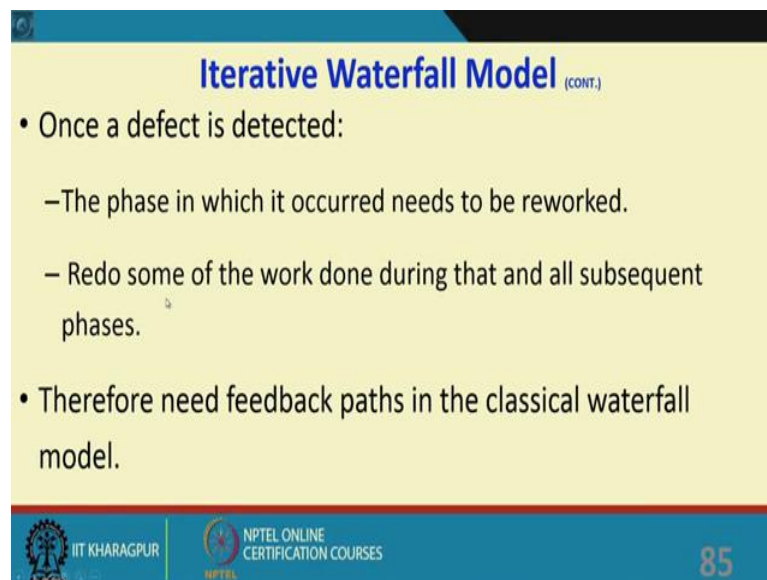


And that is the reason why it may be necessary to go back to a previous phase once a defect is identified. For example, a defect may be identified in the design phase that a requirement specification a functionality was missing and then need to take up the

requirements activity rework the requirement specification document and then again come back to the design phase, that is the reason why we need the feedback path. Once the defect occurs in a phase there is a mistake by a developer it often remains unnoticed until a later phase and then once it is noticed in a later phase, it is then removed.

But the question here is that the later the phase the more expensive is to correct this defect, why is that? The answer is not very far to seek because if defect occurs and it is removed immediately, then that is the best thing the cost will be minimal, but let us say the defect is removed after one phase that is the next phase then only two phases are affected; only two phase documents need to be changed. But if let us say a defect is occurs in the requirement specification phase and it is discovered during testing, then not only the requirement specification document has to change the design needs to be changed, the code needs to be changed and these are lot of activities and lot of documents need to change. And that is the reason why the later the phase in which the defect gets detected the more expensive is the removal of the defect this is a very important concept and it is known as the phase containment of errors.

(Refer Slide Time: 05:54)



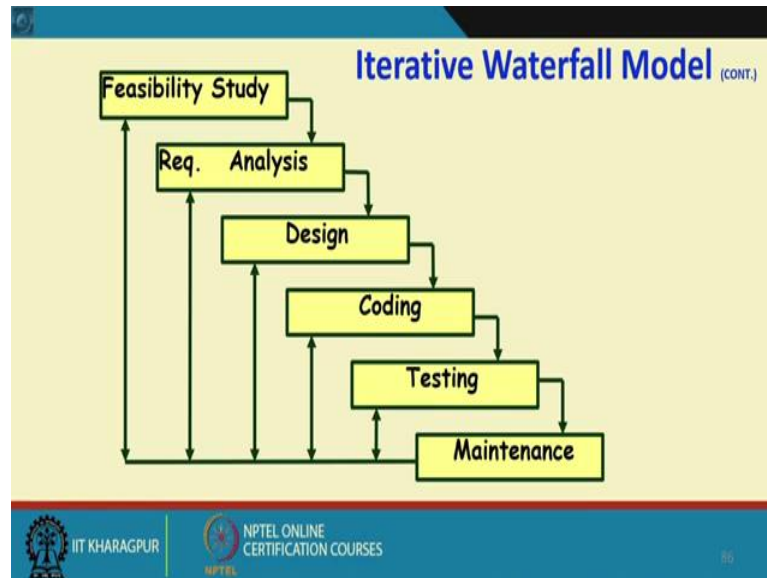
Iterative Waterfall Model (CONT.)

- Once a defect is detected:
 - The phase in which it occurred needs to be reworked.
 - Redo some of the work done during that and all subsequent phases.
- Therefore need feedback paths in the classical waterfall model.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 85

So, that is what is written here that once the defect is detected redo the work in the previous phases. And therefore, if there are lot of intermediate phases it has crossed many phases before the defect is detected, then lot of rework is needed.

(Refer Slide Time: 05:24)



In the iterative waterfall model is very similar to the classical waterfall model, so you can see the waterfall here, but also there are feedback paths; this feedback paths make the model realistic unlike the classical model where one phase completes and then the next phase completes and so on. Here we assume that there can be a defect which will get noticed during the testing phase and we need to go back to the design phase or maybe the requirement analysis phase and then fix that defect and not only that fix the work in the later phases as well.

(Refer Slide Time: 06:17)

The slide is titled 'Phase Containment of Errors (Cont...)'. It contains the following bullet points:

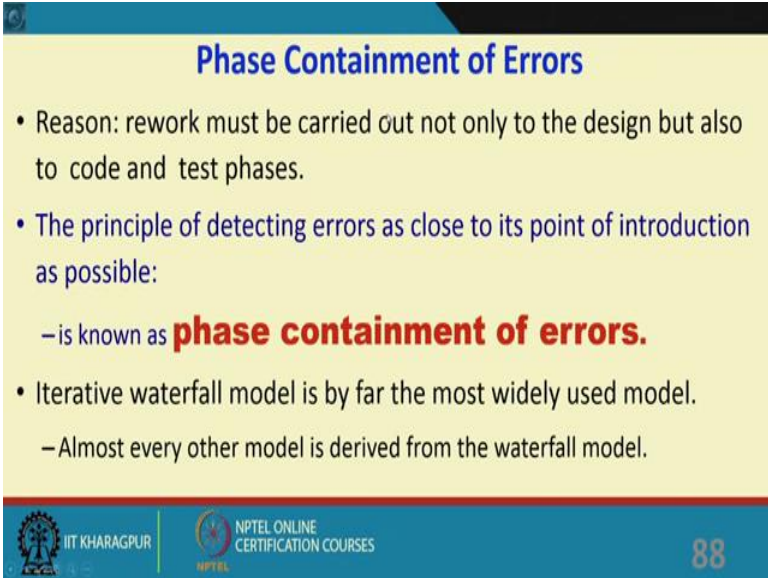
- **Errors should be detected:**
 - **In the same phase in which they are introduced.**
- For example:
 - If a design problem is detected in the design phase itself,
 - The problem can be taken care of much more easily
 - Than say if it is identified at the end of the integration and system testing phase.

The footer includes the IIT KHARAGPUR logo and the text 'NPTEL ONLINE CERTIFICATION COURSES'. The slide number '87' is located in the bottom right corner.

As we are saying that it is an important concept for the errors when they are the mistakes are made by the developers they must be fixed in the same phase that is the ideal that the defects are detected and fixed in the same phase that will incur the lowest cost. The more delay occurs in detecting the defect, the more expensive it will be.

If a design defect is detected in the design phase itself, then it just needs to change the design document, but if it is detected during the testing phase not only the design document needs to be changed, but the code needs to be reworked and that will be much more expensive. So, this is called as the phase containment of errors it is an important principle that, once a mistake occurs mistake or error occurs this must be contained to the same phase it is called as a phase containment of errors.

(Refer Slide Time: 07:27)



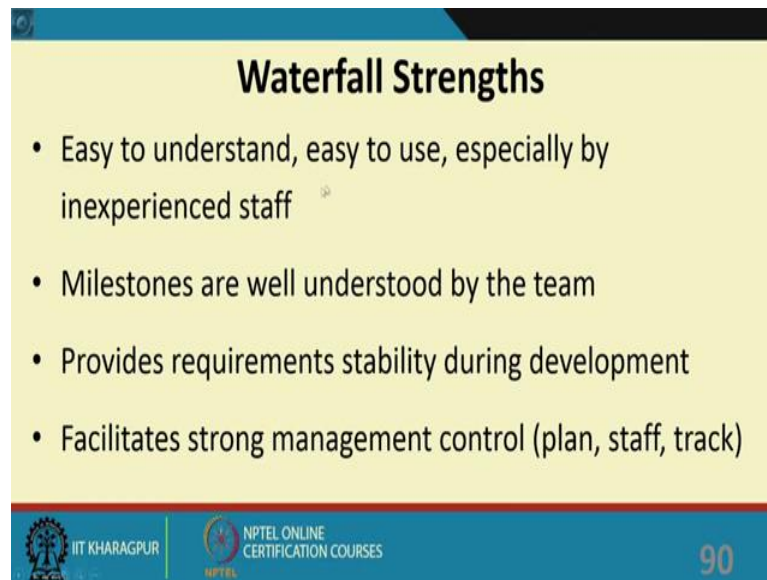
Phase Containment of Errors

- Reason: rework must be carried out not only to the design but also to code and test phases.
- The principle of detecting errors as close to its point of introduction as possible:
 - is known as **phase containment of errors.**
- Iterative waterfall model is by far the most widely used model.
 - Almost every other model is derived from the waterfall model.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 88

The principle of detecting errors as close to its point of introduction is called as the phase containment of errors it is a very important principle and very intuitive also. So, this is the iterative waterfall model, the phases are identical to the classical model excepting that the feedback paths are available.

(Refer Slide Time: 07:57)



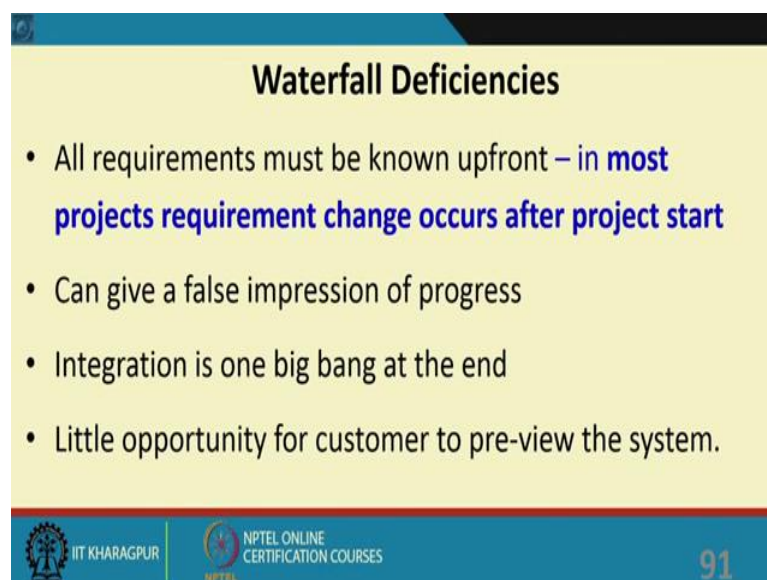
Waterfall Strengths

- Easy to understand, easy to use, especially by inexperienced staff
- Milestones are well understood by the team
- Provides requirements stability during development
- Facilitates strong management control (plan, staff, track)

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 90

The main strengths of the waterfall model is it is very intuitive easy to understand all the developers can very easily understand and use the model. The milestones are clear understood by the team and the project manager can easily monitor the progress, it provides the required stability. Management the project manager can have a strong control put scheduled milestones and monitor, but then it has lot of deficiencies.

(Refer Slide Time: 08:39)



Waterfall Deficiencies

- All requirements must be known upfront – **in most projects requirement change occurs after project start**
- Can give a false impression of progress
- Integration is one big bang at the end
- Little opportunity for customer to pre-view the system.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 91

One of the major problem with the waterfall model is that it is not very suitable to handle changes to the functionalities, but in almost every software project the requirements

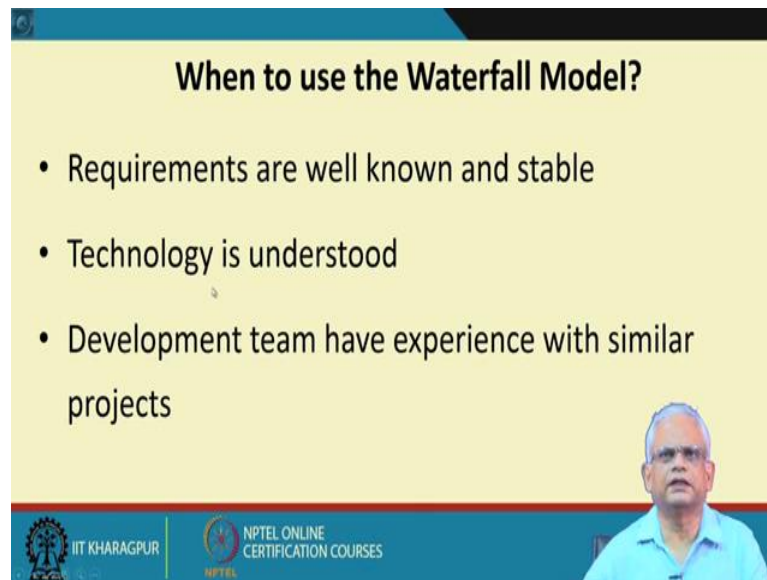
change there are many reasons why the requirement may change. For example, the customer may not be may not have thought that some functionalities will be required because he did not understand or did not think you cannot grasp what are all required at the beginning of the project, but as he looks at functionalities then he can make out that certain functions will be required. The other is that as the development takes place, the business itself might undergo some change and therefore, changes will be required.

So, there are many reasons why changes will be required during the development of the software, but in the waterfall model the requirements are fixed upfront and then the design coding and testing are undertaken and there is no way provided by the waterfall model to make any changes, this is possibly the biggest shortcoming of the waterfall model. The other shortcomings are that here lot of documents are produced phases are completed, the requirements is complete, design is complete documents are produced, code is complete, but during testing found that most of the requirements are not met.

So, the project manager gets a false impression of the progress that, the progress is the project is proceeding nicely according to the plan, but during the testing phase finds out that very little has been done. The main reason for that is that, the integration is a big bang at the end till then everything was fine, but at during integration it has found that most of the functionalities are not working.

Another problem with this model is that, once the customer gives the requirement they have no chance to see the system till the end when it is delivered to them. It would have been really good if the customers were involved they could see that what are the functionalities that's they are being developed they could have suggested modifications and so on. These are the major difficulties with the waterfall model the biggest one is that it does not have the flexibility to handle the requirement change, false impression of progress because phases are getting complete, but integration is on big bang at the end where it is noticed that the progress is very less; most of the functionalities are not working and the customer involvement is very minimal, the customer cannot give suggestions.

(Refer Slide Time: 12:28)



When to use the Waterfall Model?

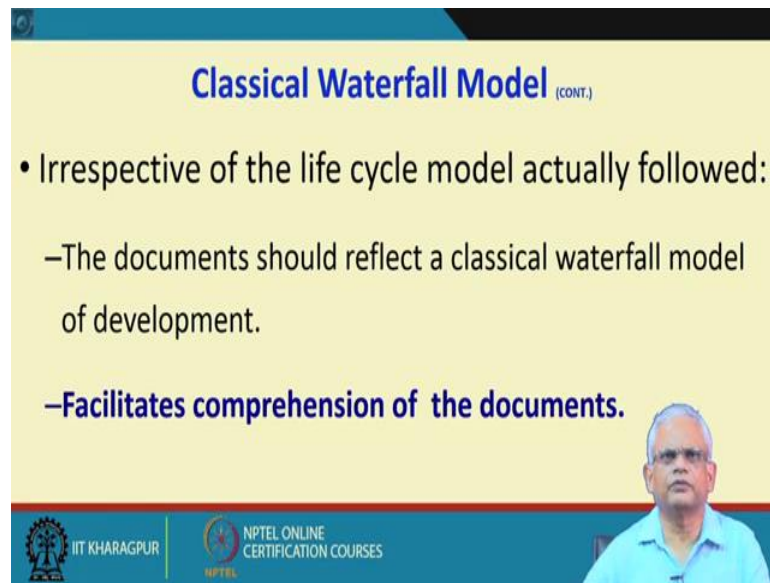
- Requirements are well known and stable
- Technology is understood
- Development team have experience with similar projects

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But then the waterfall model was popular and it is also successfully used in some types of projects. The projects for which waterfall model is suitable is that, the requirements are very well understood and known to the customer they have used similar software they know what all is required, it is not that something they are thinking of a new type of software it is a regular software. Requirements are well known and stable, technology is understood, development team have experience with similar projects.

So, as you can see here that the project is rather routine project where the development team have done similar work, they are just trying to develop one more of that you can say that the challenges here are less because the requirements are known what the software will finally, look like what functionalities is very well known. The technology is understood, development team had experience in similar projects and for such simple projects waterfall model can be used and this happens to be the most efficient way to complete these projects.


(Refer Slide Time: 14:13)



Classical Waterfall Model (CONT.)

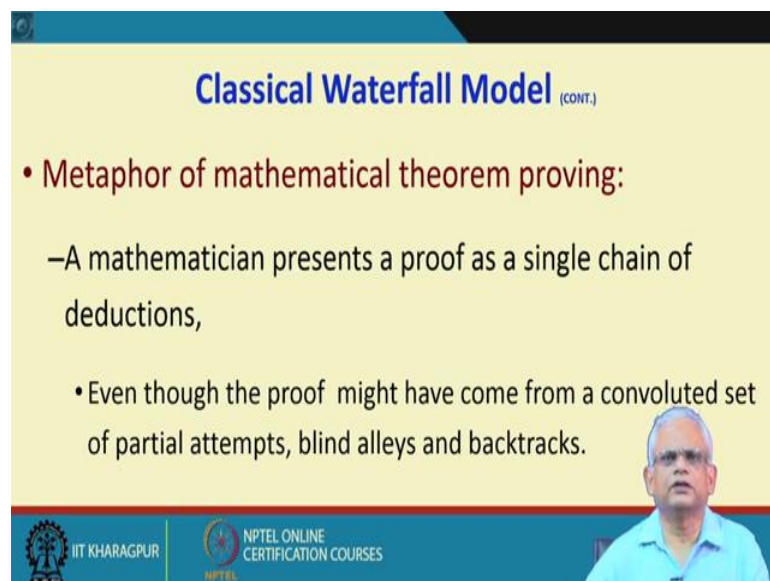
- Irrespective of the life cycle model actually followed:
 - The documents should reflect a classical waterfall model of development.
 - Facilitates comprehension of the documents.**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES



But what about the classical waterfall model, the iterative waterfall model has its use we saw that for simple projects they can be used, but what about the classical waterfall model? Because here this was the idealistic model where it was assumed that there is no defect in any phase all phase proceed idealistically, but the use of this is that the final documentation should appear like the software is developed in a using a classical waterfall model, there are no backtracking no mistakes nothing and if the document is written that way then it facilitates comprehension of the documents.


(Refer Slide Time: 15:06)



Classical Waterfall Model (CONT.)

- **Metaphor of mathematical theorem proving:**
 - A mathematician presents a proof as a single chain of deductions,
 - Even though the proof might have come from a convoluted set of partial attempts, blind alleys and backtracks.

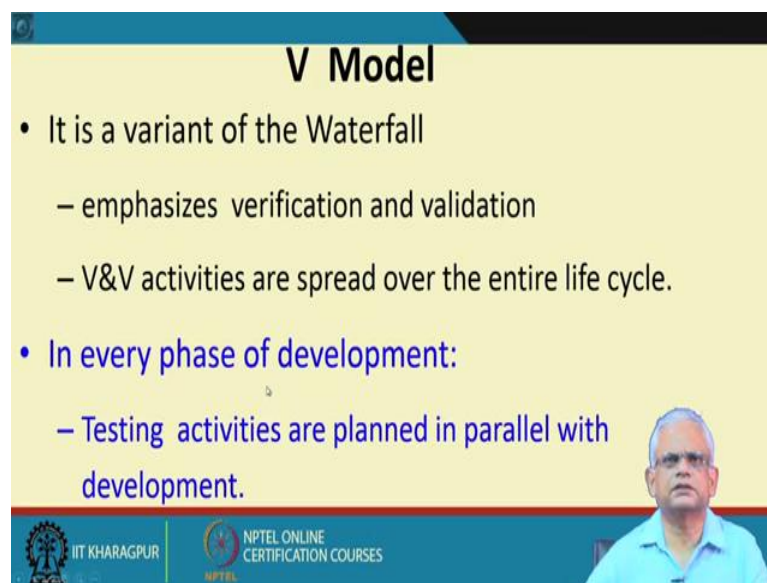
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES



Just to give an analogy you may be trying to do a mathematical theorem proving or maybe trying to solve a problem mathematical problem and you might many times do rework mistakes, go back to the previous phase previous step and so on. But if your final document for this problem has all these things that where you did mistake where you went back corrected and so on, then it becomes extremely complicated for somebody trying to understand.

The best for somebody to understand is that you give him the correct steps as if you had not committed any mistakes and so on and that is what is basically the classical waterfall model that all phases proceed without any mistakes. Now, let us look at the V model this is a variant of the waterfall model only small changes are there small derivative of the waterfall model essentially all the activities are there the waterfall model.

(Refer Slide Time: 16:37)



V Model

- It is a variant of the Waterfall
 - emphasizes verification and validation
 - V&V activities are spread over the entire life cycle.
- In every phase of development:
 - Testing activities are planned in parallel with development.

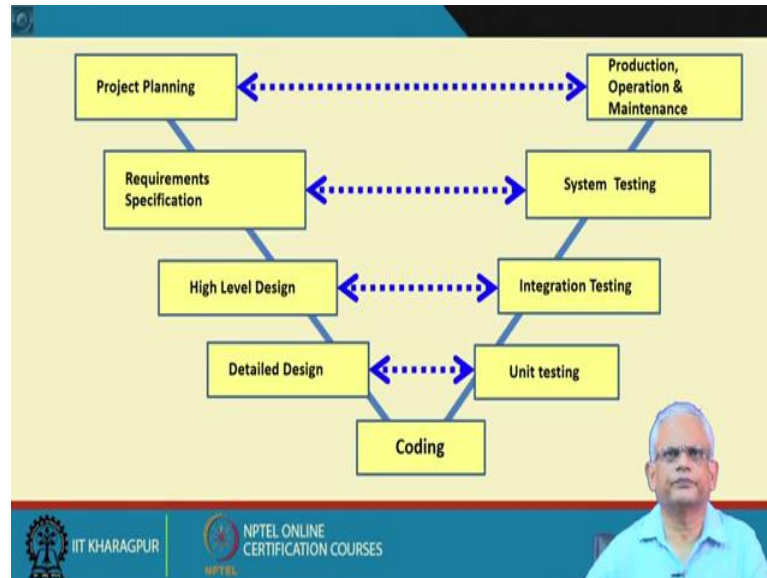
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, let us look at the V model, the v model emphasizes on the verification and validation activities throughout the lifecycle. We had seen that in the waterfall model the testing activity was only at the end, initially there is requirements specification then there is design coding, but the testing activities are only at the end and that is the reason why the testing activities take long time. And the other problem is that what do the testers do, till the testing phase starts what do they do?

The V model gives answer to that, it emphasizes verification and validation and normally this model is used where the software is supposed to be more reliable. For example,

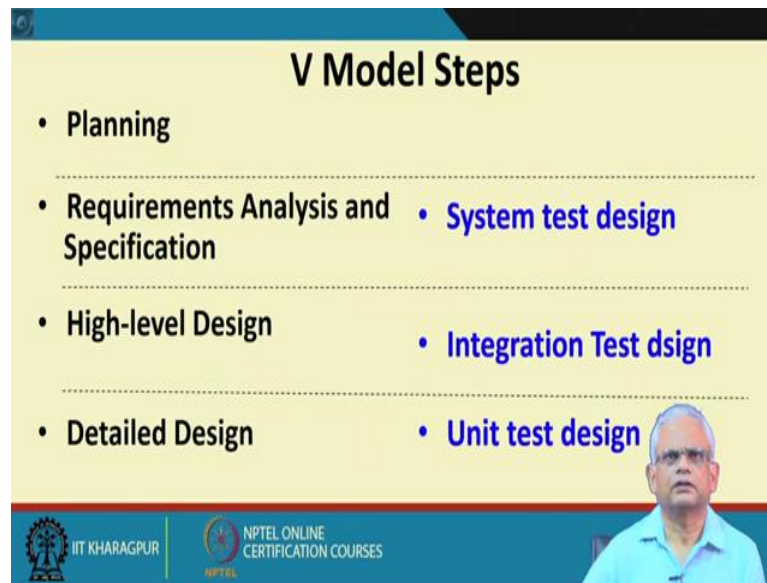
those software being used in critical applications the V model is used. Here in every phase of development there are some test activities for example, planning the test cases and later they may get executed, but every phase there are some test activities that occur.

(Refer Slide Time: 18:14)



So, this is the diagrammatic representation of the V model looks like a V here the model representation and that is the reason why it is called as a V model. During the requirement specification, the system test cases are developed which are executed during system testing during high level design the integration test cases are developed, during the detailed design the unit test cases are developed which are used during the unit testing after the coding is complete. As you can see here that there is a emphasis on verification and validation activities and every phase there are some test activities unlike the waterfall model where the test activities occurred only when only in the integration and system testing.

(Refer Slide Time: 19:14)



V Model Steps

- Planning
- Requirements Analysis and Specification
- High-level Design
- Detailed Design

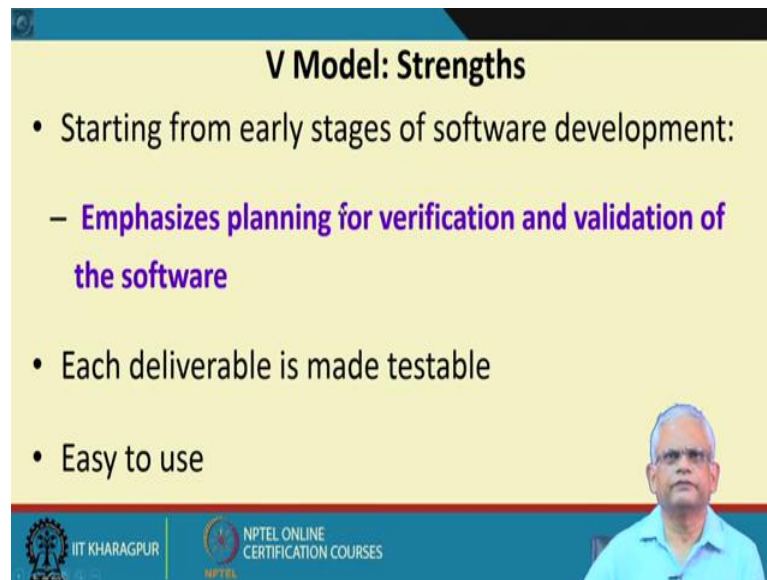
- System test design
- Integration Test design
- Unit test design

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The slide features a yellow background with a blue header and footer. A small portrait of a man in a light blue shirt is visible in the bottom right corner of the slide content area.

Here this slide it just says what are the system the test activities occur during the requirement analysis and specification system test case design activities occur, high level design the integration test design activities occur and during the detailed design the unit tests design activities occur.

(Refer Slide Time: 19:38)



V Model: Strengths

- Starting from early stages of software development:
 - Emphasizes planning for verification and validation of the software
- Each deliverable is made testable
- Easy to use

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

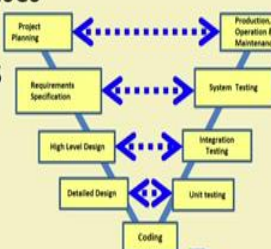
The slide features a yellow background with a blue header and footer. A small portrait of a man in a light blue shirt is visible in the bottom right corner of the slide content area.

The strength of this model is that it emphasizes verification and validation, its very similar to the waterfall model excepting that every phase has some verification and validation activities.

(Refer Slide Time: 19:58)

V Model Weaknesses

- Does not support overlapping of phases
- Does not handle iterations or phases
- Does not easily accommodate later changes to requirements
- Does not provide support for effective risk handling



The diagram illustrates the V Model, where the left side represents development phases: Project Planning, Requirements Specification, High Level Design, Detailed Design, and Coding. The right side represents testing phases: Production, Operation & Maintenance, System Testing, Integration Testing, and Unit testing. A central diamond shape connects Detailed Design and Unit testing. Blue dashed double-headed arrows indicate bidirectional relationships between Project Planning and Production, Operation & Maintenance; Requirements Specification and System Testing; and High Level Design and Integration Testing.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But then it has many weaknesses just like the waterfall model here it does not support overlapping of the phases that is one phase has to complete then the next phase starts, what if some of the developers few of the developers complete first should they wait for the other developers to complete just idle or they can get started with the next phase this model does not allow that, they wait for all phase the phase to complete and only then they can start the next phase. Does not easily accommodate later change requirements, does not provide support for effective risk handling.

(Refer Slide Time: 20:52)

When to use V Model

- Natural choice for systems requiring high reliability:
 - Embedded control applications, safety-critical software
- All requirements are known up-front
- Solution and technology are known

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, these are some of the problems of the V model, but then it is the natural choice for systems requiring high reliability and it is used when all requirements are known upfront and the technology is known. Now, let us look at the prototyping model which is also a small variation of the basic waterfall model.

(Refer Slide Time: 21:23)

Prototyping Model

- A derivative of waterfall model.
- Before starting actual development,
 - A working prototype of the system should first be built.
- A prototype is a toy implementation of a system:
 - Limited functional capabilities,
 - Low reliability,

```
graph TD; PC[Prototype Construction] --> D[Design]; D --> C[Coding]; C --> T[Testing]; T --> M[Maintenance]; T --> D;
```

The diagram illustrates the Prototyping Model as a series of five steps: Prototype Construction, Design, Coding, Testing, and Maintenance. The steps are arranged in a descending staircase pattern from top-left to bottom-right. Arrows connect the steps in a downward sequence. A feedback loop is shown with an arrow pointing from the Testing step back up to the Design step.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

If we look at the diagrammatic representation of the prototyping model the initial phase here is prototype construction and after that the design coding etcetera are carried out. Here just like the waterfall model the activities are organized, but then before starting the actual development a working prototype must be built. What is a prototype? A prototype is a toy implementation of the system which has limited functionalities, low reliability and so on. But the question is that why do we need to develop a prototype before we start to develop a system, what are the advantages of developing the prototype?

(Refer Slide Time: 22:24)

Reasons for prototyping

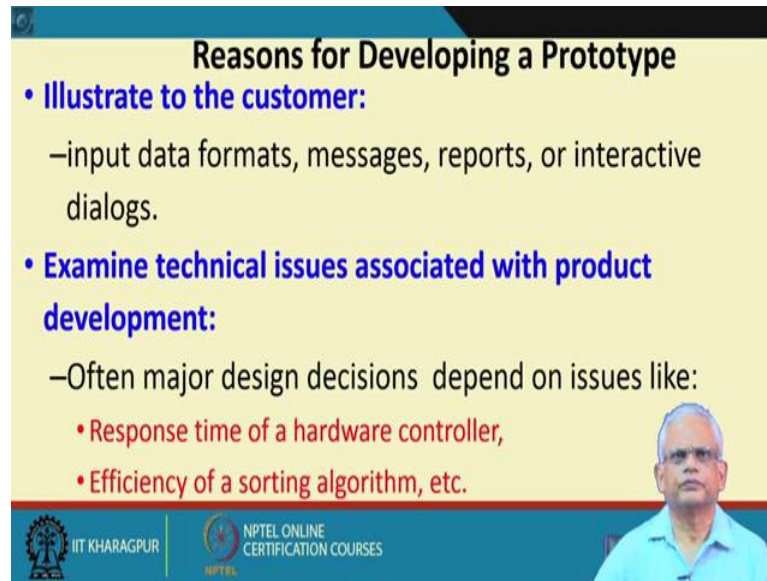
- **Learning by doing:** useful where requirements are only partially known
- **Improved communication**
- **Improved user involvement**
- **Reduced need for documentation**
- **Reduced maintenance costs**

```
graph TD; PC[Prototype Construction] --> D[Design]; D --> C[Coding]; C --> T[Testing]; T --> M[Maintenance]; M --> D; M --> C; M --> T;
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 104

The first thing is that once you develop the prototype we can know what are all the requirements that should be there, if there are any requirements which are missing. If the customer can look at the prototype, then you can find out what are the missing requirements. There is a improved communication with the customer, there is a user involvement because unlike the classical or the iterative waterfall model where the after the requirements are given the only thing that the customer sees the delivered software here in contrast in this model the prototype is constructed and given to the customer for his comments and also since a prototype exists, it reduces the documentation, reduces the maintenance costs. And also if there are technical risks that the development team anticipates then they can through the prototype construction they can address those risks.

(Refer Slide Time: 23:51)



Reasons for Developing a Prototype

- **Illustrate to the customer:**
 - input data formats, messages, reports, or interactive dialogs.
- **Examine technical issues associated with product development:**
 - Often major design decisions depend on issues like:
 - Response time of a hardware controller,
 - Efficiency of a sorting algorithm, etc.

The slide features a blue header with the title, a yellow background for the text, and a blue footer with logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES. A small portrait of a man in a light blue shirt is visible in the bottom right corner of the slide.

The reason for developing the prototype is to illustrate to the customer what the software is going to look like, what will be the input data formats, messages, reports, interactive dialogues. The technical issues associated with the product development, the developers can experiment and find out what is the best way to handle technical issues sometimes the major decisions depends on issues like the response time of the hardware controller.

And here by developing the prototype the designers can have their solution based on the hardware, the response time of the hardware controller at the upfront rather than proceeding throughout all the development activities and finding out that the hardware controller response time is slow and they need to change everything. So, developing a prototype helps in getting the customer feedback and also many technical issues are clarified and the development can proceed correctly.

(Refer Slide Time: 25:31)

Prototyping Model (CONT.)

- Another reason for developing a prototype:
 - It is impossible to “get it right” the first time,
 - We must plan to throw away the first version:
 - If we want to develop a good software.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 106

Also when a prototype is developed the once it is used for illustrating to the customer it is thrown away and the actual system is developed, but then the experience that the development team gets in developing the prototype that goes a long way in developing a good software.

(Refer Slide Time: 26:00)

Prototyping Model

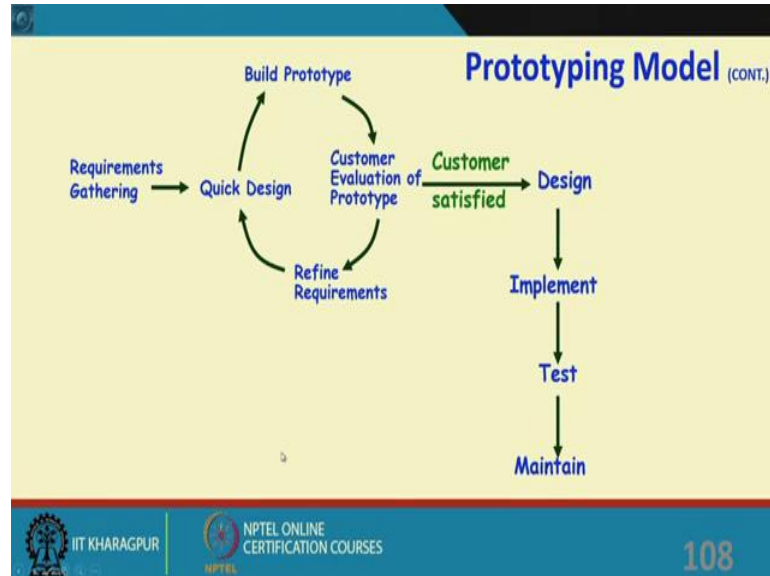
- Start with approximate requirements.
- Carry out a quick design.
- Prototype is built using several short-cuts:
 - Short-cuts might involve:
 - Using inefficient, inaccurate, or dummy functions.
 - A table look-up rather than performing the actual computations.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 107

Here in the prototyping model first the approximate requirements are obtained, a quick design is construct is made and then the prototype is constructed using various shortcuts.

The shortcuts can be like inefficient, inaccurate or dummy functions; a table lookup rather than writing the code for the actual computation.

(Refer Slide Time: 26:34)



We can represent that in this diagram that initially a quick design is done based on the requirements that have been gathered prototype is built. Submitted to the customer for evaluation based on customer feedback, refine the requirements again have a quick design change the prototype and so on until the customer is satisfied with the prototype then a waterfall model is used for developing the software

(Refer Slide Time: 27:08)

Prototyping Model (CONT.)

- The developed prototype is submitted to the customer for his evaluation:
 - Based on the user feedback, the prototype is refined.
 - This cycle continues until the user approves the prototype.
- The actual system is developed using the waterfall model.

The diagram in this slide is a smaller version of the one in slide 108, showing the same process flow from Requirements Gathering to Maintain.

So, the it is a small variation of the waterfall model only the initial prototype construction that is added here.

(Refer Slide Time: 27:22)

Prototyping Model

- Requirements analysis and specification phase becomes redundant:
 - Final working prototype (incorporating all user feedbacks) serves as an **animated requirements specification**.
- **Design and code for the prototype is usually thrown away:**
 - However, experience gathered from developing the prototype helps a great deal while developing the actual software.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 110

And because of the prototype is kind of a requirement specification animated requirement specification, a requirement specification document may not be required, but one thing that we must be clear here is in the prototyping model once the prototype has been used to demonstrate to the customer and address the technical issues, the prototype is thrown away and then the development starts.

(Refer Slide Time: 28:02)

Prototyping Model (CONT.)

- Even though construction of a working prototype model involves additional cost --- **overall development cost usually lower for:**
 - Systems with unclear user requirements,
 - Systems with unresolved technical issues.
- Many user requirements get properly defined and technical issues get resolved:
 - These would have appeared later as change requests and resulted in incurring massive redesign costs.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 111

But does the prototype construction add to the cost and the development becomes more costly? No not really because for systems with unclear user requirements with unresolved technical issues the prototyping is actually a cost effective way of developing the software because without a prototype we might have to change the software many times and that would make it much more expensive that would incur massive redesigned costs.

And even though there is a small upfront cost of developing the prototype, but that will be more effective cost effective, then not using the prototype and finally, finding out that the customer has lot of dissatisfaction and suggests many changes and also the technical issues are unresolved as a result many changes occur. With this discussion we will just stop here and continue discussing a few more lifecycle models in the next lecture.

Thank you.