

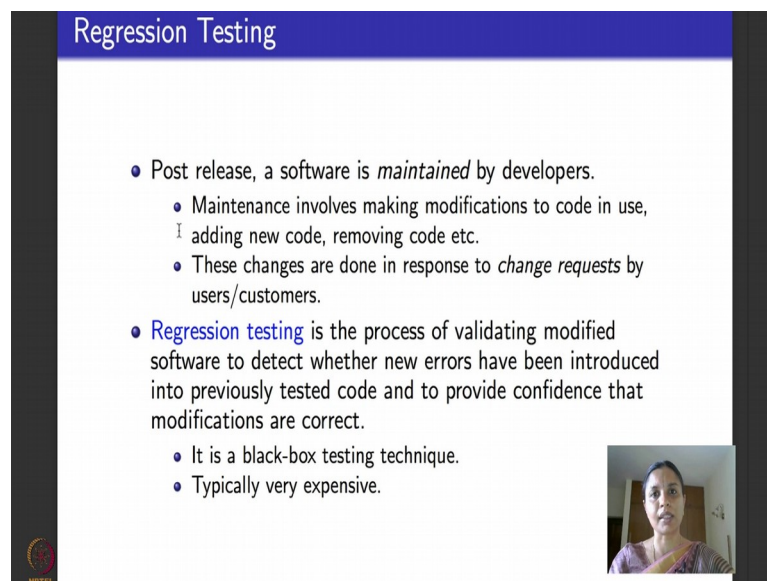
Software testing
Prof. Meenakshi D'Souza
Department of Computer Science and Engineering
International Institute of Information Technology, Bangalore

Lecture – 58
Regression Testing

Hello again, this is the next module for week 12. I am going to do regression testing today. So, this is again one of the modules that came as request from the takers of the course. So, what is an overview of today's lecture, I will introduce you to what is regression testing, give an overview of it and then one of the biggest problems with regression testing is how to select a set of test cases that is apt for regression testing. So, what is apt depends on what are you regression testing, what is the problem that you are trying to solve and what will be an ideal set of test cases that I can use for regression testing. That problem is the problem of regression test selection.

There are several known techniques for regression test selection because this is one of the important problems. So, we will spend a while understanding an overview. I will give you a survey of 5 different techniques for regression test selection and I will end as we did for the other 2 modules by giving you an idea of some of the tools that I am familiar with to do regression testing.

(Refer Slide Time: 01:21)



Regression Testing

- Post release, a software is *maintained* by developers.
 - Maintenance involves making modifications to code in use, adding new code, removing code etc.
 - These changes are done in response to *change requests* by users/customers.
- **Regression testing** is the process of validating modified software to detect whether new errors have been introduced into previously tested code and to provide confidence that modifications are correct.
 - It is a black-box testing technique.
 - Typically very expensive.

So, what is regression testing? Throughout in this course, we have seen testing as it applies in the software development life cycle. That is while the software is being developed, we do testing, unit testing, integration testing at the software level, at the hardware level system testing acceptance testing and so on. Regression testing comes after all this, this comes after the software is released and is being maintained. Usually it is very rare that a piece of software is released and forgotten about. People release it and also maintain a software. Maintenance of the software involves periodic upgrades that are done to a software, upgrades because some user customer wanted a change in the software, in some functionality of a software or upgrades or changes because there was an error found in the software after release.

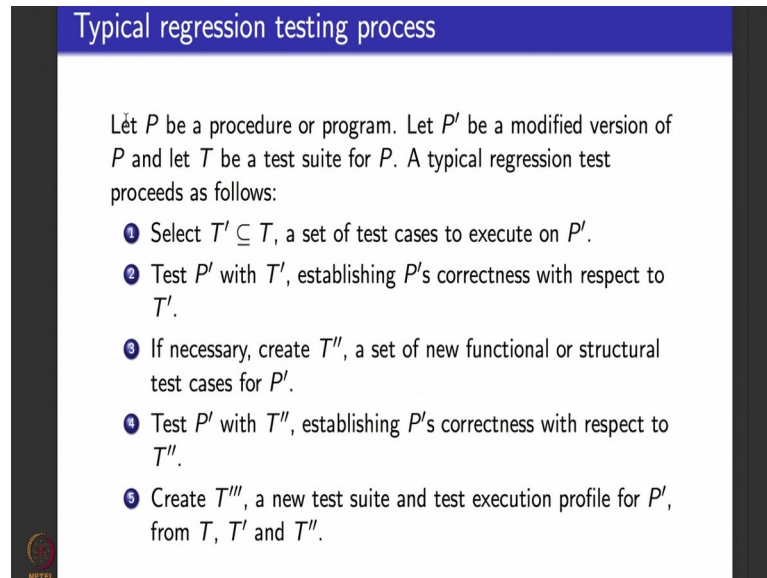
Lot of companies usually develop software that lasts for many years. In fact, in some kind of fields like aerospace and other safety critical system, software is developed to run for many decades, that is 10s of years, 20 years, 30 years, 40 years. Same holds for traditional things like banking software; you know software that was written in 70s and 80s using a programming language like COBOL is still being maintained by several IT companies. How do they do it? They have a team that maintains the software that takes requests for changes, observes errors, writes new code in the language that the software supports and along with writing new code, the team also has to test whether the new code is working fine or not. Such a test that is done by a maintenance team that maintains a software is what is called regression testing.

So, regression testing is a process of validating or testing the modified software. It usually focuses on the piece of software that is modified around the piece of software. A whole software along with the modified component is rarely tested all over again. Obviously, the whole software is put together with a modified software, but the testing focuses on the modifications done and not on repeating the entire set of the tests of the old software was subject to.

So, regression testing is the process of validating or testing the modified software. The goal is to be able to detect whether new errors have been introduced, one, in the modified software and because of the modified software have new errors been introduced in previously tested and released code. Regression testing is also meant to provide confidence that the modifications done are correct. It is typically considered a black box testing technique and is typically very expensive. Why is it very expensive? It is

expensive because people have no idea or clarity about how to select a set of test cases to do regression test.

(Refer Slide Time: 04:52)



Typical regression testing process

Let P be a procedure or program. Let P' be a modified version of P and let T be a test suite for P . A typical regression test proceeds as follows:

- 1 Select $T' \subseteq T$, a set of test cases to execute on P' .
- 2 Test P' with T' , establishing P' 's correctness with respect to T' .
- 3 If necessary, create T'' , a set of new functional or structural test cases for P' .
- 4 Test P' with T'' , establishing P' 's correctness with respect to T'' .
- 5 Create T''' , a new test suite and test execution profile for P' , from T , T' and T'' .

We will spend some time understanding it. So, is it clear please? What regression testing is? A software is being maintained and the process of maintaining a software people make periodic changes to the code. The change that is made to the code is tested to see if it has no errors, if the changed code does not result in any errors in existing software and to gain confidence in the changed modification. Typically what is the process for regression testing. Let us consider P to be a procedure or a program and let P' be a modified version of P the changed version of P and let T be a test suite for P . P was a program that was developed tested and released being maintained why being maintained, it was modified to obtain P' . Now we already have a test suite that was tested for P and documented and stored aside for use during modifications, if any happens.

So, regression tests process proceeds for P' as follows. So, from the set T which is a set of test cases for P , we select the subset T' which is a subset of T which is a set of test cases to execute on the modified version of P . This is a very difficult process. We will spend some time on how to select this T' . Once you have selected T' , what we do is we test the modified P' with the T' establishing P' 's correctness with reference to T' . If there is an error of course, it goes through the natural course. Sometimes in regression testing we also create a new set of functional or structural test cases for the modified version P' . In

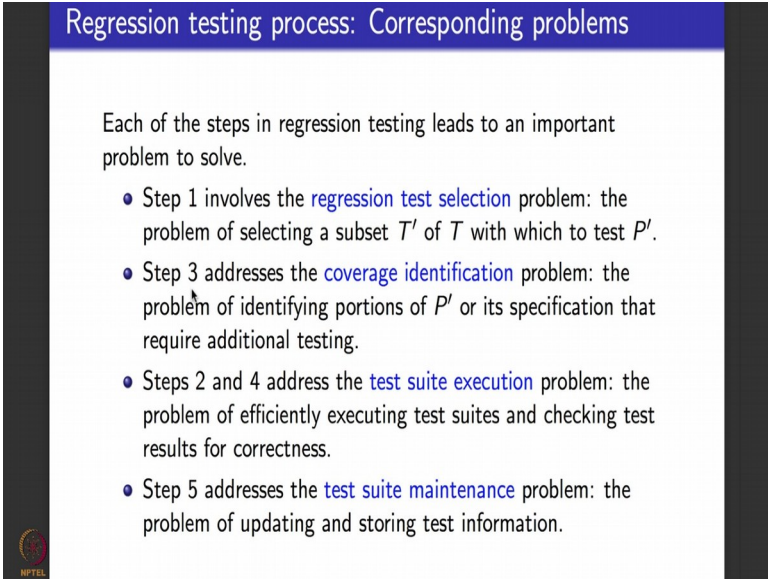
addition to T' we create a new set T'' which is exclusively meant to tests the modified version P' . Then what, once we create it, we test P' with T'' showing that P' is correct with reference to the set of test cases T'' .

Then for now P is no longer exists P' is the one that modified version of P which is going to be integrated into the software and kept now. Later P' might be modified, a portion of P' might be modified. So, you might want to document and record a set of test cases to be used when P' is modified that is the last step. So, a new set of test cases T''' is created from T' and T'' and kept as a record for, as a test suite in a test execution profile to be used for P' .

So, just took succinctly repeat what I said we have a procedure or a program P that is modified to P' , there is a set of available test cases T for P' is tested with. The sub set from P' is tested and P' is also tested with the new set of test cases T'' and P' is tested with T''' which is kept which is derived from T and T'' and T' and kept as a record for the test cases to be used in case P' is modified. So, there are 5 steps that we understood here each step deals with different problem when it comes to regression testing.

So, step one which is this selecting T' which is the sub set of T is called the regression test selection problem. It is a problem of selecting a sub set T' of T with which P' is tested. Step 3, sorry step 3 which is T'' which is a new set of functional test cases is done because sometimes T' may not cover P' .

(Refer Slide Time: 08:03)



Regression testing process: Corresponding problems

Each of the steps in regression testing leads to an important problem to solve.

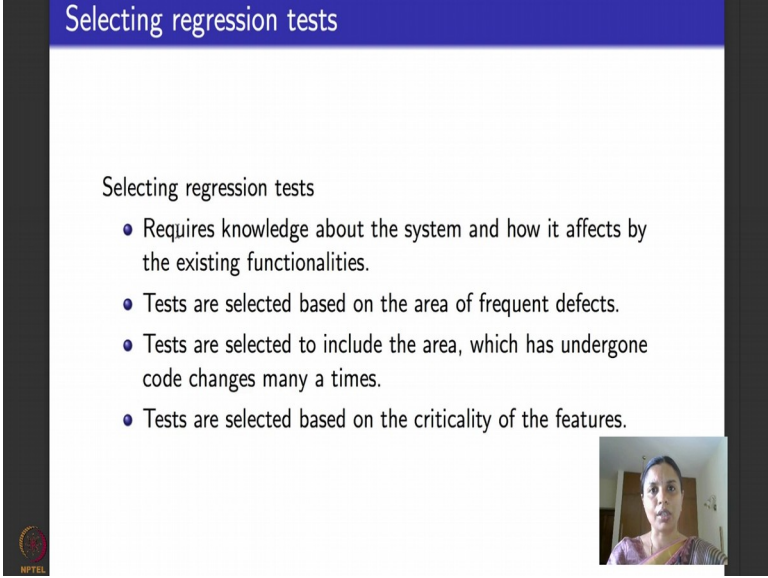
- Step 1 involves the **regression test selection** problem: the problem of selecting a subset T' of T with which to test P' .
- Step 3 addresses the **coverage identification** problem: the problem of identifying portions of P' or its specification that require additional testing.
- Steps 2 and 4 address the **test suite execution** problem: the problem of efficiently executing test suites and checking test results for correctness.
- Step 5 addresses the **test suite maintenance** problem: the problem of updating and storing test information.

So, step 3 addresses the coverage identification problem that is the problem of identifying portions of P' or the specification of P' that require additional testing steps 2 and 4 basically are execution, test case execution. So, it is the set of test suite are selected they are executed.

So, the steps 2 and 4 talk about the test suite execution problem, that is the problem of efficiently executing test suites checking the results for correction. Step 2 and 4 can be fairly largely automated like in other cases of testing because it is just test case execution, does not need much of algorithmic knowledge, but needs extensive tool support. Step 5 is the part where a set of test cases are kept aside to be used for P' if required later. So, step 5 addresses the test suite maintenance problem that is the problem of updating and storing test information. What we will be spending in this lecture is I will tell you some techniques for regression test selection problem which is step 1 and I will also tell you some tools that help you in test suite execution which is regression testing automation. Before we move on just a small point about regression and specifications. Regression test selection is applicable both in cases where specifications have changed and specifications have not changed. When do I do regression testing with specifications have not changed? Maybe I found an error in the software that is released. So, I can develop a batch that rectifies the error and regression testing is done for that.

If this is the case, then it means the specifications have not changed I am doing regression testing because I found an error. So, in the case where the specifications have changed, we have to identify test cases that are obsolete for P' , prior to performing test case selection. So, how, what do you mean by test cases that are obsolete? They are obsolete for P' if the test case specifies an input to P' that is invalid for P' , that means the format of the input, the syntax of the input the type of the input does not match P' . P' cannot even be executed on that input or the test case specifies an invalid output, input output relation for P' . So, once we identify these test cases that are obsolete, our goal is to remove them from T and then do regression test selection on the remaining test cases. So, test selection process, we do not worry about identifying obsolete test cases.

(Refer Slide Time: 10:42)



The slide has a blue header with the title 'Selecting regression tests'. Below the header, the text 'Selecting regression tests' is repeated. A bulleted list follows, containing four points. In the bottom right corner, there is a small video inset showing a woman with dark hair and glasses, wearing a purple top, speaking.

Selecting regression tests

Selecting regression tests

- Requires knowledge about the system and how it affects by the existing functionalities.
- Tests are selected based on the area of frequent defects.
- Tests are selected to include the area, which has undergone code changes many a times.
- Tests are selected based on the criticality of the features.

So, now coming to regression tests selection problem, here is an overview of what happens while selecting regression tests. Regression tests selection requires, obviously, knowledge about the system which could mean domain knowledge and how the modified code affects the existing functionalities without which we cannot effectively tests the code. Tests are selected based on the area of frequent effects. Where do the defects happen? The Pareto principle which is 80% of the defects occur in 20% of the software applies to regression testing also and tests are also selected to include the area which is undergone code changed many times because that is where the problems if any are likely to occur and tests are also selected based on the criticality of features. Let us say you changed because you wanted to cater to a particular feature which is highly critical then you focus on regression testing for that feature, you could do any of these.

(Refer Slide Time: 11:38)

Minimization techniques

- Minimization test selection techniques for regression testing attempt to select minimal sets of test cases from T that yield coverage of modified or affected portions of P .
- The problem, in its generality, is undecidable.
- 0-1 integer programming based techniques are available for procedures that are basically a sequence of statements with single entry and single exit (no branching, looping etc.).



So, we will walk through a set of 5 different techniques for regression test selection. Towards the end of this lecture, I point you to a reference to a survey paper which is slightly old, but still very relevant in terms of its use which talks about all these test selection techniques that we have learnt and papers that I will be referring to, all of them are referred to or cited in the survey. So, the first set of techniques are called minimization techniques. What do they do? They attempt to select minimal set of test cases from the set T that yield the coverage or modified, effected portions of P . Minimal means minimal in number, so the number of test cases is as less as possible.



So, the sub set T' that we choose from T is as minimal as possible usually this problem is un-decidable in its most generic case. I do not know what is the minimal set of test cases, but for these kind of programs or procedure that are basically a one block of statement, a sequence of statements with the single entry and a single exit meaning, there is no branching, no looping, etcetera, there are some solutions based on 0 1 integer programming techniques that do minimization test case selection for regression testing. As I told you the survey paper that I have, I will be referring to towards the end of this lecture will point you to more reference if you are interested in knowing about it.

The next is data flow techniques. Data flow coverage based regression test selection, they select test cases that exercise definitions and use or data interactions that have been effected by the modified portion. Here is one paper slightly old, but again a very good paper.

(Refer Slide Time: 13:03)

Dataflow techniques

- Data-flow coverage based regression test selection techniques select test cases that exercise data interactions that have been affected by modifications.
- For example, the technique of Harrold and Soffa [1988] requires that every definition-use pair that is deleted from P , new in P' , or modified for P' be tested. The technique selects every test case in T that, when executed on P , exercised, deleted or modified definition-use pairs, or executed a statement containing a modified predicate.




It is technique by Harold and Soffa which requires that, what sort of data interaction is tested here every pair of definition and use that is deleted from P or is newly added in P' or is modified for P' , is tested. So, this technique is basically selects a test case T in T such that when it is executed on P , it is exercised, deleted or modified definition use pair or executed on a statement containing a modified predicate. So, it observes attracts values of data definition and uses that have changed from P to P' and develops a set of test cases that basically cater to these changes of definition and uses.

(Refer Slide Time: 14:05)

Safe techniques

- Techniques that are *not safe* can fail to select a test case that would have revealed a fault in the modified program.
- In contrast, when an explicit set of *safety conditions* can be satisfied, safe regression test selection techniques guarantee that the selected subset, T' , contains all test cases in the original test suite T that can reveal faults in P' .
- For example, the technique of Rothermel and Harrold [1997] uses CFG representations of P and P' , and test execution profiles gathered on P , to select every test case in T that, when executed on P , exercised at least one statement that has been deleted from P , or that, when executed on P' , will exercise at least one statement that is new or modified in P' (when P is executed, a statement that does not exist in P cannot be exercised).



The next set of regression test techniques are what are called safe techniques. So, we will understand what is safe technique and what is not safe regression testing. A set of regression test techniques are considered not safe, if they can fail to, they fail to detect the test case that would have revealed a fall fault in the modified test program. Obviously, this is something that we will not desired when we do regression test selection. We want it to be safe in the sense that if there is a fault that is present it is always detected by the set of test cases that I define. So, an explicit it set of safety conditions can be satisfied, a safe regression test selection technique guarantee that the subset T' that I chose contains all the faults that, I mean, contains a set of test cases that can reveal all the faults that are present in P' .



Again, it is quite difficult to, in general, guarantee that I will always be able to define a set of test cases that will detect the faults in P' , but we can try and do our best and this is the paper that does a good attempt. So, here is a paper that basically comes up with the control flow graph representations of P and P' and what it does is that it takes the test execution profiles of T being executed on P and then it says which was at least one statement that was exercised which was actually deleted from P or, the statement when executed on P' , will exercise one statement which is new in P' .

So, basically what it does is that it pulls out set of test cases which make sure that if executed the execution behavior of P' is different from that of P . So, these set of test cases exercise statements that were deleted from P , to get P' or set of statements that were modified in P to get P' and so on, if you do this, then the belief that you are guaranteed to obtain a safe set of regression test techniques which will find a fault in P' if it exists because basically what it tries to do is it tries to cover all the modifications that were done to P to obtain P' .

(Refer Slide Time: 16:15)

Random techniques

- In this case, a randomly selected set of test cases for P are executed on P' .
- This is often useless and done only to show that some regression testing is done.





The next 2 selection techniques have no great technical depth in them, but are the most used in industry ironically. So, what is this one do? This one basically is called a random test selection technique. It randomly chooses a subset the prime of T and tests P' from it. Obviously, because it randomly chosen, there is no bases no coverage and no guarantee that the faults if present in P' will be adopted, but it is a fairly standard practice in industry because there is no time to do regression testing. So, people just randomly select and do a set of test cases. The next is what is called retest all which is basically take all the test cases that were there in the original set T and test P' from it. This is obviously safe because it tests whole thing again.

(Refer Slide Time: 17:00)

Retest-All techniques

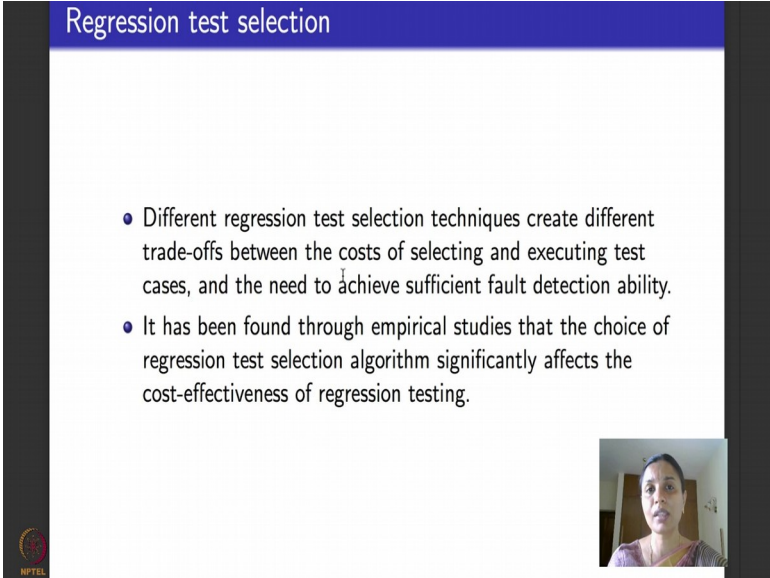
- The retest-all technique simply reuses all existing test cases.
- To test P' , the technique effectively *selects* all test cases in T .



But it may not be efficient because it can be very time consuming and you might have to write some new test cases that are specific to changes done in P' if you just blindly reviews the set of test cases that were using for P, it may or may not be effective for testing P'.

So, we learnt 5 techniques to summarize: minimization which selects minimal set of test cases for regression testing, data flow techniques which select a set of test cases that basically ensure that for every variable that is changed in P' its definition and use attract safe techniques which basically ensure that faults if present in P' are revealed because it executes, selects a set of test cases which either modified or delete a statement in P to obtain P', random which is basically a random selection of test cases, T' to execute on P', and retest all it is execute every test case on P that was present in P'. So, these are 5 techniques that are known for test case selection I will point you to a survey paper from which you can learn more about these techniques and also get data to the exact papers that we referred to while giving an overview of these techniques.

(Refer Slide Time: 18:41)



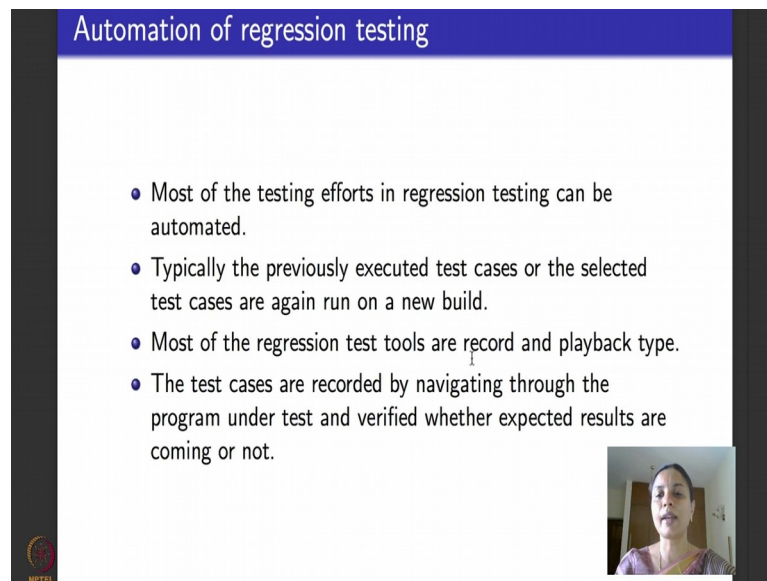
Regression test selection

- Different regression test selection techniques create different trade-offs between the costs of selecting and executing test cases, and the need to achieve sufficient fault detection ability.
- It has been found through empirical studies that the choice of regression test selection algorithm significantly affects the cost-effectiveness of regression testing.

Moving on, now different regression test selection techniques can create trade-offs between the costs of selecting and executing test cases. Obviously, because based on what your priority is you need to select test cases accordingly, it is been found through empirical studies that the choice of the test selection algorithm significantly affects the cost total cost of the regression testing. So, what is, when it comes to automation of

regression testing, like all other testing, this is heavily automated, cannot be manually done on always, especially the execution part of regression testing. So, typically the previously executed test cases or the newly selected test cases are again run on a new build that is created for regression testing. They are basically record and playback type of tools.

(Refer Slide Time: 19:10)



Automation of regression testing

- Most of the testing efforts in regression testing can be automated.
- Typically the previously executed test cases or the selected test cases are again run on a new build.
- Most of the regression test tools are record and playback type.
- The test cases are recorded by navigating through the program under test and verified whether expected results are coming or not.



Which means what happens there? Test cases are recorded by navigating to the program under test and they are verify whether the expected results are coming or not.

So, here are some of the regression testing tools that I have seen. The first one, none of them are open source. All of them are proprietary. The first one is called regression tester that is available from this website. It is pretty decent tool that exclusively focuses on this recording and playback type of execution. Then there is one more TimeShiftX which basically tries to if you typically when you record a test case, you will also time stamp it and record.

(Refer Slide Time: 19:39)

Some regression testing tools

- Regression Tester: Available from <http://www.regressiontester.com/>.
- TimeShiftX: Available from <https://www.vornexinc.com/>.
- Tools from Micro Focus: Available from <https://www.microfocus.com>.





So, this works by telling that you re-executive those state of test cases by shifting the time stamp and which it was recorded. The build is created just by basically manipulating that. The next is a set of tools from micro focus, there are 2-3 tools that do regression testing of some very old code like COBOL programs and all that stuff you can refer to that tool from this website, Micro Focus. Please remember that none of them are open source, I am not aware of very good open source regression testing tools, all of them are proprietary.

(Refer Slide Time: 20:29)

Some references

- Tools: Links already provided, Last access October 2017.
- A nice survey paper on regression test selection:
T. L. Graves, M. J. Harrold, J-M. Kim, A. Porter and G. Rothermel, An Empirical Study of Regression Test Selection Techniques, in ACM Transactions on Software Engineering and Methodology, 30(2), April 2001.



So, to end with here; here is the reference that I told you it is a 2001 survey paper which talks about an empirical study of regression test selection techniques. So, it focuses on

the 5 techniques that we learnt about and tells you how to trade off and how to choose the best test selection problem and some of tools I have already given the link provided please note that these URLs were valid as of this presentation. So, that gives us an overview of regression testing. So, and it pretty much brings us to the end of this course, feel free to ping me in the forum, if you have any doubts for the exam. It is a pleasure interacting with all of you.

Thank you.