

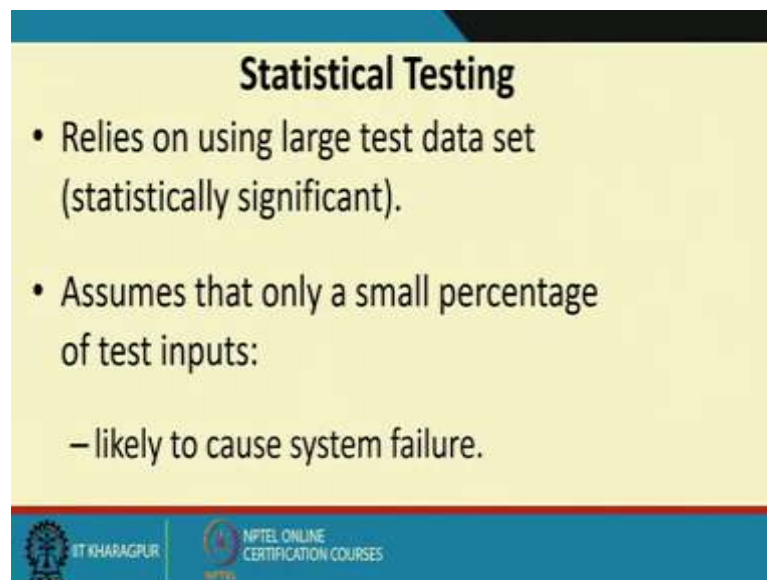
Software Project Management
Prof. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 60
Software Testing

Welcome to this lecture. In the last lecture, we had discussed about how to estimate the reliability of a given software application. One way we had said is a reliability growth modeling. We can consider a model; there are several models have been proposed, we discussed only three models. So, these models are basically graphical representation of the failure rate over time.



And, for the specific software application that we are interested we just collect its failure data over time and then try to plot it and see that which model is the most accurate which fits this our failure data most accurately. And, based on that, we can predict the reliability of the software after certain time at the present time and so on.

(Refer Slide Time: 01:39)



Statistical Testing

- Relies on using large test data set (statistically significant).
- Assumes that only a small percentage of test inputs:
 - likely to cause system failure.

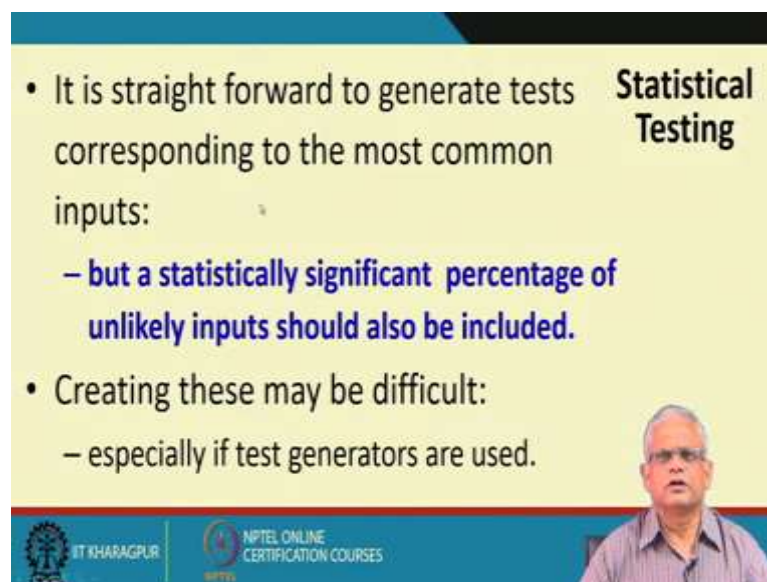
 IIT KHARAGPUR |  NPTEL ONLINE CERTIFICATION COURSES

The second approach we discussed was statistical testing. In statistical testing, we use the operational profile and design test cases. Even though name is testing, here, the objective is not to detect bugs. The objective of statistical testing is to estimate the reliability of the software. Here we design the test cases such that the test data correspond to the frequency of invocation of the functions by the users. If the users on the average invoke

some functionality 40 percent of the time, then our test data should actually have 40 percent test data corresponding to that functionality.

The other important thing is that, it relies on using large test data because as the name says it is a statistical technique, we should have a significantly large data based on which we can infer and predict the reliability after certain time. If we use only handful of data that will give a very wrong result and also we assume that only a small percent of test input is likely to cause system failure. If there are system failures occurring in almost every test cases, then of course, the prediction of reliability will not only be poor, but it will be very gross it won't be accurate.

(Refer Slide Time: 03:45)



The slide is titled "Statistical Testing" in the top right corner. It contains the following text:

- It is straight forward to generate tests corresponding to the most common inputs:
 - but a statistically significant percentage of unlikely inputs should also be included.
- Creating these may be difficult:
 - especially if test generators are used.

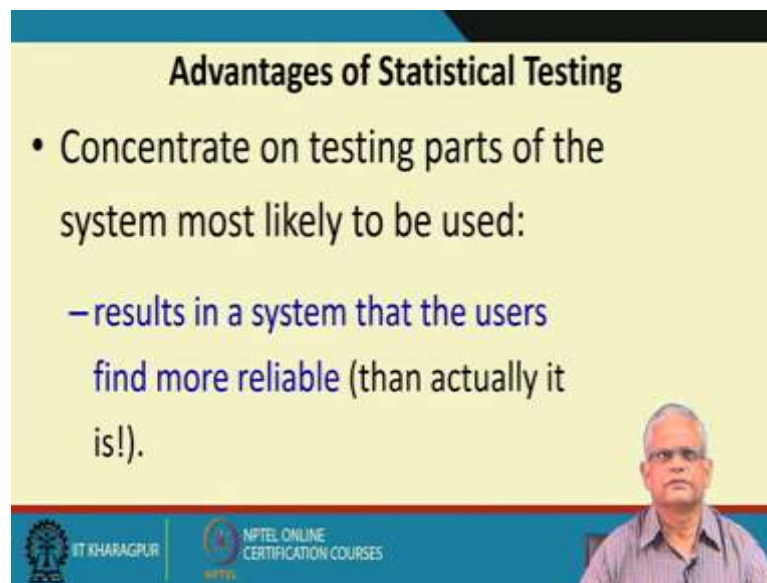
In the bottom right corner, there is a small video feed of a man with glasses speaking. The bottom of the slide features logos for "IIT KHARAGPUR" and "NPTEL ONLINE CERTIFICATION COURSES".

But, the complexity of using statistical testing is that how do we generate test data because we need a significantly large number of data. It should be statistically significant number of data and not only that if we give data that are most likely to be input by the user that is ok, but then we must have a significant percentage of unlikely input data should be also included.

These basically correspond to the corner cases that is the input value that the users give on normal operation that is fine, but then we should also include the input values or the combination of the input values which the users normally do not give, but may sometime give and those are the ones which actually cause failures. Those are called as the corner test cases.

But, creating those test cases are actually the challenge. The normal values that we can easily generate, but the unlikely values that becomes a big challenge and the statistical testing, the accuracy of the result depends largely on whether we have taken care to have enough corner cases, significant percentage of these corner cases.

(Refer Slide Time: 05:31)



Advantages of Statistical Testing

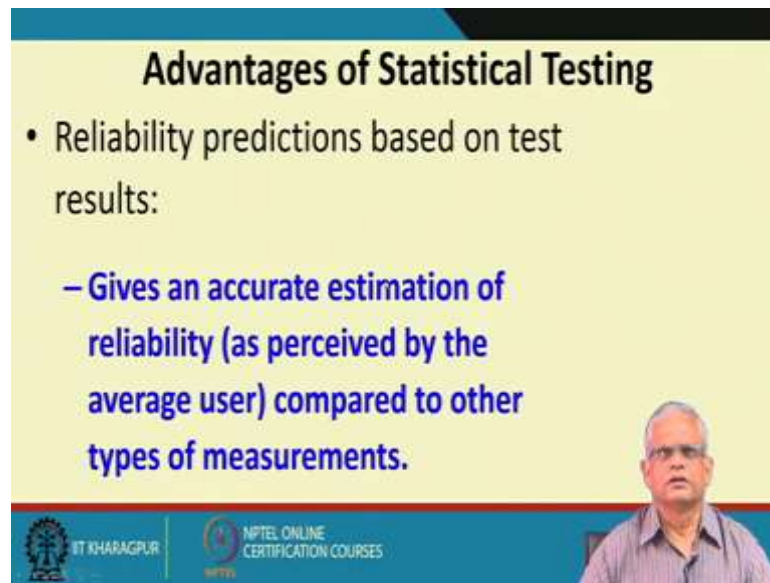
- Concentrate on testing parts of the system most likely to be used:
 - results in a system that the users find more reliable (than actually it is!).

The slide features a video inset of a man with glasses in the bottom right corner. At the bottom, there are logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

Here we give a large number of input for the ones which are frequently used by the average user and therefore, the average user will find that the system is portrays his estimation or his understanding of the reliability of the software. And, here the idea is to test the parts that are frequently used with more test cases and therefore, those are the bugs in those areas are removed.

The frequently used functions they were fine because lot of test cases have been designed for that and therefore, not only that we get a reliability estimation, but also significant number of bugs are removed and that too from the frequently used functions. And, therefore, the reliability of the software appears to be higher compared to when the bugs are removed uniformly from all functions. Here we are concentrating on testing the function which are most frequently used.

(Refer Slide Time: 07:03)



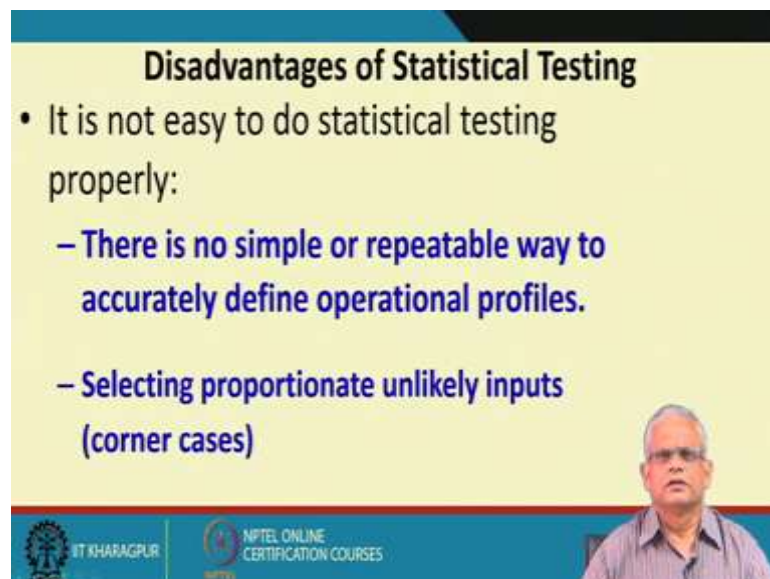
Advantages of Statistical Testing

- Reliability predictions based on test results:
 - Gives an accurate estimation of reliability (as perceived by the average user) compared to other types of measurements.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And, here we define the test data based on the operational profile and therefore, it gives a accurate estimation of the reliability as perceived by the average user compared to the other type of measurement because the test data is generated based on the operational profile.

(Refer Slide Time: 07:29)



Disadvantages of Statistical Testing

- It is not easy to do statistical testing properly:
 - There is no simple or repeatable way to accurately define operational profiles.
 - Selecting proportionate unlikely inputs (corner cases)

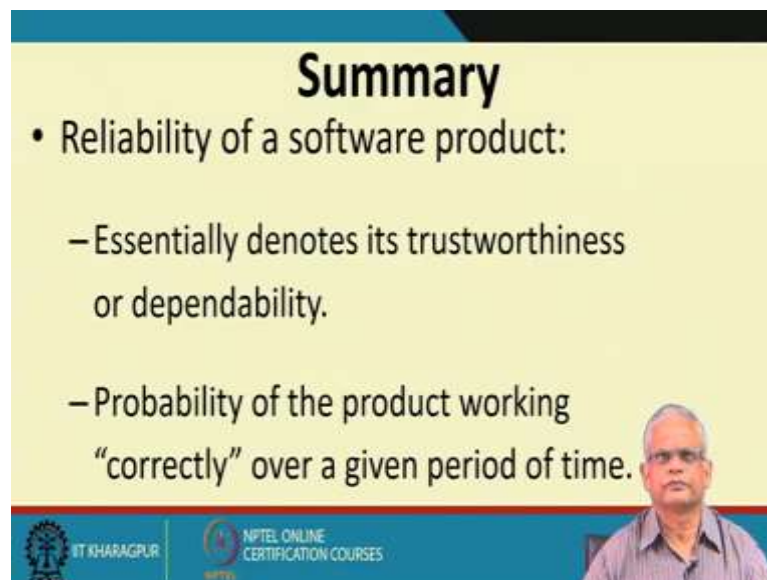
IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, if we think of the disadvantages of the statistical testing one is that the results to a large extent depend on how do we define the operational profile and also how do we design the test cases based on the operational profile. Even if we define the operational

profile by a laborious study where we collect the logs across a large number of users; in actual users scenario of the software, but then even if we have the operational profile correctly the first challenge is to define the operational profile. It has to be collected from the actual users situation and also a large interval of users.

The second problem is that even if we have the operational profile correct, how do we design the test cases. Designing the test cases that are rather usual test values is not difficult, but selecting a proportionate unlikely input or the corner cases is a big challenge and if we do not do this the results would not be accurate.

(Refer Slide Time: 09:03)



Summary

- Reliability of a software product:
 - Essentially denotes its trustworthiness or dependability.
 - Probability of the product working “correctly” over a given period of time.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

In the summary of our reliability study we can say that the reliability of a software can be defined as the trust worthiness or dependability. More accurately this is a very overall definition trust worthiness or dependability is their intuitive idea of the reliability of a software. More accurately we can say that the probability of the product working correctly over a given period of time.

(Refer Slide Time: 09:41)

Summary

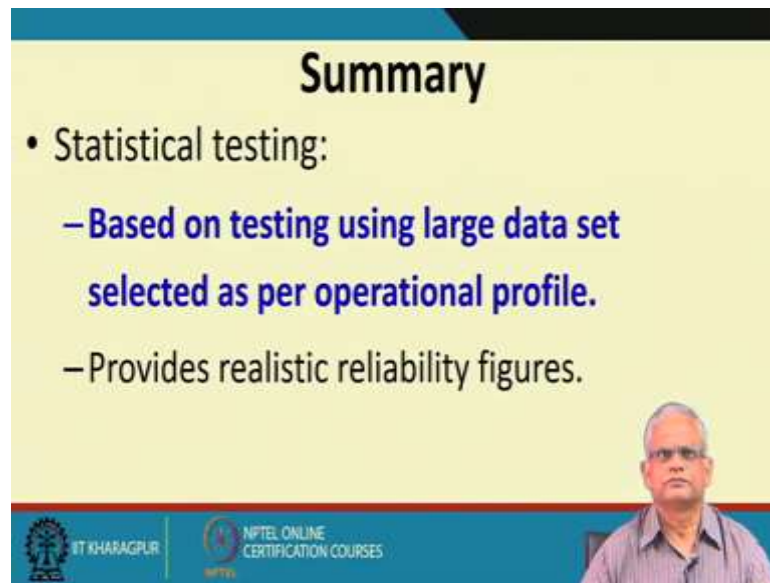
- Operational profile of a software
 - Reflects how it will be used in practice.
 - Consists of specification of:
 - classes of inputs
 - probability of their occurrence.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And, we had seen the problems in software reliability measurement. It is seen that the reliability changes as bugs are detected, unlike the hardware and not only that it is observe dependent. The reliability of a software depends on who is using and therefore, we had said that we must use the operational profile to estimate the reliability.

Our test cases should be designed based on the operational profile of the software then we can get a result of the reliability which is agreeable across most of the users and the operational profile is based on the different classes of input and the probability of their occurrence.

(Refer Slide Time: 10:39)



Summary

- Statistical testing:
 - Based on testing using large data set selected as per operational profile.
 - Provides realistic reliability figures.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And, the statistical testing is based on testing using a large data set. Tens of thousands or millions of data, test data which are designed based on the operational profile to get a very realistic figure. We will now discuss about software testing. The project manager needs to also understand the software testing issues because finally, getting a good quality product testing is a major contributor.

And, to manage the software, so that the software is a good quality the project manager needs to understand some of the basic issues in software testing. With that motivation, we will discuss some issues on software testing.

(Refer Slide Time: 11:41)

Faults and Failures

- A program may fail during testing:
 - Essentially a manifestation of a fault (also called defect or bug).
 - **Mere presence of a fault may not lead to a failure.**


The slide also features a screenshot of a Windows error dialog box titled "PCFTest - Fatal error" with the message "CLR error 80004005. The program will now terminate." and an "OK" button. At the bottom right, there is a small video inset of a man speaking. The footer includes the logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

The first thing we will discuss is the difference between a fault and a failure. When a program is being tested it may fail. For example, on a given input it may display fatal error, the program will not now terminate. What we are observing here is not a bug, if we gave the input test input and we observed that this message came that the program will now terminate fatal error. We are observing the failure not the error.

So, what we observe on testing is a failure, but the failure is caused by a defect, bug or a fault. In other words, we say that a failure is a manifestation of a fault. There is a bug somewhere in the code and that cause the failure, but we observed the tester the failure. But, then, do every bug in the software cause a failure? No, there may be bugs which may never cause a failure. We will just discuss about that little later. You can also think about it that how can a bug be present, but not cause a failure.

(Refer Slide Time: 13:25)

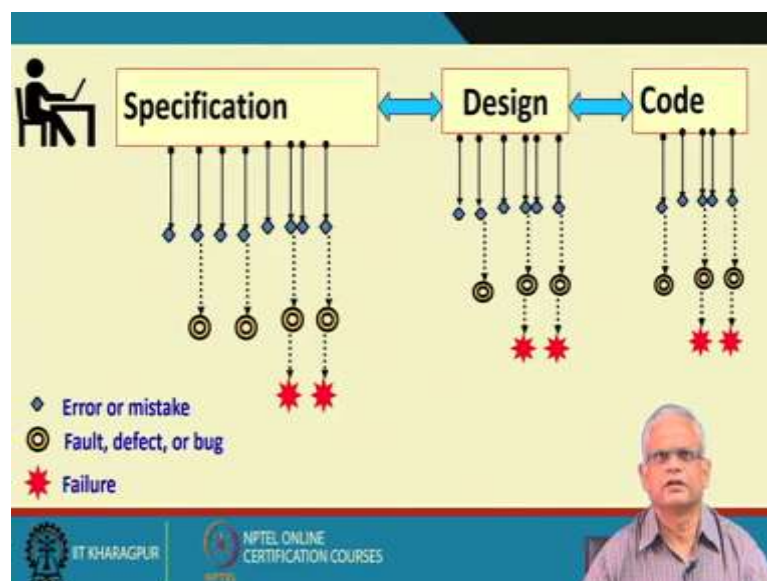
- Programming is human effort-intensive: **Errors, Faults, Failures**
 - Therefore, inherently error prone.
- IEEE std 1044, 1993 defined errors and faults as synonyms :
- **IEEE Revision of std 1044 in 2010 introduced finer distinctions:**
 - For more expressive communications distinguished between Errors and Faults



IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Let us distinguish between error, fault and failure. Programs, even though now a lot of automation automated tools are available, still programming is largely effort intensive. In 1993, IEEE standard 1044 errors and faults were considered to be synonyms. But, in 2010 revision finer distinctions were proposed, because there was a need to be more expressive in communications, we need to distinguish between error and fault because coding, designing etcetera are laborious manual effort and people do mistakes. And, therefore, we must know what is a error, mistake, fault, failure because these are the terms that are frequently used in this area.

(Refer Slide Time: 14:51)



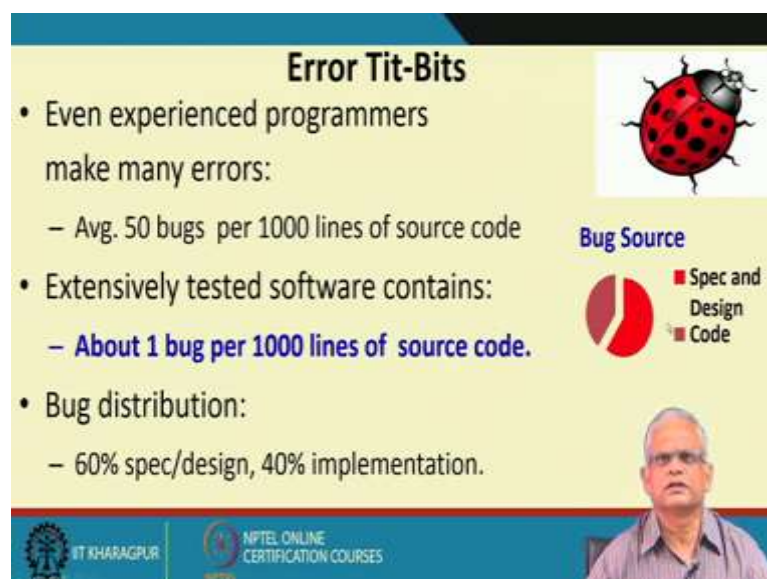
To understand we will just see this picture here the programmer here is typing using his laptop or desktop and initially he does the specification, design and then code. During the specification may make several mistakes. The mistakes we represented using this blue rectangles, but many mistakes they do not create problem, they are not bugs. Some of them are bugs, but then all bugs do not create failure, some bugs create failure.

Similar is the design and the code. The code the programmer may make mistakes or errors, but all errors are not faults. Let us say the programmer made a mistake instead of writing a is equal to b made a mistake wrote a equal to c, but then it so happened that b and c had the same value and therefore, it is a not a bug, it is not a fault. But, it was a mistake on the part of the programmer, he should have written a equal to c, but he wrote a equal to b.

But, then some of the faults may cause failure. There may be a fault let us say erroneously wrote that on some condition b is equal to 2 into b, but then let us assume that condition never occurs for the normal inputs. Instead of writing b into 3 written b b is equal to b into 2. There was a mistake committed and that is a fault, but then that does not cause failure because those inputs are not given during the actual uses.

Anticipating that such conditions may occur the programmer had given, but that condition never occurs and therefore, that bug does not express does not manifest as a failure.

(Refer Slide Time: 17:37)



Error Tit-Bits

- Even experienced programmers make many errors:
 - Avg. 50 bugs per 1000 lines of source code
- Extensively tested software contains:
 - **About 1 bug per 1000 lines of source code.**
- Bug distribution:
 - 60% spec/design, 40% implementation.

Bug Source

Source	Percentage
Spec and Design	60%
Code	40%

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

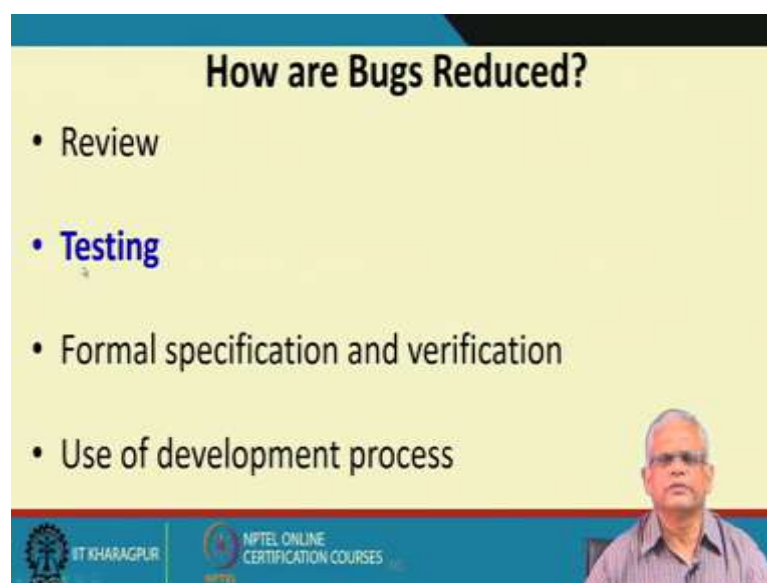
All programmers however proficient make mistakes and many of these mistakes they are basically bugs remain in the code and even with the most experienced programmers the data collected shows that average there are 50 bugs per 1000 lines of source code, is a large number. Even after testing thoroughly by professional testers, we do not really eliminate all the bugs.

Out of the 50 bugs that are there after the programmer completed programming by the time the testing is over most of the bugs are detected, but still 1 bug per 1000 lines of source code is typically present, is a study saying that even after though testing, 1 bug per 1000 line of source code is still there.

For some applications this may be acceptable, but let us consider a banking application where there is a bug and when the bug gets triggered into failure, there may be significant financial loss or may be a nuclear power plant, 1 bug per 1000 line of source code is not acceptable because those are safety critical systems, can cause huge damage if there is a failure.

But, what is the distribution of the bug? Typically majority of the bug that remain after development pertain to the specification and design, 40 percent of the bugs are based on the coding mistakes. Specification and design contribute the larger percentage of bugs about 60 percent; 40 percent bugs are present due to coding mistakes.

(Refer Slide Time: 20:13)



How are Bugs Reduced?

- Review
- **Testing**
- Formal specification and verification
- Use of development process

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, given that program development is a manual activity, bugs are inevitable because people do mistakes after all, how does one go about reducing the bugs to the minimum? One technique is review; review the work at various stages that is very well accepted technique to reduce the bugs. Testing – this is possibly the most important technique, most effective technique to reduce the number of bugs.

Formal specification and verification this can also be used to reduce the bugs. And, even use of a proper development process, developing good quality software then also reduce the number of bugs. So, as we can see that there are various ways in which we can remove the bugs and testing remains as one of the most promising ways of reducing the bugs.

(Refer Slide Time: 21:27)

How to Test?

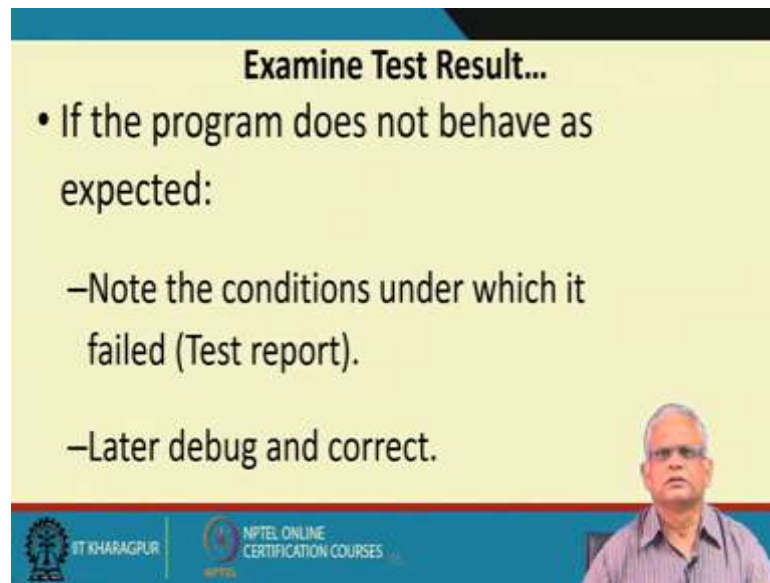
- Input test data to the program.
- Observe the output:
 - Check if the program behaved as expected.

The slide features a diagram illustrating the testing process. On the left, a yellow stick figure sits at a computer workstation. An arrow labeled 'Input' points from the figure to a computer system. From the computer, an arrow labeled 'Output' points to a magnifying glass held by another yellow stick figure, symbolizing inspection. The word 'System' is written vertically in the background of the diagram. At the bottom right of the slide, there is a small inset video of a man speaking.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

But, given a software how does one go about testing? To test any software we need to give it input data and then observe the output, and see if the generated output matches with the expectation we have for the given input. That is, observe the output and check if the program behaved as expected; is the system software to be tested we give input and observe if the output is as we expected otherwise we say that there is a failure.

(Refer Slide Time: 22:21)



Examine Test Result...

- If the program does not behave as expected:
 - Note the conditions under which it failed (Test report).
 - Later debug and correct.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The slide features a yellow background with a blue header and footer. A small video inset of a man with glasses is in the bottom right corner.

If the program does not behave as expected, we find what are the test data we gave and under what condition we gave the test data that is this software was at what state when we gave the test data. And, this we note it down the condition and the test data and this is called as the test report which contains all the conditions under which different data were given, the output produced and whether it was a failure or not.

And, these are taken up later by the developers. They reproduce the failure, debug and then debug and then correct the software.

(Refer Slide Time: 23:19)



Testing Facts

- Consumes the largest effort among all development activities:
 - Largest manpower among all roles
 - Implies more job opportunities
- About 50% development effort
 - But 10% of development time?
 - How?

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The slide features a yellow background with a blue header and footer. A small video inset of a man with glasses is in the bottom right corner.

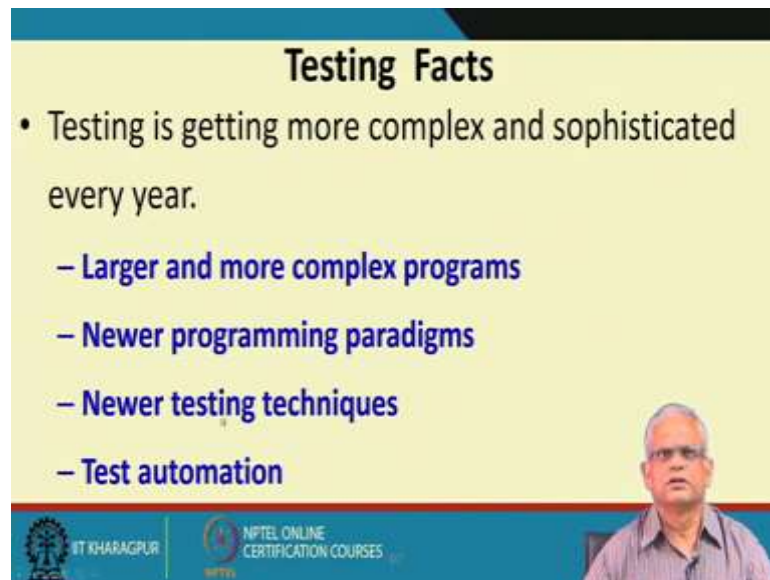
Out of all development activities, testing consumes the largest effort. Largest man power among all roles, companies invest heavily on testing, give much more effort to testing than design, coding or specification. And, of course, that means, that if you walk into a company you are more likely to meet a tester because number of testers are much more than other roles, which also implies that more job opportunities in testing because the companies need large number of testers.

For a typical software, 50 percent of the development effort is spent on testing, but then if we look at the time for which testing is carried out it is about 10 percent of the development time. If it is a 1 month project, then may be hardly 3 – 4 days are given for testing, the rest are for specification, design, coding and so on.

But, then 50 percent of the development effort is spent in 10 percent of the time how is it possible? It is possible because the testers can work parallelly there is lot of parallelism in the testing activities. We can deploy a large number of testers to carry out testing at the same time, they test the different functionalities, different test data and so on.

On the other hand, in design we can have only few designers designing it. We cannot really put large number of designers hundreds of designers designing it. There is typically a couple of designers. Similarly, coding we cannot deploy thousands of coders. We need to assign at least some module to each developer, each coder and therefore, the parallelism in other development activities are less and that is why they take more time even though less effort is given. During testing significantly larger effort is done in a shorter time.

(Refer Slide Time: 25:59)



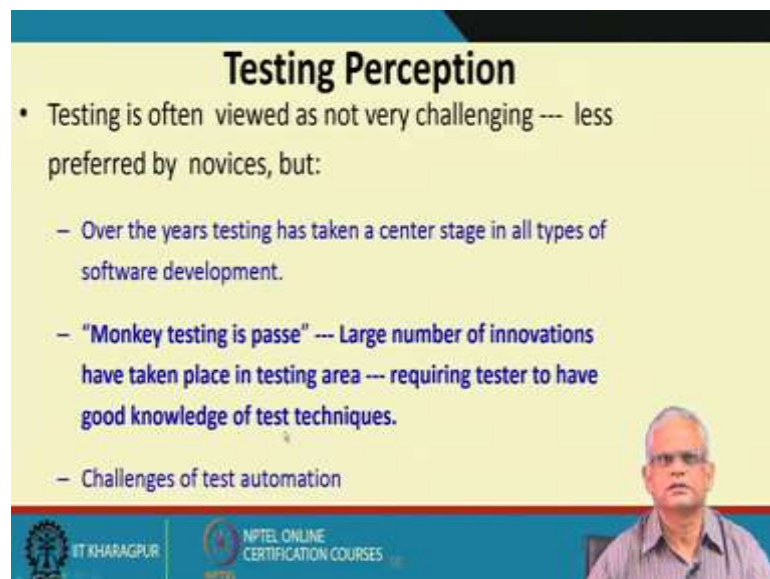
Testing Facts

- Testing is getting more complex and sophisticated every year.
 - Larger and more complex programs
 - Newer programming paradigms
 - Newer testing techniques
 - Test automation

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The testing techniques are evolving very rapidly that becoming more complex and sophisticated. The main reasons are larger and more complex programs; newer programming paradigms; test automation and for this and also new testing results. So, these all contribute to making testing a very challenging work.

(Refer Slide Time: 26:27)



Testing Perception

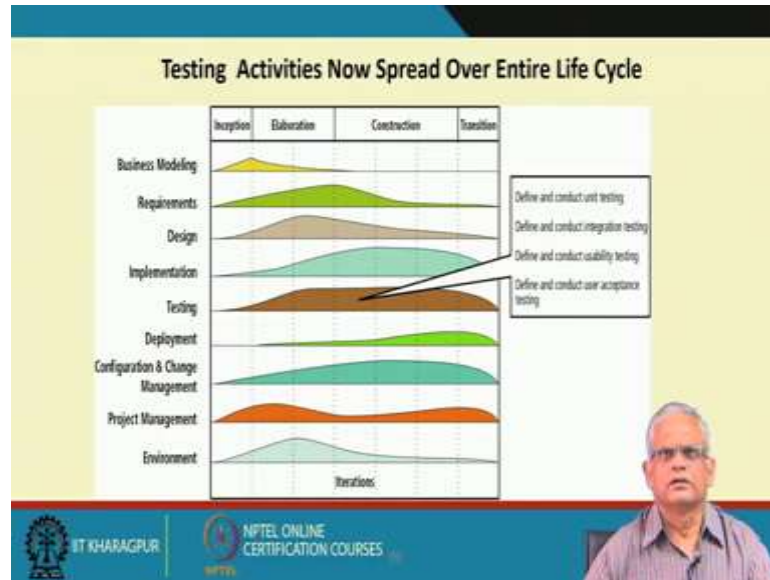
- Testing is often viewed as not very challenging --- less preferred by novices, but:
 - Over the years testing has taken a center stage in all types of software development.
 - "Monkey testing is passe" --- Large number of innovations have taken place in testing area --- requiring tester to have good knowledge of test techniques.
 - Challenges of test automation

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Testing is a actually very challenging even though the old time view was that testing is only a routine work, but now that has changed because testing has become very challenging. Earlier about 50 years back the testing was largely monkey testing where

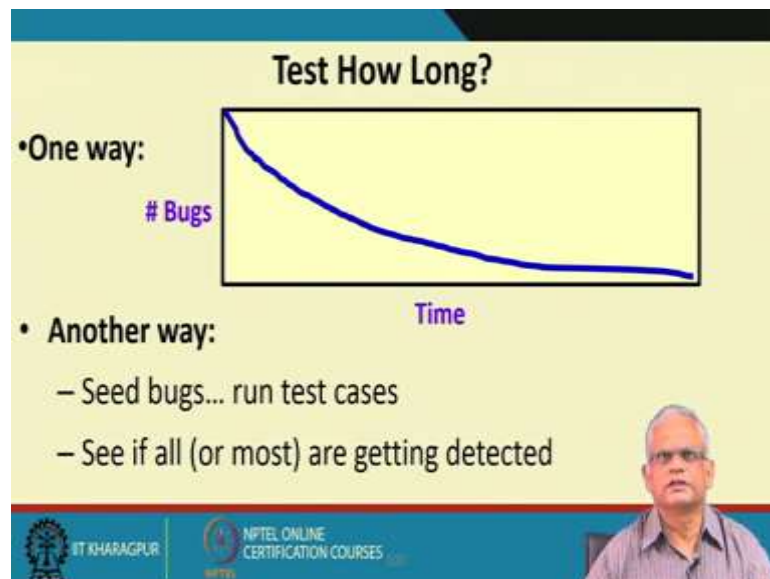
only the random values were given, but now large number of innovations have taken place. Many testing techniques testing tools etcetera and the tester needs a good knowledge of these techniques.

(Refer Slide Time: 27:09)



In reality, how long does a testing go on? As we can see on the unified process representation here the testing is actually carried out over the life cycle. At different time of the life cycle the test cases are defined, unit testing is conducted, integration testing defined, conducted, usability testing, system testing is defined and conducted.

(Refer Slide Time: 27:47)



But, then one important question is to test how long, because as more and more test data defined bugs get detected. One way is that we observe the number of bugs that are getting detected over unit time and as testing progresses, we find that the number of bugs detected per day is decreases and then we have a situation where testing takes place for a day or a week and no bug is detected and that is the time when we may stop testing.

The other way is to seed bugs. The manger introduces bugs into the program without telling the developers and testers and then observes how many of the bugs that he had introduced are getting detected. If he finds that most of the bugs have got detected, then the manger knows that it is the time to stop testing.

(Refer Slide Time: 29:11)

Verification versus Validation

- Verification is the process of determining:
 - Whether output Of one phase of development conforms to its previous phase.
- Validation is the process of determining:
 - Whether a fully developed system conforms to its SRS document.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Another terminology that manger needs to be aware is the difference between verification and validation. In verification, we determine when whether output of one phase of development confirms to the previous phase. For example, if the design results are consistent with the specification whether all specification have been taken care properly.

Similarly, for coding whether all the design modules that were designed have been taken care properly and that is called as the verification whereas, validation is the process of determining whether a fully developed system confirms to its specification. So, that means, the validation is carried out on the completed system whereas, verification are done as the development proceeds to check whether the development is preceding

correctly, that is, one activity produces result that are consistent with the output produced by the previous activity. We are running out of time for this lecture and we will stop here.

Thank you.