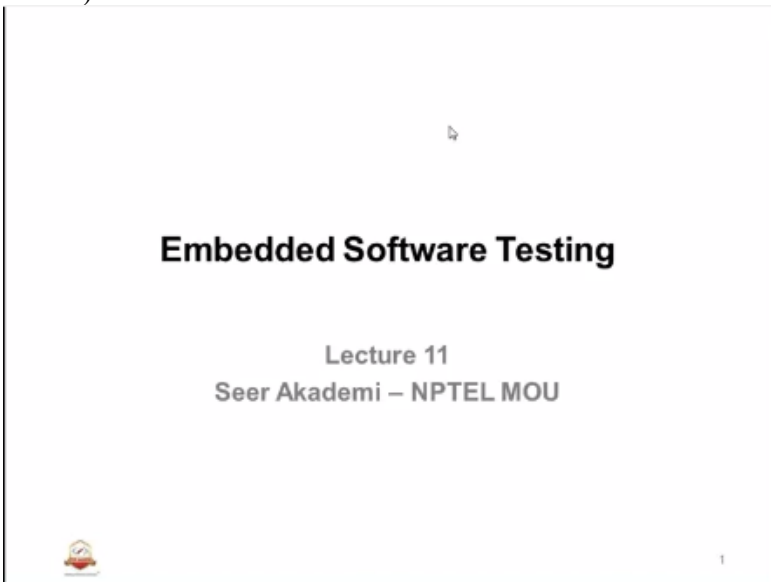
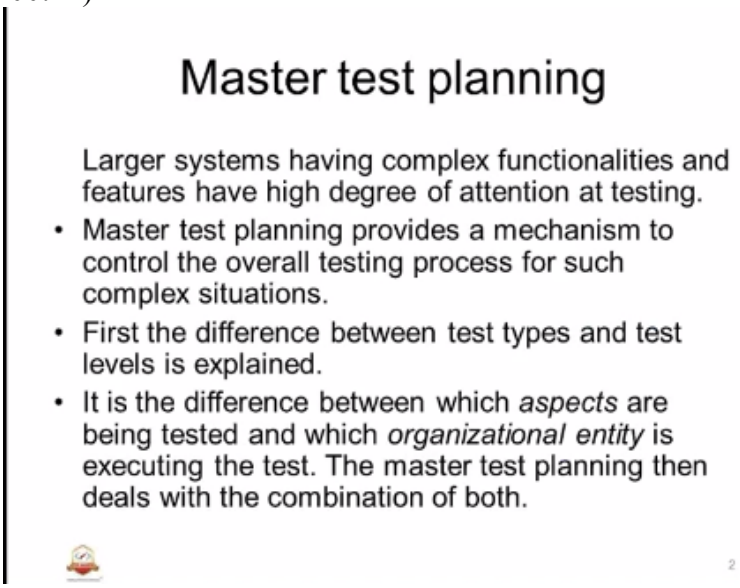
Welcome you to the next session on embedded software testing,
(Refer Slide Time: 00:08)



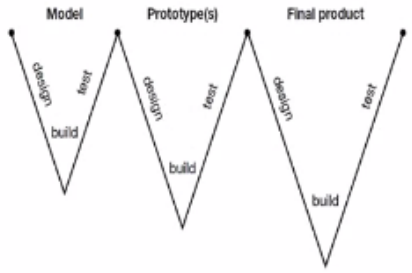
This is the last session of this embedded software testing, here we will go through the master test planning and,
(Refer Slide Time: 00:22)



In details and in the end we will analyze the unit 1 details what we have studied so far. And represented we have understood that different modules multiple we started with life cycle modules, prototype and formalin cycles and we studied pre model having decor ant design implementation of the left hand side and on the right hand side. We have unit testing integration and system testing.
(Refer Slide Time: 01:13)

Multiple V-Model Life cycle

Figure 3.1
Multiple V development lifecycle



3

And we also went through the multiple V model life cycle module prototypes final products, this one is considered for multiple cycles,
(Refer Slide Time: 01:28)

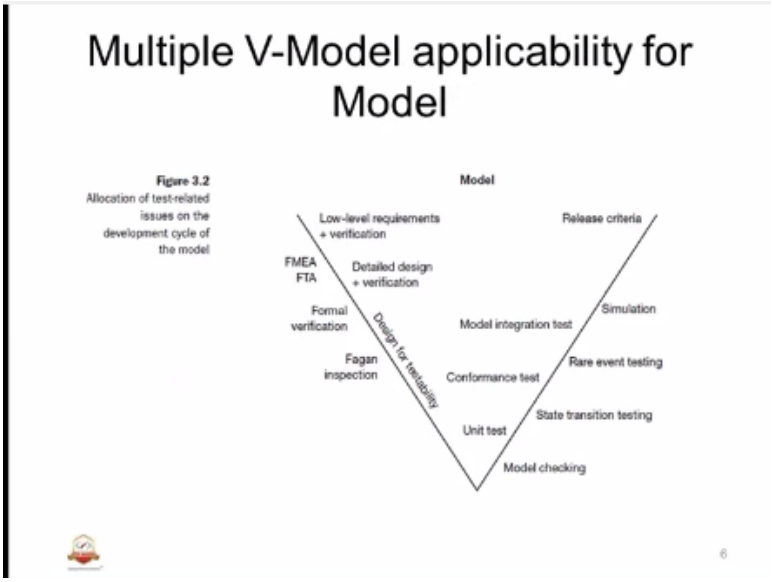
Multiple V-Model : Activities and issues that are to be considered

Table 3.1
Test related activities and issues that need to be allocated throughout the development and testing lifecycle

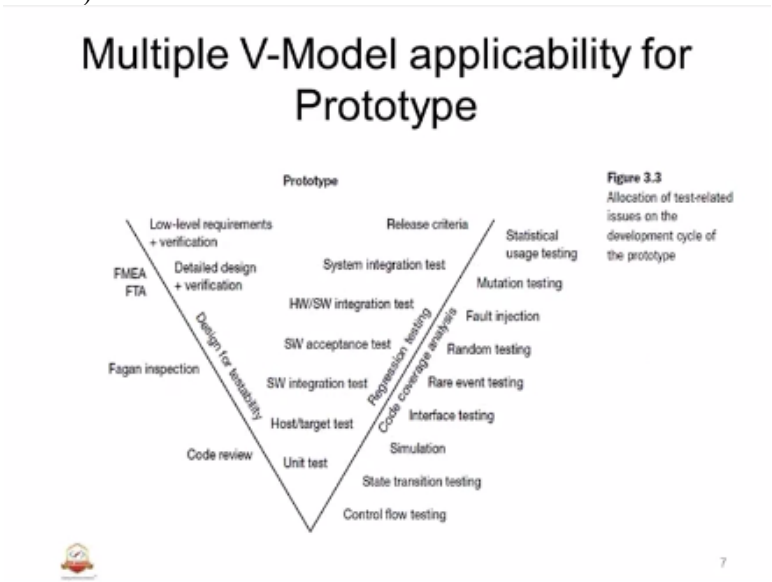
Techniques	Test levels, types	Other issues
Code coverage analysis	Architectural design verification	Architectural design
Control flow testing	Code review	Certification
Fagan inspection	Conformance test	Detailed design
Failure mode and effect analysis (FMEA)	Detailed design verification	Detailed test plan
Fault injection (FTA)	Hardware/software integration test	Design and build tools
Fault tree analysis	Host/target test	Design and build simulator
Formal verification	Model integration test	Design and build stubs
Interface testing	Real-life test, or field test	Design and build drivers
Model checking	Regression test	Design for testability
Mutation testing	Requirements verification	High-level requirements
Random testing	Software acceptance test	Legal requirements
Rare event testing	Software integration test	Low-level requirements
Simulation	System acceptance test	Master test plan
State transition testing	System integration test	Production requirements
Statistical usage testing	Unit test	Release criteria/advise
		Safety plan

5

So we also went through the activities that involved techniques, test levels, types, other issues. Comes in to the multiple V model,
(Refer Slide Time: 01:37)



We also told you the allocation of test plated issues on the development cycle,
(Refer Slide Time: 01:48)



On the prototype and on the final product how it is applicable for the multiple V model then we had,
(Refer Slide Time: 01:57)

Nested multiple V-model

- When the V-model at the system level is combined with the multiple V-model at the component level, it results in the so-called "nested multiple V-model" (Figure 3.6).

9

Gone through the nested multiple V - model which will be having multiple V model nested based on the components we had.

(Refer Slide Time: 02:09)

Nested Multiple V-Model contd.

With this model, all test-related activities and issues can be allocated to the correct level in the correct place. At the system level, higher-level test issues can be allocated to the overall development cycle, as shown in Figure 3.7.

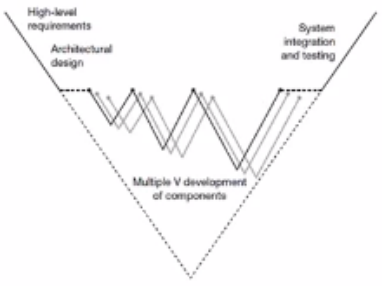
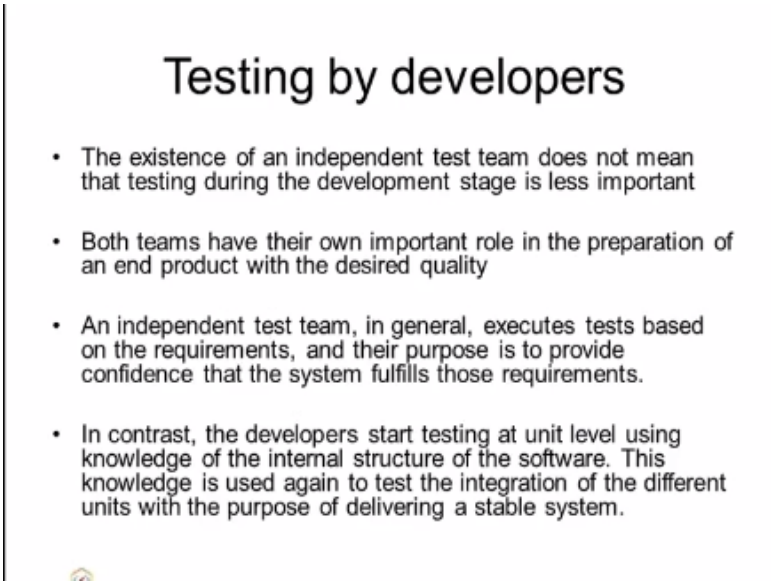


Figure 3.6
The nested multiple V-model

11

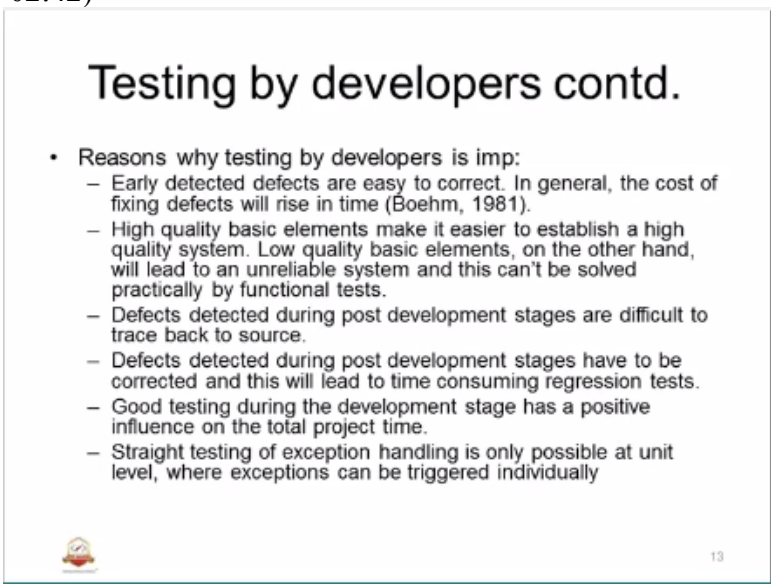
Usually we allow the systems and also we went through drop slides on testing bite developers that means the initial stage of the testing bite they can do it.

(Refer Slide Time: 02:26)



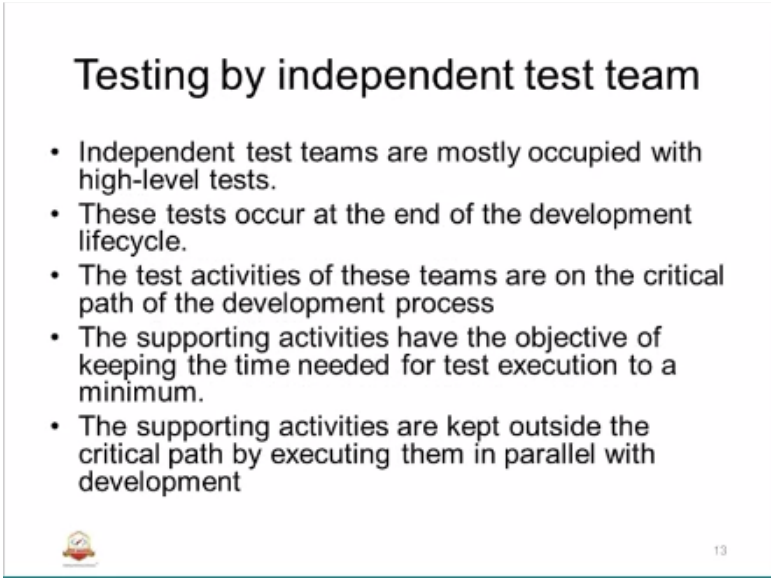
Since they have the necessary debugging and sprinkler testing environment what the integer for testing? So that they can fix errors and bugs,

(Refer Slide Time: 02:42)



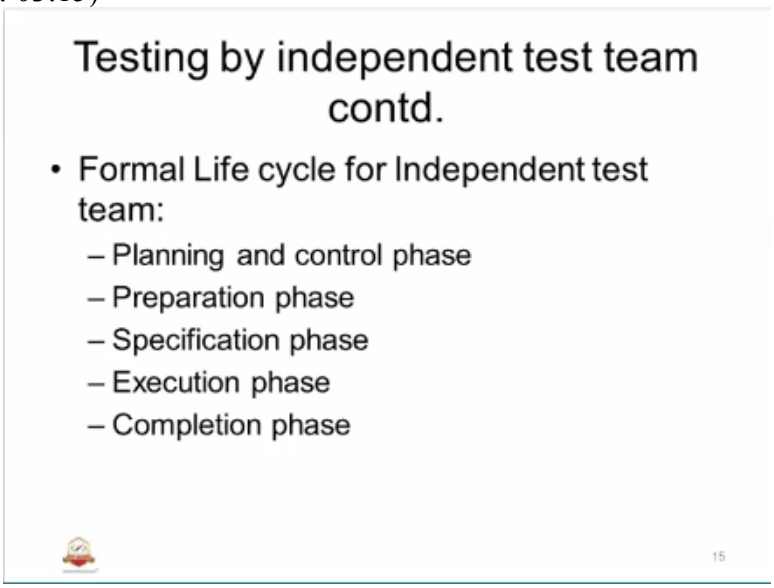
This is slide on why testing is important at the developer stage is in terms of finding the defects are leave and to gain the complants to coed with the next with level.

(Refer Slide Time: 03:00)



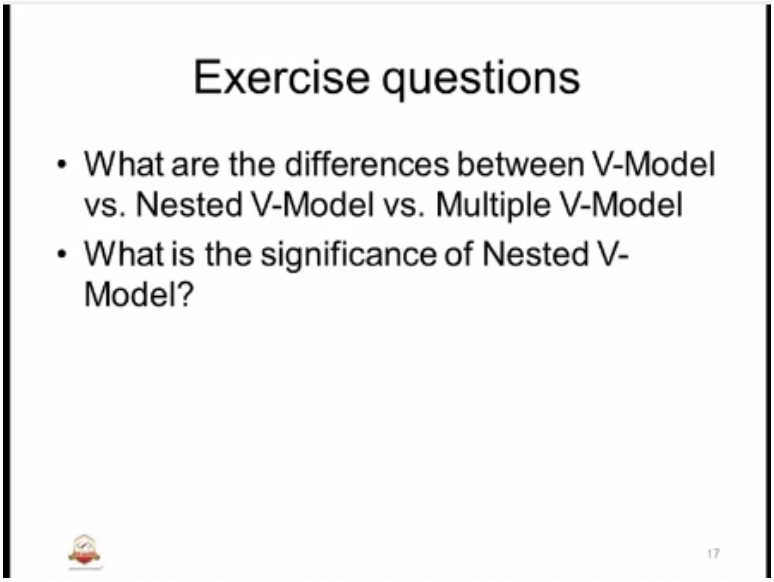
Then we had gone through slide on testing by independent test team, we know that all the test are driven by them including the test plan also activities so,

(Refer Slide Time: 03:15)



There are five stages of formal life cycle for independent testing. Planning and control phase, preparation phase, specification phase, execution phase, completion phase, we had a exercise.

(Refer Slide Time: 03:30)



On that, we study about master test planning what is master test planning? Typically testing of a larger system with many hardware components address of down server, code, lines of code it is also there, behave complex that means it is a complex behavior so involving many people performing different testing tasks various forms variables required, and especially complexity in terms of testing software stage or testing stage.

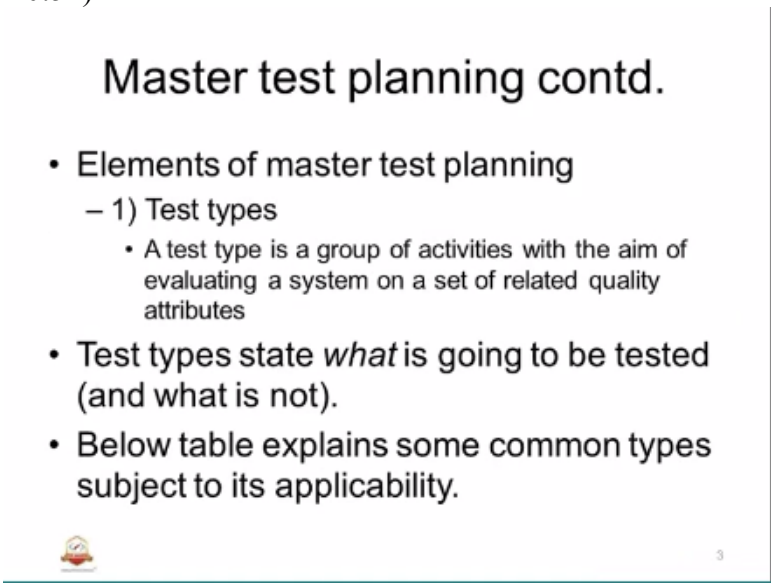
And in consistence certain components exhaust we need to test it, with the higher integrant that is based on the complexity the system is what? And the features all this have to be considered as an individual testing. So what we do is we will independent teams, so you remember here testing is done by a larger team so larger team means we can further sub divide that larger team into sub teams and some team they can focus dedicated in the performance requirements, specialized features etc can be segregated for the entire software testing.

For these complex situations master test planning is very much necessary so what it does is it formulates the mechanism to control the overall testing, so has been master test planning as some to pick up it basically requires the complex in terms requires good plan. So that is why we need to have a master test planning. So there is a term master test planning so larger system complex personalities and features that are higher degree of testing.

Master test planning provides and controls the overall testing process in the complex systems, what you tell first test because between the test types it plays, so what are the test types? What are the test levels that we see in our files? So basically it will defers between these two because that is very much important identifying the key elements of the plan so that is why passed in to a test planning is conducted based on this.

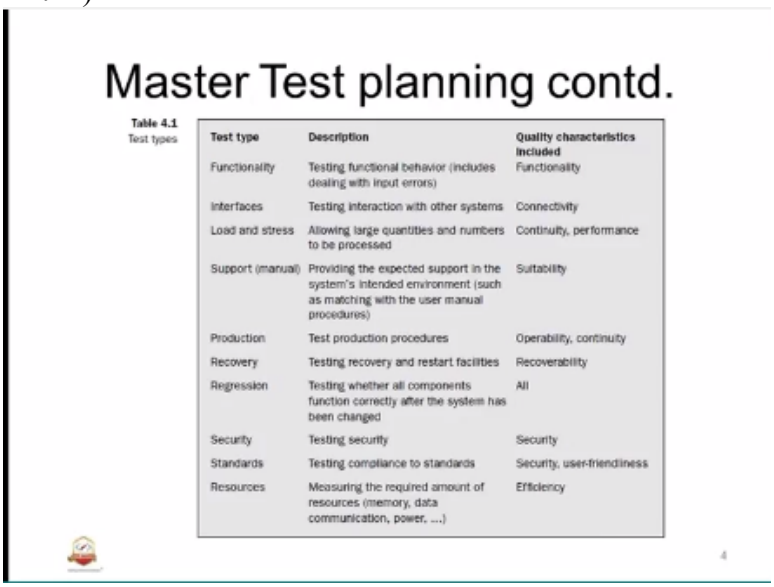
If the data is between which aspects are been tested? And which are been initiable? The master test planning then deals with the combination of the both. So what it primarily does is in each or what aspects are been tested? That it will identify and it will differentiate with who is doing the testing or executing the test? What is organizational entity? It is nothing but a set of team or set of allocated specialist or who is being involved for the entire testing activity. So they finally call this all will be finally put together in to the master plan.

That is why it is very much required master test planning in order to test the complex systems. Complex systems correct example I can code something like break control systems, engine control systems we have flight control systems fuel measurement systems, fuel management systems. Total recap fuel management it is a there will be lot of control systems, control subsistence are involved along with the sensors and their feedback.
And there is something called redundancy management, redundancy management is something like we have control sub systems one or the other channel it could be second one second sub systems takes over for controlling the intended of the systems or it could be on or the other systems which will be used in or to arrive at a subset condition that is where written so it is very important so we have two channel systems four channel systems etc.
Where almost each channel will have signal functionality each other inputs are very much used to arrive at initial which is good and which one is bad which can be discarded which can be used, so typically the flight control systems, this kind of intonencies very much so definitely they will be a complexity of software, hardware whatever the assumptions are involved within that embedded system.
So to have a development life cycle or a test execution we need to definitely have a master test plan so because it is a complexity systems where you identify lot of activities, lot of types mechanism or method etc so that is why it is so important master test plan. So what are the elements of master test plan?
(Refer Slide Time: 10:52)



Th4e basic elements what I try to highlight here are the generated to further locut of into driving the minuter details as veridical sector about test planning and test cases procedures etc, the order that test types define that is first thing of the master test planning. A test type is a group of activities with the aim of evaluating a system on a set of related quality attributes. So this test type identifies an evaluation mechanism for certain things like.
It could be upper form as it could be certain criteria it could be quality aspects in terms of coverage or standards so basically it does the it basically accumulates the group of various activities aim is to evaluate the system based on certain activities or mission. Test type's state

what is going to be tested and what is not so something likes a scoping test type, the test type should define what will be tested?
What are the activities? How I am going to group it? And what are the tests that will have to be taken care or getting arbiter? So that is what test types state about. That means system can be tested form parts of the, it can be tested functionality performance which requirements user aspects, so things are some of the quality attributes that describes the various aspects of system behavior.
Suppose there are standards exist for example ISO so that is with the 9126 DO178 so some of this standards also can be used of that arrive in that certain cartentures for test type. So these are the quality attributes so that can be used for the test types, that is several quality attributes are often command and just can have a single quality attributes likewise and test types state what you are going to be tested that is not having.
It will define who will do what types? And performance related or vice versa does it relate performance anti functionality in terms of which tester it could do which one? The below table we can see some of the test types based on the applicative all the mention it is the name depending on the complexity of the underneath embedded system.
(Refer Slide Time: 14:42)



This table we have test type, description and quality attributes that are included, what are the test types that are consequeted for master test planning? They are functionality, interfaces, load and stores, support manual, production, recovery, regression, security, standards, resources. So let us see one by one, what is the functionality test type? Testing functionality and behavior that means the functionality is been addressed here.
S0 all the features or the functionality of the embedded system will be categorized as the functionality test type, basically that is driven with the inputs, so input definitely will have error inputs to test the behavior at the particular functionality, and interfaces, basically it is interaction between couple of sub systems or whatever it is so that is what is addressed testing interaction with other systems.

Other systems could be sub systems or as to ESML entertain. So this is coming into call connectivity sometimes we can also mention the interfaces. Some communication is important so sensible component others, simultaneous communication quality attribute that we can attribute to interfaces. Then we have load and stores allowing large quantities and numbers to be processed that means we have certain load are inputs , certain quantity of input say 5000 rpm or 7000 rpm so others is something like 1000 to 7000 rpm let us say7000 to 10000 rpm.

Those kind of requirements, those kind of test type is coming into load and store, so what you want to substitute embedded system means allowing the higher power with a fully loaded systems that we will test the behavior of that particular condition with a large number of inputs are all strategies but it is fully processed with inputs. Next type is support providing the expected support in the system's intended environment that means the types the test types which will help us in terms of directly or indirectly arrive in the some of the results.

So it is the expected support in the system intended environment that means over all understanding of the some of the feature functionality behavior have some analysis etc all this will be part of this support. It is attributed as system test type. Next one is production test production procedures, that means we have a production environment all the production based testing will be done so that will have a quality attribute in terms of operative and continuity that means a decision should be able to operate with a new integration or maximum automation.

And then should continue same way as expected project one+ two in the given set of test aspects, then we have the recovery, testing recovery and restart facilities that is the like same rot power of requirements which may be close connected testing could be coming under test type of recovery where we have a test recovery and restart facilities we will help in restart type of tests and recovery quality characteristics.

Then we have regression you all know how testing whether all the components comes from this correctly of the system has been changed, so why system is changed? Because it could have been bug fixed or any upgrade would have done or any testing failures it will able to reproduce. So all this comes under regression any this can have any of this quality characteristic attributes all. Then we have our security, testing security.

All security requirements are sometimes we can have safety requirements also and terms of types called security. Then we have standard relating compliance which percentage of coverage will be there, complexity coverage etc will all come under standards so again some standards sub systems will have which also categorized under security of safety because for example if you take N space process what process? They call it as safety how the product is safe while this is subjected to all those features comes under this standards.

And this also should be in standards in to that which operate in that. Then we have resources, some of the sub systems that take the sub system primary can be used like power, memory, communication enterprise, etc all these will be equipments, all called as resources this is type of test type under this type. You can combine with this also interface as well as interfaces we can brought this is categorized as communication.

Since we have this in objective based on the complexity of the system, so these are some of the elements under test type unit some of the test types that are listed you can add more these of the

lead of the complexity that we have we need to process because this is very important in terms of master test planning, the next one is test levels second of the4 master test planning is test levels. A test level is a group of activities that is organized and managed as an entity.

So test type is a group of activity with the aim of evaluation system or set of greeted quality attributes here we define the test types under the quality attributes what is going to be tested? In test levels what we do is? We list out the group of activities and how it is organized and managed as a set of activity in going to be defined. So the table below is some of the common test levels of the embedded systems subject to are applicability,

So before that towards is that test activities is performed by different testers and teams as I said because we are dealing with the larger systems so we may use the different environments are various environments pertaining to some set features that is in under test so all this organizational aspects. As the reason for introducing this test forward that is why it is important to have ea test level defined for master test planning test levels.

Basically state who is going to perform the test and when So it tell three state specifically who is going to perform the testing and when is going to be tested the different levels are related to the development life cycle of the system it structure the testing process by applying principle of inter testing you know the interment testing is development process.

We develop small, small process we integrate one, one by and intimately we embedded a pull stop when the various compounds are development of satisfactory quality they are integrated into the larger compounds or substances so likewise we will develop the embedded systems then tested to check if they confirm the higher level requirements so we will start with the small substances and compounds then we will grow up it is higher in the interment testing.

If they confirm to the higher level requirements what is intermediated for that particular embedded systems and we that low level testing and high level testing, low level testing is focus on micro compounds like related to the earlier life cycle in the bottom most part of the remodel higher level test are test in the integrated systems or substances they executed later behind embedded systems of the remodel it could be simulated on the various places etc basically on the real life environment.

So we will go through the table of test levels all of this we have gone through seen in our earlier slides in different class session in this unit we have detailed but these about master planning it defines each approaches type of test or levels of test categories under the master test planning okay hardware unit test, hardware integrations test model in the loop software unit test host or target testing software integration test hardware software integrations test system, test acceptances test field test again these are all subject to applicability of the particular embedded systems.

As we need we will draw a different levels patent into the embedded systems hardware unit test the test level is low and it is done in the laboratory the lab testing the behavior of the hardware of the component in the entire solutions as definite it is very component units level it is called you can also mentioned it is as a unit test hardware integrations test means it was a low level it should be done in laboratory testing the hardware connections and protocols.

Basically we have can MDA whatever it is all part of the hardware integrations where we basically plug all the hardware or substances under the basic embedded systems and we will test

it there we have the model in the loop that is this categorized either high and low depending on what are the type of models that we use so we know that any usage of the project that we develop the models prove it whether it is going to what so basic purpose is proof of concept testing control laws optimizations is around the combines logarithm when it be proven first before we actually develop and employed of course in terms of memory performance we may have to optimize.

So in order to architect efficiently we may have to do internal optimizations all it will be covered under model in the loop also they called as MIL okay the next type is software unit test host are target based system of course I told you earlier hardware unit test basically a different hardware components want to be tested we can do it or if it not required then we can go ahead with the pure software components test actually.

Because this is components that those it is a FI because what will happen is while doing the software unit test on the particular unit we may come across bugs or issues so we assumed that sometimes they could be hardware problem they could be software problem etc to make it clear that we are good at hardware we start with the hardware unit test is can be done by the hardware team or essentialist we know about a or who is knowledge.;

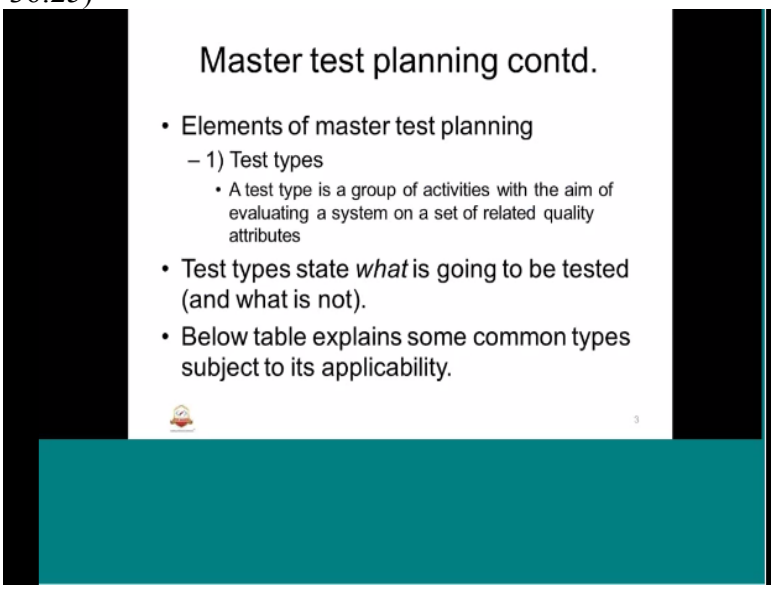
In a on the hardware components is again a low test laboratory plus host target processor is used of course laboratory has both of this typically it is used in the lab testing the behavior of the software components isolations then we have the SIT software integrations test is also a low level it is done with the help of the target code connected to the host we know that how the target is interface with IDE debugging, break point and that we use it move to the integrations basically here we try to integrate the different software piece into the system and integrate it we will pass the parameters test.

Or who is knowledgeable enough on the hardware components test hardware should be used, of course as both of this typically used in the lab, since we have this components, then we have the F18 software integration test, these are the low level, which used on the help of the target code connected to the host we know that how the target reaching the gate with IDEN whatever, and all that reduce to the integration basically here we try to integrate different software cases within the system and we pass it against the code, the integral behavior of the code, so it could become slow the data flow, that will be debugging breaking point.

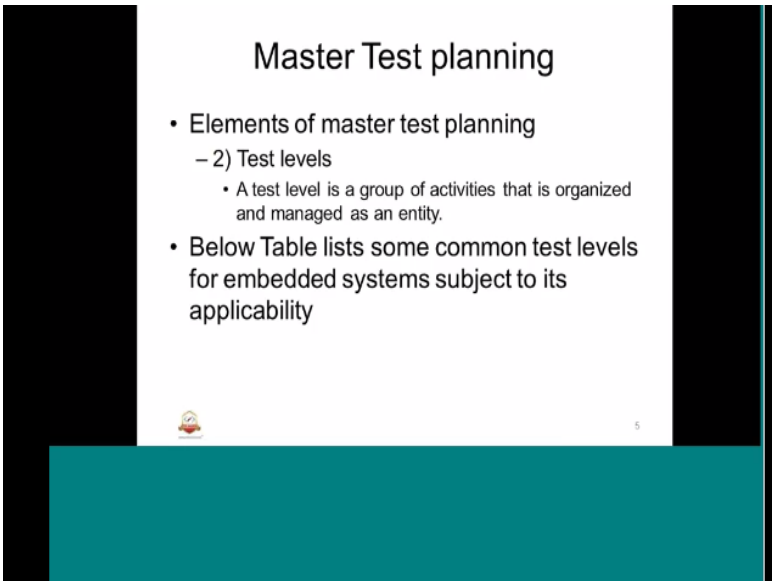
So testing integration between software, and debugging it to test, similarly, for next debugging test is happens, where we focus on the system and the register, while having to integrate the components and those software components how they have the things, so we only focus on the integration so software to hardware or hardware to software, this thing is hardware to software, here also we have squares and measurements, and target goal, whatever it is, to prove that the software is influenced on the entire register, and target system.

Then we have the system test as an integrated unit of the embedded system, because it works fine as an efficient, it is done with the help of this level called system test, it is high level of course simulated to real life, which is expect few items we will almost have the entire, say suppose system is what, simulated to real life means what is the actual system of running on the field suppose the fuel measurement or it is a speed, so we will want to take the embedded system

target on to the road which is an high speed of 100 miles which ever, instead what we do is how it become to that speed, the fuel actual fuel which going to feed is simulated as an inputs with those help of how it is begin what they are going to test.
So this is under system test and high level requirements, basically these system has large code as far as the software testing requirements of the system, a set of high level specification is business requirements or user requirements this is tested under processors so, the product can be accepted this is also done with the help of the simulated real life with the actual en-target, so testing the system interface is the purpose of the user specifically we use for the user perspective to the all aspects of expectations.
So, and the last one was on this perspective you can use the field testers some of the complexities features with the tested with the help of the environment something like Ableian or other Hoffman, like we have a temperature but as there integrated system sometimes we may have to test on the fuel.
So those things are categorized on the test, testing the system keys are done, it could be the attitudes, whatever it is all this will be taken care with the help of the features, so this is the teat plan under the master test planning.
(Refer Slide Time: 36:25)

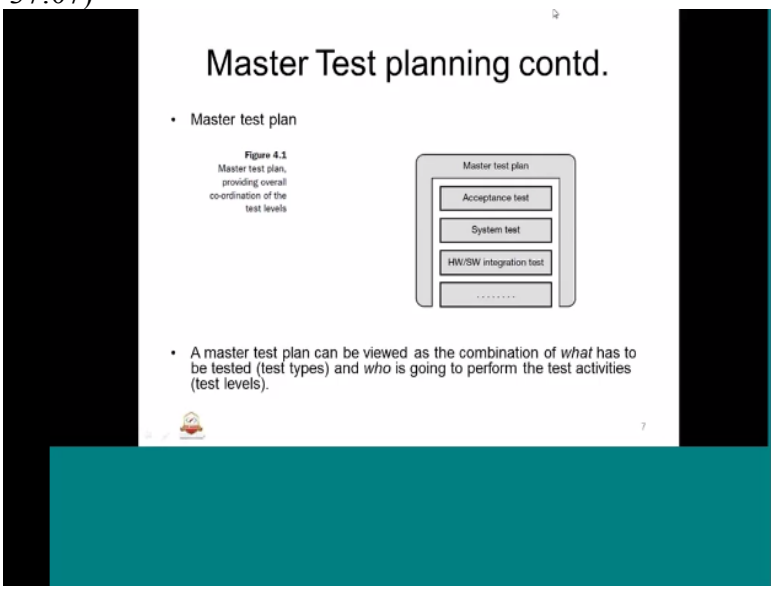


So we have two types test type which is what is going to be tested and what are not going to be tested and also it is a group of activities with the aim of evaluating a system on a set of related quality attributes.
(Refer Slide Time: 36:34)



Then we have the level all these attributes, how they are going to be tested, model loop or proper integration, system types of the components all these are considered for master test plan.

(Refer Slide Time: 37:07)



So, integration what the master test plan here is the picture we all know that we have understood already about this, another we have the defecting master test plan, don't get confused through what we understood is the same.

Here we drawn the picture of several levels of test, we have acceptance test, system test, HW/SW integration test which is one part of the master plan, so master test plan can be viewed in the combination of what has to be tested, that is the test types and who is going to process on it.

So that is about the master test plan, of course we need to called with the various test levels, it should prevent the tendency to be sent, it ensures that requirement for all test strategy is promoted, decisions has to be made at which tester is not support to this thing.

So, test in which system require the acceptance of course the specified resources such as special lines or just it could be paired, and expertise to use at all those have to be allocated to the various test levels of the certain functionalities of features.

Of course there is a timeline that also need to be drawn in the plan, so over all test plan has to be mapped of which defines to tasks and responsibilities for each tester, in the terms of expertise or limitations of each other of the testing plan which plans across this master test plan.

So that is why this test plan is over all called as master test plan, basically this is under project manager and the project manager or the test manager depending on how it is organized in the master test plan, so you have to delegate the each other.

The responsibilities are done by the test manage team or the test managers, so he is over all responsible for this, so what is about master test planning test level and test types, so three main areas of the master test plan let we consider it has the project manager or test manager, test plan before it should be a master test plan. Test strategic choices, what to test and how it is or how thorough it should be, that is the one of the important test strategies choice that has to be done allocation of resources. Next choice is, the resource could be form the proper equipment's and the resource could be from the human resources in terms of contd. The experience that user required to be taken care.

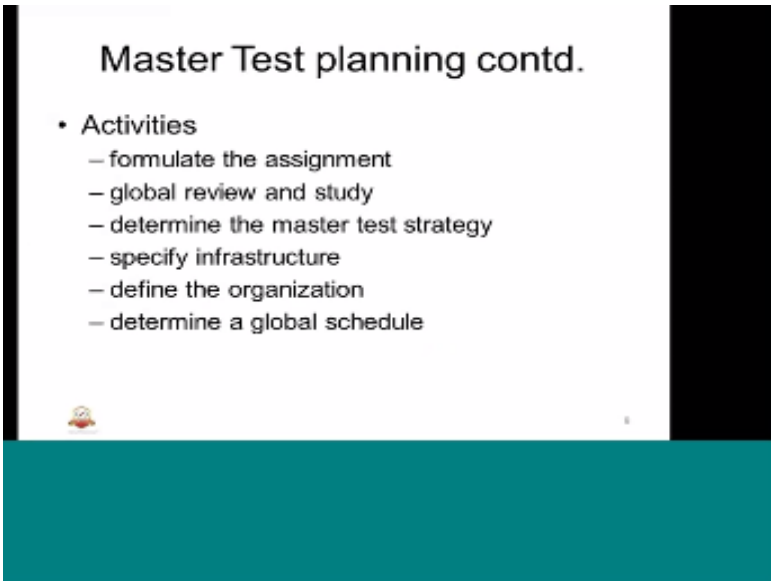
And how all these different in the tools for test resources or the cooperates all they are going to communicated, this could be involvement of vendor also, sometimes we may need to out some of the power supplies or measuring the equipment's, how they want to communicate between the disciplines, there also user an inserter for the master test plan and off course some master test plan will know, exactly what is expected from each of them, what level of support and resources they can expect.

The master test plan and source test plan that reach test level, we can have detailed test plan for each of the level, we can have a test plan for acceptance test level, we can have test plan for system test, integration test, team test etc., all this combine together from the master test plan is formed. It does not need to prescribe of each test level, but it can point to what can be done, like that it will be in test plan.

So whatever the details write it. So master test plan deals with the decisions that come in the area where the various test level is can either help or in the richer, so all these pointers will be part of the master test plan. Of course there are other aspects in the master test plan also, test list out sake of twisting it out these are maganentry that for understanding the related, that is test strategy, we know that it need to have a test strategy and the team for the same infrastructure and organized or organization.

Who is going to do what, this is basically some of the key element of the master test plan and we know that in Tm method all these above are drawn with the help of ultra-principal, what is ultra-principal? Life cycle, infrastructure, techniques, and organization, so this all will formally one test plan,

(Refer Slide Time: 45:15)



So, master test plan activity will highlight few points which are important. Formality assignment, global review and study that means what is the statically doing the outcome of the test and how the TPP in stack in terms of study, then it determine the master strategy, whatever date is on which the test strategy can be determined, specify infrastructure, define the organization, determine the global schedule.

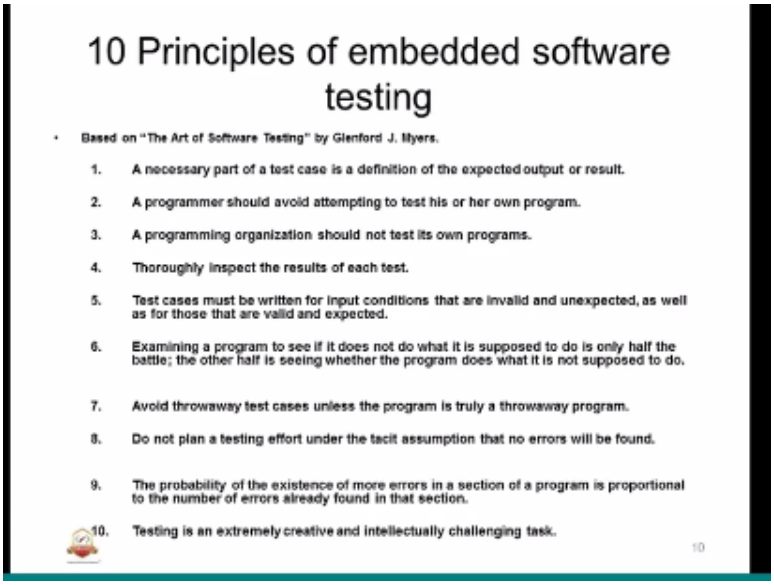
Okay, that is based on all these find above the schedule is drawn in terms of using the coming out of the execution part what is same, so these are some of the activities, so these are done by test manager and it starts other hourly activity in the term of developing process itself or other test process. This is done by developing a master test plan, having formulation of overall object is responsibilities for testing the data's in the scope, then the global analysis for the system or testers on the development processing these standards.

The next step is to use the information, what is being discussed and any doubt about how you can measure and about to taking the next step in terms of it could be a quality or performance etc., basically it should have the aim of successful in the system, having a done with the complete test. So this is the determination of the master test strategy and advance choosing what to do, what not to do and how to realize.

The test plan may be responsibility to pass the test strategy, it accompliate this whatever we had discussed about, he is getting accomplished with the help of master test plan, the required resources infrastructure or personal it should be defined. To manage this communication between all the involved with the required reporting all must be defined. How individuals are going to report and what will happen here, to whom all these will be part of the organization and test plan.

A good master test plan is all created in the quiet and isolated office, it is a highly, what it is called? Political task, it has lot of people involvement, basically it require the discussion along with the percolation throughout the organization. That is the good master plan drawn by the project manager or the test manager, so that is about master test plan.

(Refer Slide Time: 48:38)



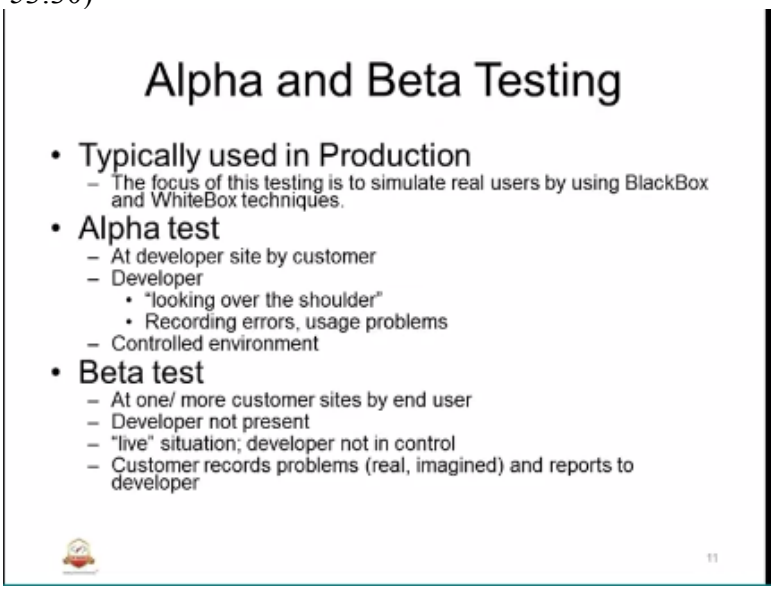
The last part of the, okay, so of the session I am coming with the couple of slides, I just to bring some of the pointers which will highlight overall what we have studied in this unit in the term of principals, this could set and done by Glen ford J.Maryse, so just want to point out this, so this is the 10 principles of embedded software testing. Necessary part of the test case it is the definition of expected result or output of this, it should have an expected output and report.

The programmer should avoid automatic testes or hardware program that means these independencies should be maintained. That means programmer should always try to test as well as in the independent way or it should not be bail towards testing. The programming organization should not test its own programs, that means one is out of program level and other one is organization, thoroughly inspect this results of the each test, that means we have the results outcome, even though is passed they should have specification of.

We should have a validation, why it is passed, we should instruct it, we should watch it, and we should make sure that the right results are achieved. Test cases must be written for input conditions that are invalid and expected as well as those that are valid purpose, so both have to be taken care. Examining a program to see that is not, what it is supposed to do, is only the half way battled, other half is seeing that program does what is not affordable, so that both of them are very proper, first we will do what it is supposed to do. The example telephone instrument, what it is supposed to do. And different telephone instrument what it is not supposed to do when our should have depilation are you should have a validation why it is pass you should instruct it you should watch it make sure that the right result are achieved. The test cases must be written input conditions there are invalid and unexpected as well as for those that are validated. Both have to be taken care. Examining the program to see it does not what it is supposed to do is only half their battle.

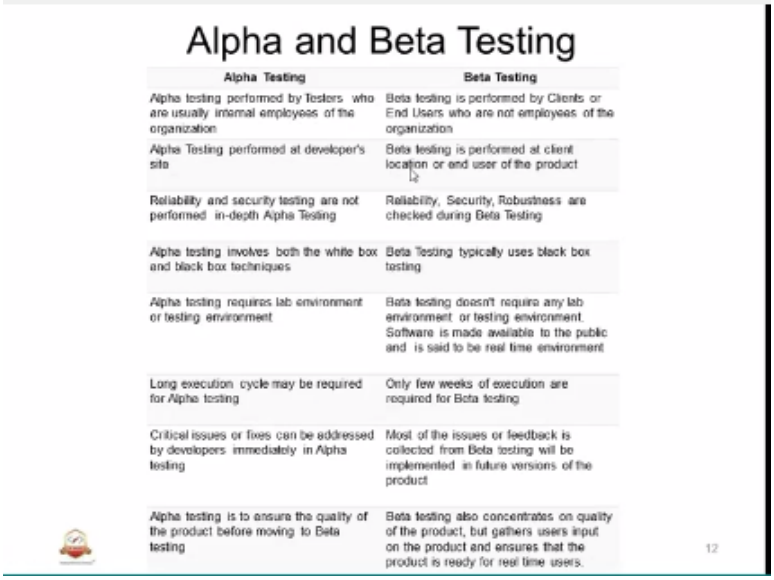
Other of you see whether the program what it is does what it is not supposed to do so that both of them are very important. First we will do what it is supposed to be the example telephone instrument for it is supposed to and the same telephone instrument is what it is not supposed to

do. Something is not required or something is not satisfied it is specification. The other principal avoid throwaway test cases unless the program is truly through a program.
So unwanted things you should unless properly before we may get a regular unwanted. Do not plan a testing effort under the tacit assumption that no errors will be found. So we should not waste having testing inputs that is for lack of putting some efforts should have an appropriate effort for achieving the success of the test.
It could be 100% good or it could have some errors. But efforts should be there always and efforts should be the right decision. The probability of for the existence more errors in the instruction of a program propositional to the number of error already found in an instruction is this very interesting principle. In always proven I have seen this and I am for this actually basically why because I have source code of I thousand lines by found.
So it some 20 bugs initially and it end of it test of thousand line I found 100 bugs definite those 80 bugs could be based on the 20 bugs which I found in the instructions. So this is the proportionality that already found errors could result in more errors. Okay, leave errors or more errors that could be extremely creative and challenging task. That means testing is equally important and equally rewarded it task that is what in the test. So that is about 10 principles of software testing.
(Refer Slide Time: 53:30)



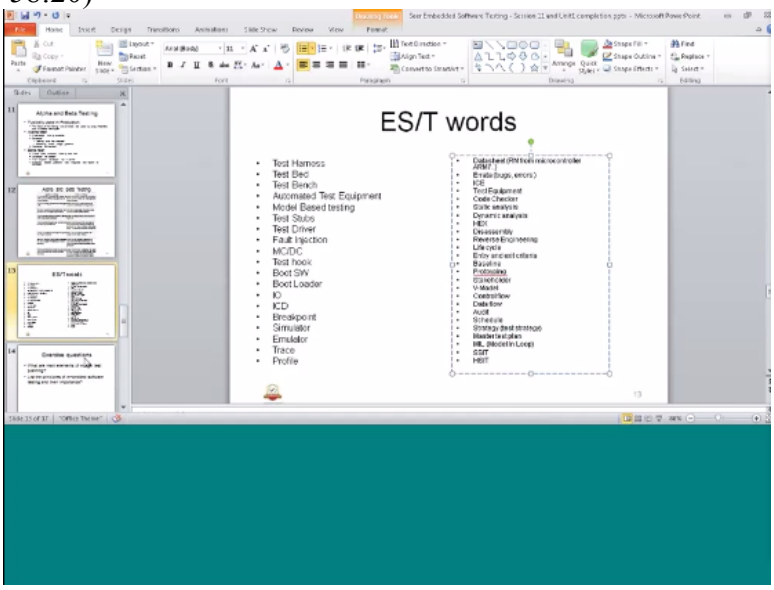
We have alpha beta this is basically the part of embedded software testing but some production is been used especially in the application set but as heard that alpha and beta testing for also it is used for the embedded system production environment. So just to understand what it is typically use in production the focus of the testing is familiar uses. Using the black box and white box testing alpha test done by the customer sorry call for test at developers set by a customer. Customer will have an idea of that product at the development set the developer looking over the shoulder.
Recording errors, usage problems etc. all these development prospective is taken from alpha test. Controlled environment, that means still it is not a fully-fledged or environment. The certain conditions that is so the next type of after the alpha test and beta test that one or more customer

sits by endings. This is enduring that is end user the real user of that will be testing the product at the requirement site.
As the result developer is involved for beta testing it something like a live situation where the product on the market to it. Developer will not have any control they will be reported of the issues that is lot of beta test.
(Refer Slide Time: 55:32)

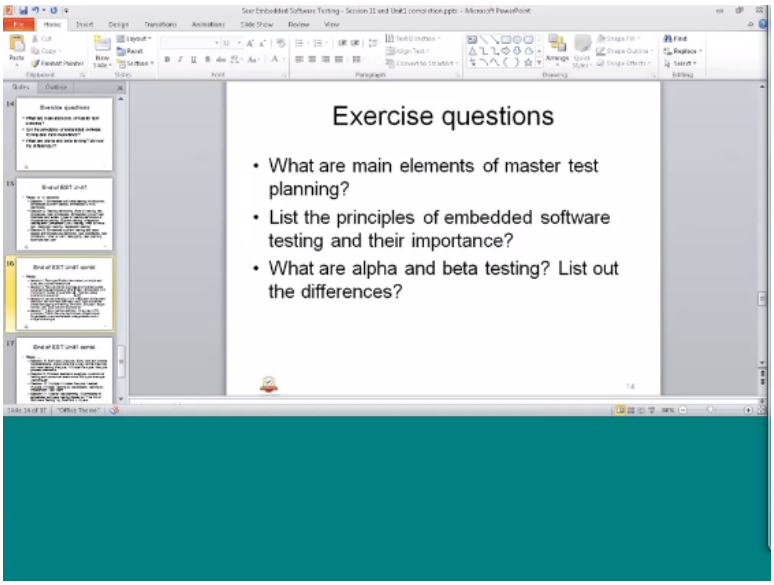


Customer record problem real or imagined and reports to developer. That is about alpha and beta testing. There is little difference between that alpha testing performed by testers who are usual employers of the organization. Beta testing is performed by clients or users who are not employees it is on the field. Alpha testing performed at developer's site. Beta testing is performed at client location or end user of the product. Influence systems have reproduced the test very uses actually for his claim. Reliability and security testing are not performed usually. This cannot be applicable for embedded testing but in general this is how they are using it. Reliability, security, robustness are checked during beta testing.
Alpha testing involves both the white box and black box techniques. Beta testing typically uses black box that is system level. Alpha testing requires lab environment or test environment. Beta testing does not require any, in lab environment or testing environment. Software is made available to the public or the end user and is said to be real time environment. Long execution cycle may be required for alpha testing.
Only few weeks of execution are required for beta testing. Critical issues or fixes can be addressed by developers immediately in alpha testing. Most of the issues or feedback is connected from beta testing will be implemented in future versions of the product. So, alpha testing will have a still chance of reverting back with updates of the product whereas beta testing it has to go through different chain of implementation versions.
Just to bring back to the factory and all that. Alpha testing is to ensure the quality of the product before moving to beta testing. Beta testing also concentrates on quality but gathers users input on the product and ensures that the product is ready for real time users. So, we have covered upon

details. Some embedded systems can be covered under this. This is basically they using intersect like telecom where they have OS like android or Google.
So, there leaves alpha test relieved products and given to the customers group of processors and testing which will use within the real time environment. And that type of testing is called beta testing to the customers.
(Refer Slide Time: 58:20)



Of course end of session we have words, it is the repetition of all hat maybe we can add some of this, some of the light touch from this class something like schedule and the strategies placed. Strategy or protective strategies both are same any other words which we have to look at. Master test plan of course the part of the test plan we all know that. One word is model intro so that is very important.
It is also called as MIN and we will also see master test planning or master plan. Okay. Model low we have SSIT, HSET, hardware software integration, software so intention. Okay the next part is the question.
(Refer Slide Time: 1:00:30)



What are the main elements of master test plan? List the principles of embedded software testing and their importance and maybe we can have one question on beta testing. List it out the differences. Okay so with that we will conclude this session. In the next session we will take-up the unit of embedded software testing. Maybe I will recap of all the sessions that we had for some duration to begin with, the ten the eleven sessions we had firstly then we will take up the second unit on testing methods, dynamic testing, model based testing, cordite testing etc. as part of the next session. Okay, thank you.