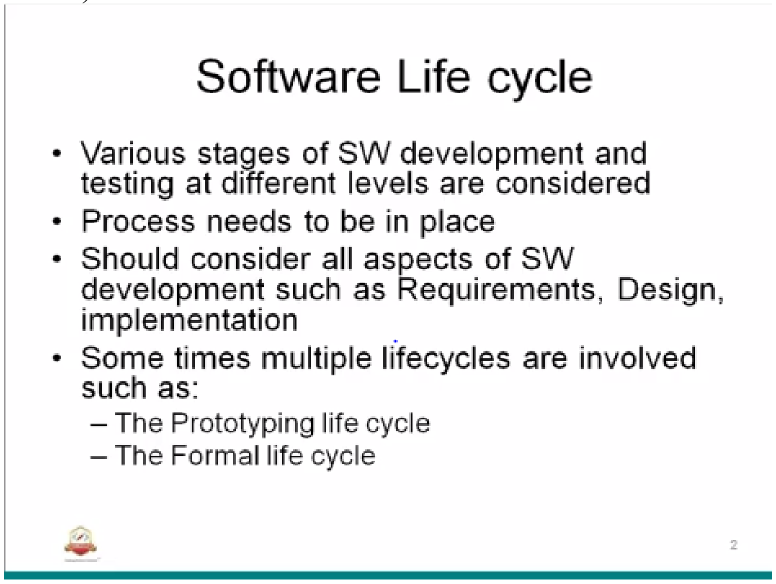
Embedded Software Testing

Lecture 8

Seer Akademi-NPTEL MOU

Welcome you to the next session of embedded software testing lecture 8.

(Refer Slide Time: 00:11)



Software Life cycle

- Various stages of SW development and testing at different levels are considered
- Process needs to be in place
- Should consider all aspects of SW development such as Requirements, Design, implementation
- Some times multiple lifecycles are involved such as:
  - The Prototyping life cycle
  - The Formal life cycle

2

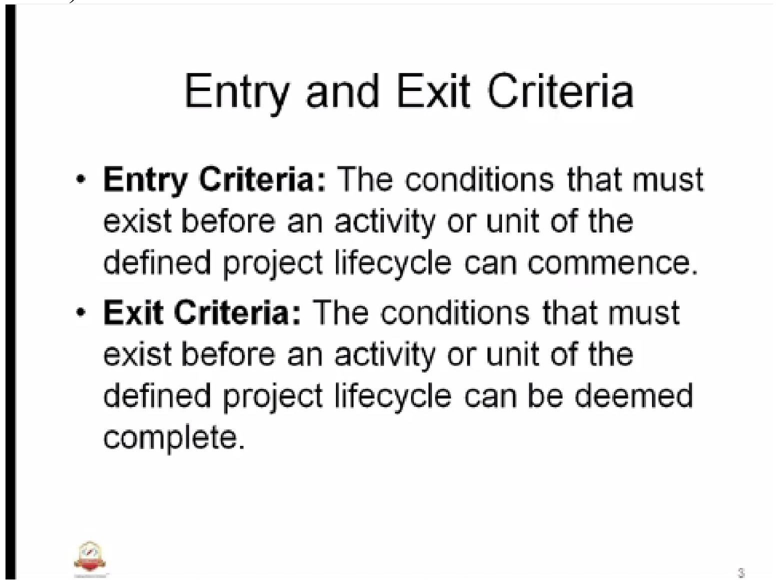
Here we go through software life cycle prototyping some chooses how; its going allocating is recorded something, that's why it restarted where embedded software testing, in this session we will study about life cycles, that are followed for embedded software development and testing typically it is both and in detail we will understand what was specific software testing life cycle can commence elements are required, so we need to know some of the definitions are there, criteria and there are sample types of life cycle, basically we need to understand what is life cycle? What are the elements that are required to have a life cycle implemented?

How it is organized, example will go through then will Study in detail about software testing life cycle and before that I just reemphasized, what we study in the session 7, we started an exercise for the session 6. then we studied about T-Emb method is one of the important method, that is being followed, what it tells you, it will pushed them earlier at its complement system or embedded system of the test and it will apply the approach.

It will phase the projectory methods, so the principles are we do like cycle, infrastructure, techniques, organization, so we studied about each of the software's life cycle, techniques, infrastructure and organization, then some of the characteristics to be related for each of the embedded system in terms of critical systems and hardware restrictions, it executes the behavior of embedded system under test it is to be the characterize based on that we will apply the t-emb method.

Then we have gone through software commercial program, what are those categorize in terms of the measurement or in terms of editing, o in terms of debugging, so all these are categorize of commercial tools, so we had an example of snapshot of a typical embedded system tools. Also we have studied about the words different words, what is software cycle, we know the various stages of software development and testing different levels are used, so those literally consider or

implementing and embedded software systems and testing and so those failure are called development to be used are consider at different level that is the things about software life cycle stages.
So some of the points are process to be there, there should be defined process for stages of development and testing and life cycle should consider all aspects of software development such as requirements, design, implementation and again requirements could have a derived requirements, requirements studies, requirement analysis then the design will have a high level design and low level design, implementation can have low level implementation and debugging testing, so all this should be consider a complete software life cycle, the complete software life cycle will be divide into multiple software life cycle, so in general.
What they do is they do a one stage prototyping life cycle then they will go for formal life cycle, this two typically they are using with in terms of embedded software system off course there are different types of life cycle that is going to be apply for each of them or the way how they want to organize it. Basically in these three implement has some under software systems, where first user prototyping then they do the formal development, so the prototyping will have a prototyping life cycle the formal will have formal life cycle, so we will go through each of this how it is defined whatever it is contents etc….
(Refer Slide Time: 6:07)

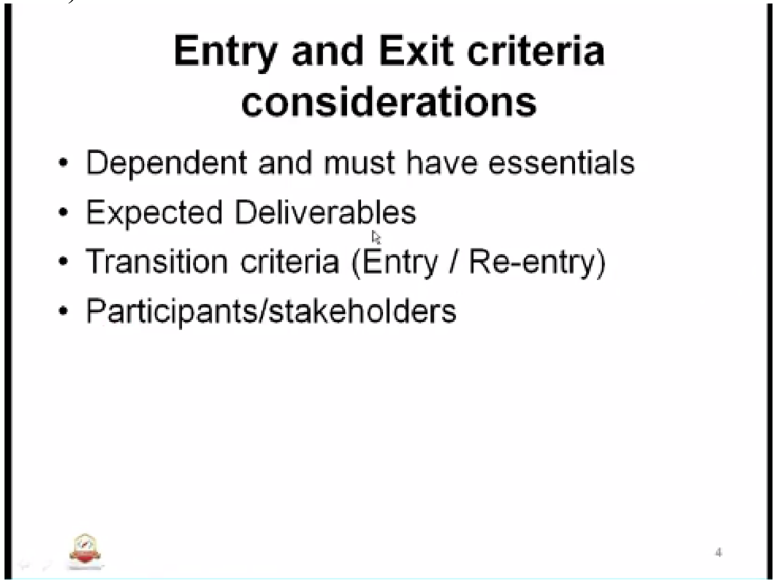


Entry and Exit Criteria

- **Entry Criteria:** The conditions that must exist before an activity or unit of the defined project lifecycle can commence.
- **Exit Criteria:** The conditions that must exist before an activity or unit of the defined project lifecycle can be deemed complete.

Before we studied the protyping and formal life cycle, we will have a understanding of couple of words which are important entry and exit criteria so we know that as software life cycle will have different stages or elements of the lifecycle so each life cycle are the stages have to have an entry, have to have an exit that means what is the entry? Entry is nothing but the conditions that must exist before activity or unit of the defined project life cycle and commands means before we start of the life cycle that conditions that are required to start that activity has to be existent or should be available.
So entry the criteria to start that, that’s why it called as entry criteria, exit criteria the condition must exist before an activity of unit of the different project life cycle can be deemed completely. That means in other way before we announce or before till that cycle or that activity is complete,

there are different condition of criteria source need to be satisfy, so those are all called exit criteria, so these two are very important items that are used in a defined in the software life cycle for each of the stage, stage one will have a entry criteria conditions, stage one will also have exist criteria of different conditions, similarly stage two will have an exist criteria of the rest stage, similarly all the pages of software life cycle will have entry and exist, this two must exist for moving different stage on the life cycle. It could be any life cycle that will take this process is very important.
(Refer Slide Time: 8:08)

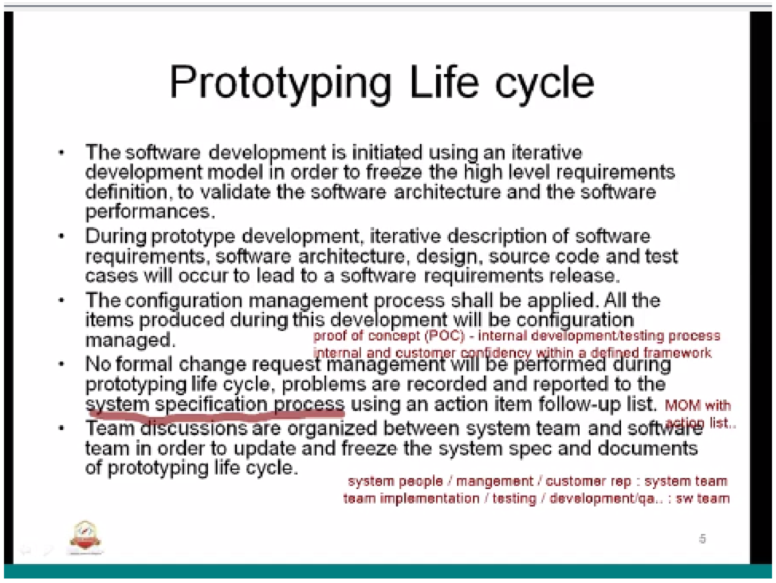


Entry and Exit criteria considerations

- Dependent and must have essentials
- Expected Deliverables
- Transition criteria (Entry / Re-entry)
- Participants/stakeholders

4

So on the consideration for entry and exit criteria as in below, it dependent must have essential, that means entry and exist should be layers and exist dependently on that particular stage, so without this it will not know. Then what are the expected delivery of expected the deliverables in that stage before we call has a exit, there are certain deliverables the stage ends with that is the part of the exit consider, then how it plan this entry and re entry, that means those the sage three can be said as complete by defining a so an so deliverables and so an so deliverable have to be place before knowing the stage three, it is stage four, s the stage three to stage four transition will have a defined elements.
Those elements are called transition criteria function from stage three to four that means the stage three will have some entry, the stage three will have some exit, also the stage three can re entry again based on certain conditions getting a transition within that so that also called entry and re entry for that particular stage and entry reentry criteria should identify participant or stakeholders for that, re enter into the same stage or process nothing but re entry functioning, expected deliverables are for each of this and certain dependent and must have items for each of the stage that need to be identify.
So basically that must be consider so that is difference of the considerations so that needs to be used for entry or re entry of the life cycle process elements, so we know that prototyping life cycle and formal life cycle are used in general embedded software development and testing, so what are those we go throw.
(Refer Slide Time: 10:59)



Prototyping Life cycle

- The software development is initiated using an iterative development model in order to freeze the high level requirements definition, to validate the software architecture and the software performances.
- During prototype development, iterative description of software requirements, software architecture, design, source code and test cases will occur to lead to a software requirements release.
- The configuration management process shall be applied. All the items produced during this development will be configuration managed.

proof of concept (POC) - internal development/testing process
internal and customer confidency within a defined framework

- No formal change request management will be performed during prototyping life cycle, problems are recorded and reported to the system specification process using an action item follow-up list. MOM with action list..
- Team discussions are organized between system team and software team in order to update and freeze the system spec and documents of prototyping life cycle.

system people / mangement / customer rep : system team
team implementation / testing / development/qa.. : sw team

5

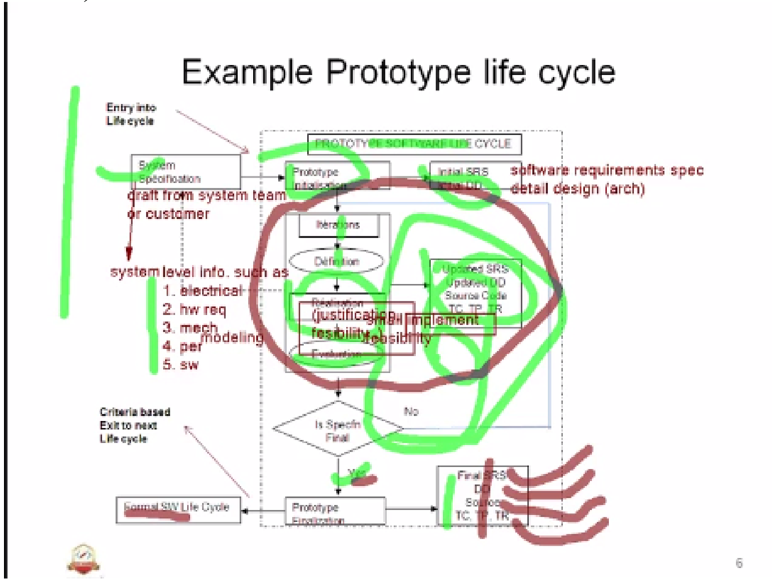
Prototyping life cycle, typically prototyping life cycle is like this it is consider of blocks that are involved instead there is an entry in to the life cycle of the prototype and there is a exit, I will explain each of them, there are in this plotted block there is nothing but the life cycle of the prototype, so its start with something like a box, this is the inventor entry and will go through what is that box, similarly I have to come out of this life cycle whether a exit mechanism and bottom you can see example is system specification there are see formal of the life cycle exit, that means it is ready to go for the formal life cycle it is the next stage the exit, example prototype life cycle what is used, it is the protect life cycle entry and exit items which are mentioned for this life cycle.
So prototyping life cycle the software development is initiated using an interative development model in order to freeze the high level requirements definitions to validate the software architecture and the software performance. That means by emberly what we will do is something like we will identify iterative model and the goal is to identify the space in high level requirements and we should able to validate in earlier the architecture what we identify the prototype and this software performance what it would be, it would be to perform the various requirements that are required and the during the prototype development it make a description of software requirements, software, architecture, design, source code and the test case will occur to lead to a software requirements release.
That means we are majorly we concentrate on requirements or else it will be architecture or majorly it is the software requirements and this but to you need the requirements unit of requirements you need have a some sort of the prototype architecture, prototype design, prototype source code something like prototype in the various aspects to prove that this requirements are variable and through to these aspects of these elements are the items that are called an as here should be considered. So what we do is we will have a configuration management also. Configuration management process and all the item produce the during this at all during the configuration management, it could be a architecture it could be design or source

code all this part of the prototype, all those should be configure and should be under the configuration management.
So the life cycle will have all this elements called under the prototyping and the n formal request management will be performed during the prototyping life cycle, so what is performed is equivalent what is not there is here actually that means there is no formal change, here formal means from the end system respective from the user prospective or from the customer prospective, so customer has given a end requirements, so achieve an end requirements what we do is we will divide the end requirements or the formal development in to prototyping. So prototyping we will not release to customer, it is something like internal information or prove of also it is called proof of concept; this is one of the important thing that embedded system industry they are following. So proof of concept is something like an internal development for testing the process, off course the difference on the management how they want to project these customers also can be involved basically it is from the internal confidence.
Internal as well as within the defined the frame work, the frame work is nothing but the prototyping and once we go through go for the formal life cycle there could be some sort of a approval from tester, it is usually take as the important whether his type of stage of the prototyping, so which is want the exit the area and for doing that we made or we may not made a formal change request for an element which will be used for prototyping. That means any of what are the way of doing all this software requirements pictures design source code and all may not required formal frame work it will all be internal defined block, no formal request management will be performed during prototyping life cycle, problems are required and reported to the system specification process using an action item follow up- list.
That means a simple action in a term which will be there I will explain one of the how action will be used it is also called as MOM with action list. So they will follow the up list for an action item list that will be used as I said during the development of required, that could be some bugs or some re in with to the acceptor or some disability updates, this needs to be reported, these all are part of the system specification because we are not specified with the requirements, so that is very important fact that specification is follow and solid in the prototyping, so that is one of the important criteria, before we move into the next stage, for doing that we will have a team discussions, that are organize between the system team and the software team and one thing that we need to know is that the system is usually defined with the help of customer as well as with the help of the system people.
It could be management or the senior, so I would note some seniors and customer can also involved or a customer representative in the system level and the team which are responsible for implementing, team implementation, testing, and development whatever it is, so you know you can call that is nothing but the embedded team or the software team, so these we will closely work an impact to make it a specification process to arrive at completion of prototyping of the prototyping life cycle.
So if it an exit model as I said prototyping is an iterative development model read of iterative mean the process item which are there within the prototyping like constructer, design, source code and test cases will be iteratively developed as you progress we previously take them again

to control on those items. So that is an iterative model and prototype development, iterative description of software requirements, architecture, design, source code and test code will occur, so finally the end goal is to use the exactly the end software requirements that should be released and the all items will be completed because some times what will happen is before we do the or why we doing the formal obtains we may need to re visit the what is the cases on which decisions are taking care, so all the decisions are taking care etc…. this also called as the POC proof of concept that is basically an interventional, internet development or confidence aiming stage.

Here customer are also can involved and there is no formal inputs for changes or required taken consider for the prototyping and system specification process will be arrived with the help of the stake holders efficient stake holder and like a system team or software team, so these are the prototyping life cycle items I will explain a within a example.

(Refer Slide Time: 21:05)



Software prototyping life cycle, you can see a exit or a block, this bock is a prototyping software life cycle, this needs to be entered then it is called software life cycle started, for that what is the inputs system specifications could be a customer given and draft from the specification and system team or customer, so this is the starting point that's what mention, entry in the life cycle will start with this procedure because what we are going to prototype we know, that has to be a some draft of specifications so with the help of that the life cycle keeps starts, so what are those drafts so what will you stage for type insilization, so will start with the stake holder T magnification who will do what and all that and those insilization as the result of that, we will come with the SRS or a DD, here is a that means software requirement and specification.

It is will design or it is called architecture we can also defined as the architecture, I give an example it could be this or it could be other one also, but the flow is this whatever the arrow of the entry and exit this is all some of the primitive items that need to be there for prototyping. Then once we have this page which the initial SRS and initial design we will do an equation of this items, by defining realizing and revaluating, so this will keep on getting and as the result of that it could result system speciation up data and same the system specification is to have a upgrade SRS and design and that design and SRS we will go through definition of that relavation, here relavation means what we do is we will try to implement or feasibility and a try to understand.

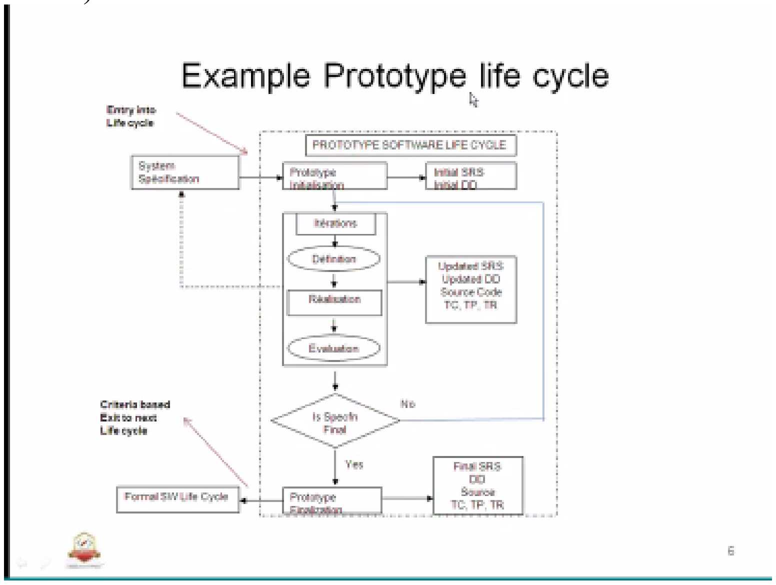
So basically what they do is they do some process in the half and system what they do is they develop a different models connected how they interact what are they inter faces, what are the signals slightly to flow and all that some part of the modeling, may be in other class I will explain what is the important test development and also there is something called model base testing, so either the model or it is a paper work or a visual or a diagram whatever it is, to make it realize that this specification will have a exactly mentioned a requirements, that is need as SRS they need the design and the prototype, so with this equations with the help of upgrade system specification will have a SRS, don't get confused with the system specification, system specification could have something like a system level, will we explain just what is the system specification, system level information such as it could be electrical requirement could be hardware requirement, it could be a mechanical, it could be performance it will also have software.

So all this will be part of this system basically in embedded software will take of the software also for doing the software implementation whether I can perform the software on the alternate system and system will have electrical hardware requirements and all that, so this session of the system level information this will be having but we need only the software related things that's why we will be concentrating on software requirements specification SRS, it will design the source code all specific to software and as for as the testing in confident we will use the test case on the prototype result of in terms of test result, so this will be iterated understand it will be iterated model part of the thing, in the prototyping life cycle, the evaluation is with the help of group of test cases procedure that will be used for evaluating it, so this will go in cyclic way or iterated type way.

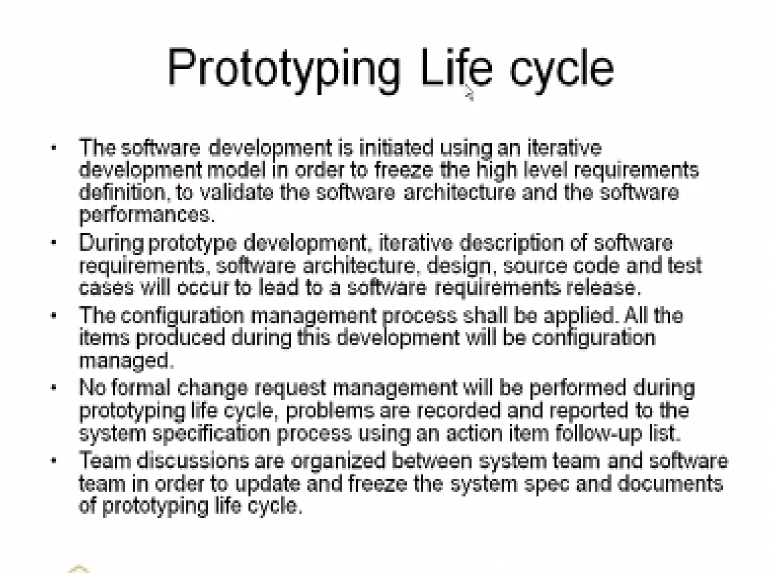
Ones it is done what would be the specification wile view or reveal the evaluation so if this specification if it's not then again we will it may talk that actually defining the specifications items how it can be reliance's so you could have a visibility important data's has to be taken care for initiative process ok once disputative items are done then once specification then we are going to have the finalize of the prototype that means we have a refined set of or an mutual working model of the SRS and a fluent concept of bromine and the impulse source code to an extent its working fine.

And we have few testing's sample procedures and reports which could evaluated that we have a proto type fine box so these are some of the prototype life cycles take example what I have got amputating in that will be used before we go into the next life cycle its nothing but formal life cycle I repeat we start with an entry into the prototype life cycle specification that is that is system specification which will be entering prototype software life cycle and with an initialization that means we will initiate with the prototype life cycle and we will initiate detail designing it could be architecture then these initial

Will be with the source code we initiate it we defined it each of the requirements of the specific functions of features that are required as the aspect of the system I said the system will have many high level system level information out of which we will consider only the components of software related items that will goal of with this prototyping we do update of the SRS and the source code sample and address procedures and sample relates.
Then we will re-evaluate against that getting fine or it requires any updates we will take this part we will come back again we will take this part we will come back again so like this we will continue once everything is fine we have to get to go for finalization and we will have a final concept working prototype which will fetch final SRS or erupted document bill code so these are all the prototype items
(Refer Slide Time: 31:01)



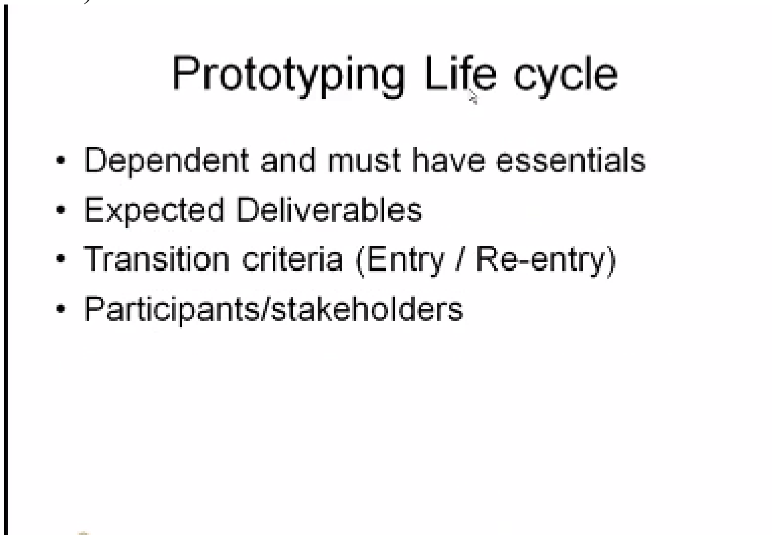
We have done with the prototype life cycle so we know that the next one will be formal level prototype which is an important aspect of embedded software systems.
(Refer Slide Time: 31:03)



Prototyping Life cycle

- The software development is initiated using an iterative development model in order to freeze the high level requirements definition, to validate the software architecture and the software performances.
- During prototype development, iterative description of software requirements, software architecture, design, source code and test cases will occur to lead to a software requirements release.
- The configuration management process shall be applied. All the items produced during this development will be configuration managed.
- No formal change request management will be performed during prototyping life cycle, problems are recorded and reported to the system specification process using an action item follow-up list.
- Team discussions are organized between system team and software team in order to update and freeze the system spec and documents of prototyping life cycle.

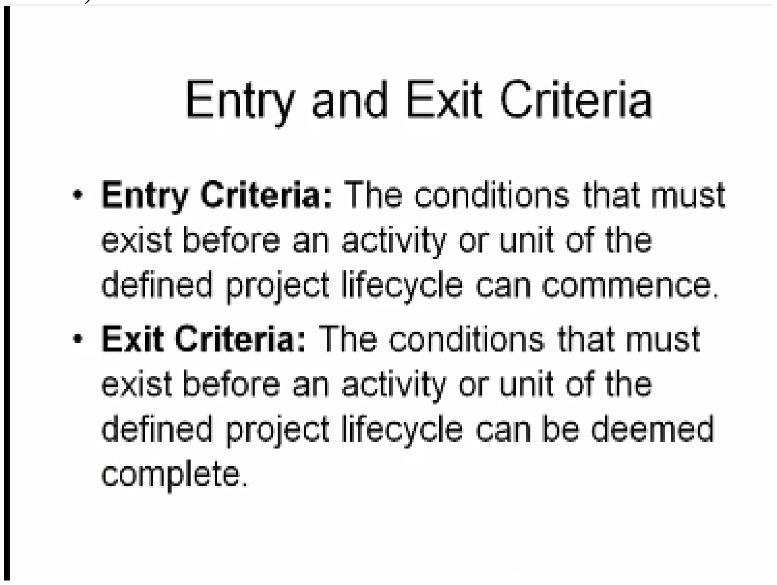
(Refer Slide Time: 31:04)



Prototyping Life cycle

- Dependent and must have essentials
- Expected Deliverables
- Transition criteria (Entry / Re-entry)
- Participants/stakeholders

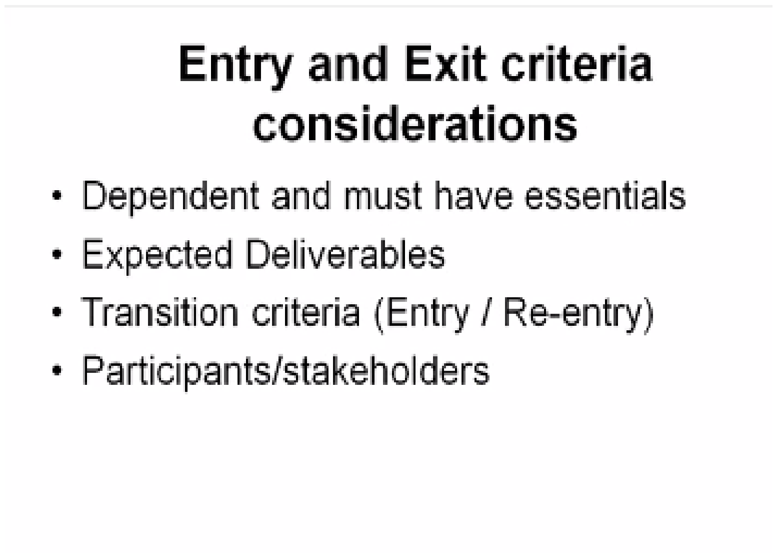
(Refer Slide Time: 31:06)



Entry and Exit Criteria

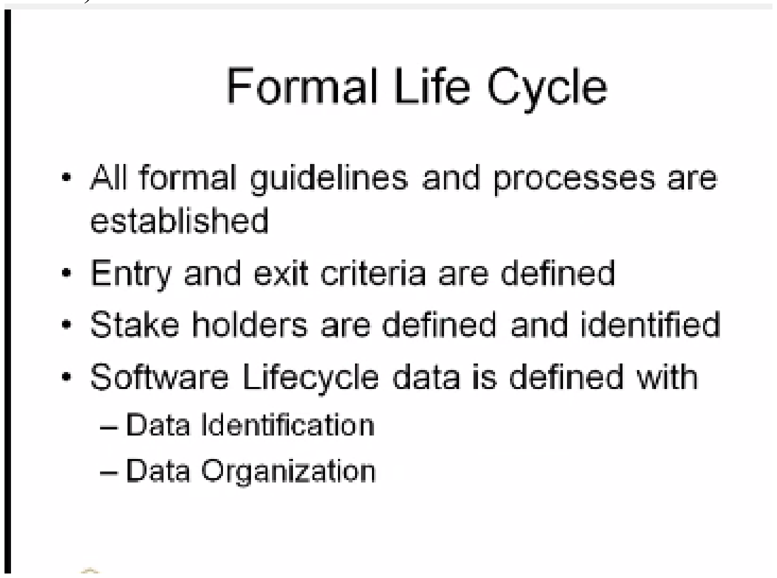
- **Entry Criteria:** The conditions that must exist before an activity or unit of the defined project lifecycle can commence.
- **Exit Criteria:** The conditions that must exist before an activity or unit of the defined project lifecycle can be deemed complete.

(Refer Slide Time: 31:09)


Entry and Exit criteria considerations

- Dependent and must have essentials
- Expected Deliverables
- Transition criteria (Entry / Re-entry)
- Participants/stakeholders

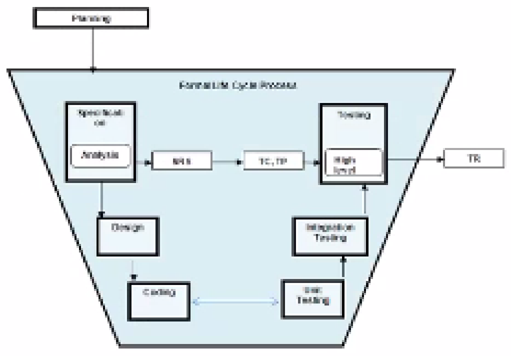
you may wonder why we need a prototype life cycle so what will happen is if I had a system and a system requirement and I don't have a confidence to start and I cannot commit to say 6 months I will develop a small embedded system and I will tell you where it is not easy actually and there has to be a through a proof of mechanism etc. so that what we will do is we will develop a small prototype it is really a short and most complete actual life cycle and we will install affiances stage and we say we are good establish of formal life cycle and prototyping to formal life cycle can happen at any stage you don't need to have everything in places according to that.
What will happen you will only be needed an only tact picture and time for that tact picture requirement for work and it could be quickly done for that you have already studied everything you come to conclusion that and move for next lecture ready formal life cycle so ready for the criteria the criteria now all that criteria will be on that life cycle so that will be for the embedded software system life cycle
(Refer Slide Time: 32:46)


Formal Life Cycle

- All formal guidelines and processes are established
- Entry and exit criteria are defined
- Stake holders are defined and identified
- Software Lifecycle data is defined with
  - Data Identification
  - Data Organization

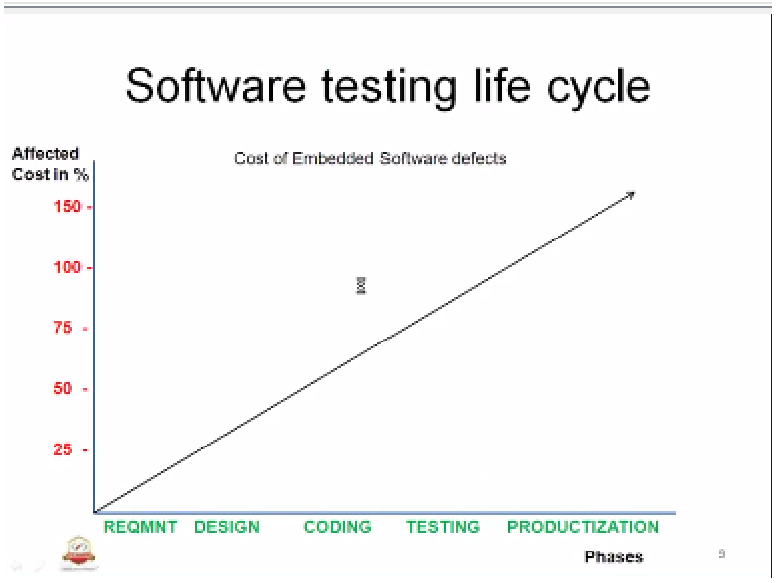
So what do we do in formal life cycle how formal Gide lines and process are established that means what are the formalities that we been in terms of implementing to complete formal life cycle using followed established entry and exit criteria are defined and entry and exit criteria for those that is the condition that must be there for each of the process icon before it starts and emulating something like it should be complete before its happening next cross data.
So that we will be defining the process we know that in prototyping we don't have an impetrating part we have an entry and overall we have and exit but in formal life cycle we have implied technique for interrupting stages it can be for formal life cycle and stay quarters are being defined who is going to do what and it will be identified as well and that means what is the theme and what are the things it would have so those are all the part of the stack hold identification and definition basically software life cycle data is being defined with data identification and data organization.
So the life cycle will have several or various type of data involved data will be identified first and then how the data is going to be organized or managed for implementing the completely the software life cycle will be part of the formal life cycle so we will take up with an example.
(Refer Slide Time: 34:56)

Example Formal Life cycle



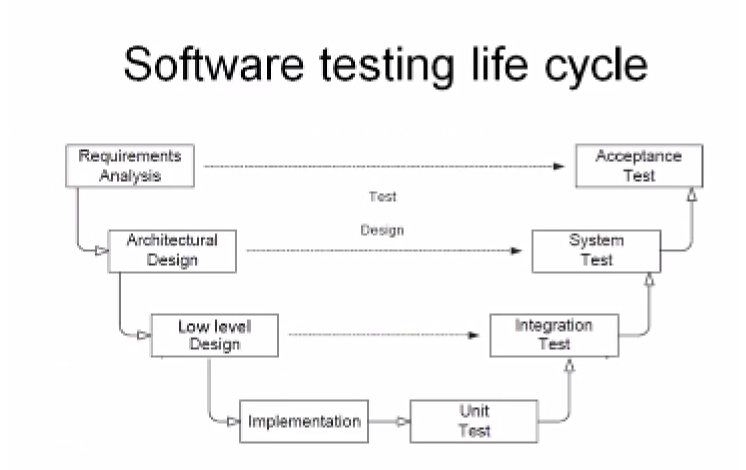
Of formal life cycle how it is so we have a plan to enter we know that what is planning in one of the last session I have told that the development on principal plan which will have all the involvement how this will be used in one of the guide lines process everything will be a part of the planning once we had a planning further we will actually start with the forms life cycle so I had put a simple life cycle that is called as a V life cycle how it comes v need an interesting part basically it looks like a V shape.
So why shape is like that V goes across the different stages in peak it's going to narrow down its going to go deep and then final elements that's why it's called as v life cycle what are the items that are in the V life cycles so we have a specifications and such specification we are being done in the stereotypic so we have a good specification which is nothing but the v schedule of the its being better of that activity so this could also be done in high level or low level so they are called as high level deny and low level deny whatever its once we have a low level deny.

We are decoding that means coding is done after its being in place coding is nothing but implementation it can be rectified development perspective so as to development perspective we have testing prospective basically we go in connection so how it goes is as earlier I told and on partial desire we go into testing prospective so for doing the testing we have an high level testing in high level testing the input is we dint care about design coding or anything we always look at input.
So we won't consider any other presentation of the desire there are certain procedures for high level and an activity adders the commercial test interacts so these will be higher level next for design we similarly develop an integration according to the procedure it's on testing procedure and outcome of that is TR so unit is called as ITP integration plan of this procedure but in integration there are high level and low level but the way in integration part is not being worried different functionality feature is being segregated
Each model how they are being integrated are all considered and that's planned and we proved it we don't care about their hieratical level improvement can be developed in terms of this out coming form but to map with what we do and integration we will definitely be taken the input but during the intense we have the source code of the implementation item sometimes what will happen is they will also use the multiparty things that depends on how you want to differ so what we do is they will give practice for high level along with the coding source code and the prime input for doing the unit testing and testing is also called as compound testing so it is also the component of each which are implemented on or coded they will have a test scale process and resting in and all this TC,TP,TR.
For doing the unit testing we use the source code complementation item sometimes what will happen is they also use the low level deny that depends on how we want to differ they will further divide this block on possible divide clock were they do unit testing along with the coding which can be taken care source code is a prime input unit testing its also called as compound testing each of the components or each of the bits which are being implemented on the code will be tested again they will have and rebuilding in result specification delighting and coding that's why it called as remodeling.
So in this session we have a retyping done in the previous prototype life cycle and formal specification for that specification we developed all the items that are required in the development so this a formal life cycle example and we know that why we need for testing identification for each stage because it's very important to identify the stages so that t6he cost of maintenance or the updates of the products.
(Refer Slide Time: 42:23)



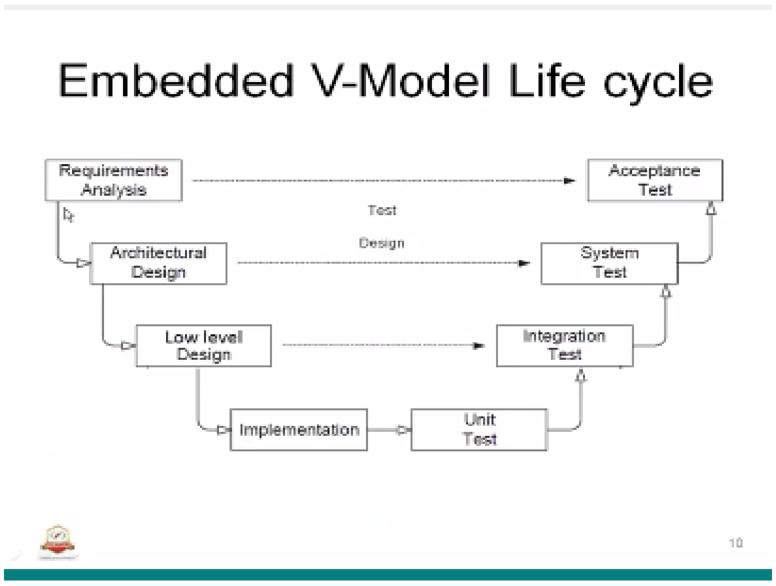
Software testing life cycle

One of the administration which has identifying the cost of the defect it grows up as we go deep into the proto initial that is the cost will be very high the goods could be at any stage so what we will do is we will identify the stages and then the product that's what the story of the slide
(Refer Slide Time: 43:10)



Software testing life cycle

Formal life cycle is being formal typically how are there actually we have high level design and low level design we have an implementation
(Refer Slide Time: 43:41)

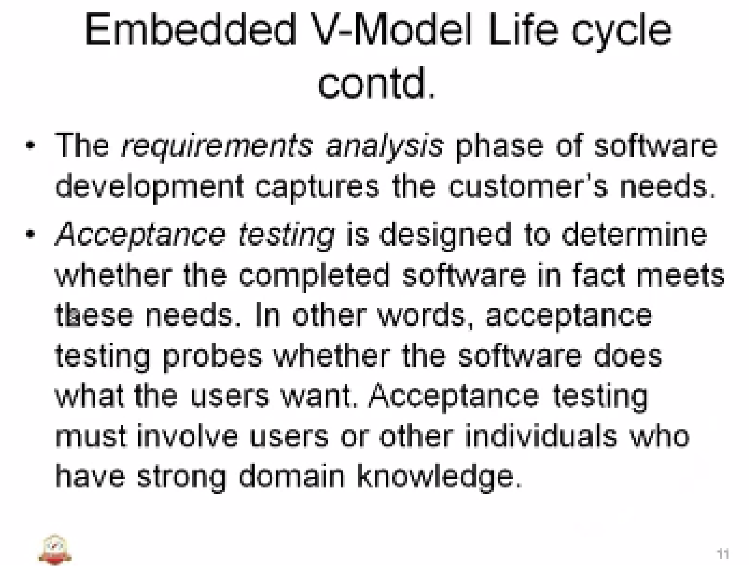


Embedded V-Model Life cycle

These are the one side and on the other side we have an integration test and there is system testing and then it goes to the requirement analysis and the acceptance test and it has the architectural design we have the low level design and high level design and then o the implementation and finally it goes to the unit test

Introduction to software book is there so in embedded there are three stages this will go in and for the each of the cycle and what typically at a this will go in parallel we don't need to have the unit one thing is done for each stage we are going to initiate this plan for high level test plan can be developed for the particular requirement we can still go with the we can rate that will go with parallel to the requirements it's done and going for the implementation the integration is being planned for low level updated or it can be started in parallel to these items the plans and the procedure in parallel with the items.

(Refer Slide Time: 46:34)



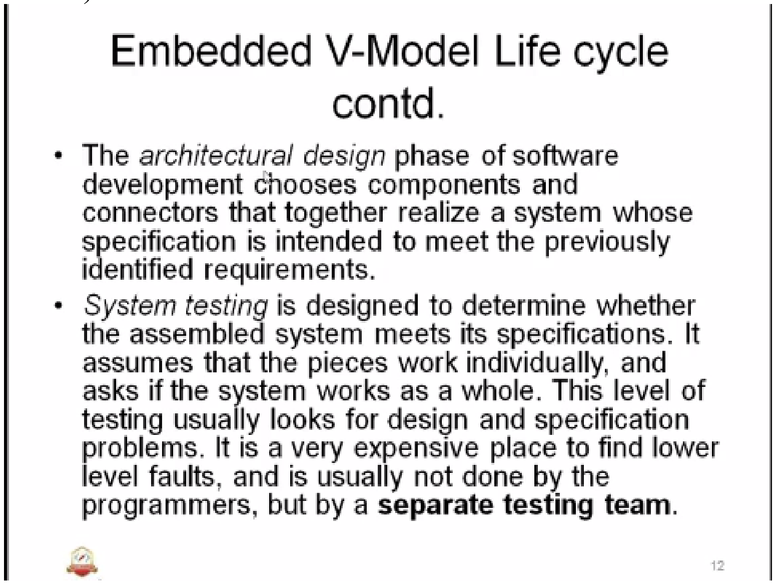
Embedded V-Model Life cycle contd.

- The *requirements analysis* phase of software development captures the customer's needs.
- *Acceptance testing* is designed to determine whether the completed software in fact meets these needs. In other words, acceptance testing probes whether the software does what the users want. Acceptance testing must involve users or other individuals who have strong domain knowledge.

11

Here are some of the points which are to be understood to develop the requirements and completed software parts these are the requirements the probes in software development this is

business prospecting acceptance testing must involve user or other individuals have strong domain knowledge who can basically report the outcome of the customer used for delivery product defined each other of this function testing methodology called integration technology.
(Refer Slide Time: 46:06)

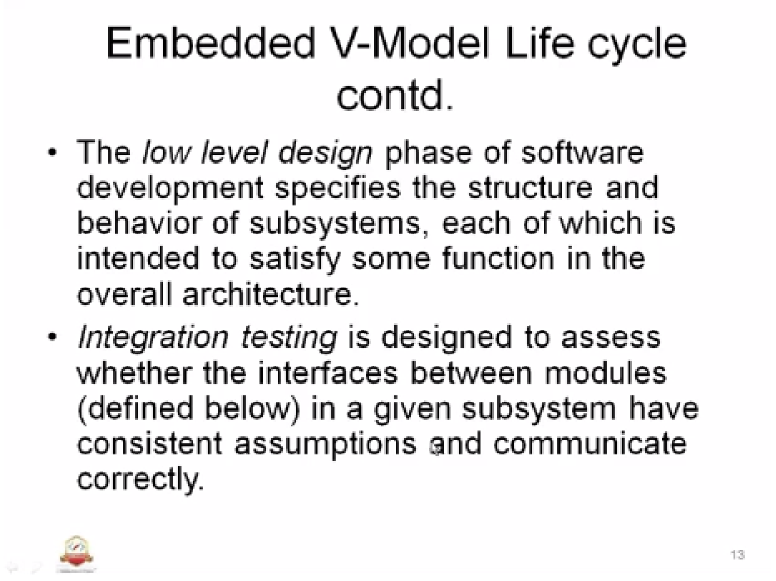


Embedded V-Model Life cycle contd.

- The *architectural design* phase of software development chooses components and connectors that together realize a system whose specification is intended to meet the previously identified requirements.
- *System testing* is designed to determine whether the assembled system meets its specifications. It assumes that the pieces work individually, and asks if the system works as a whole. This level of testing usually looks for design and specification problems. It is a very expensive place to find lower level faults, and is usually not done by the programmers, but by a **separate testing team**.

12

Next one is the architectural design phase of the software development chooses the components and connectors that together realize a system whose specification is intended to meet the previously identified requirements, that means interaction between the various modules, that means we have the high level requirements as the overall interaction.
System testing is design to determine whether the assembled system meets its specifications, it assumes that the pieces work individually, and asks if the system works as a whole, this level of testing usually looks for design and specification problems, it is a very expensive place to find lower level faults, and is usually not done by the programmers, but by a separate testing team.
(Refer Slide Time: 49:44)

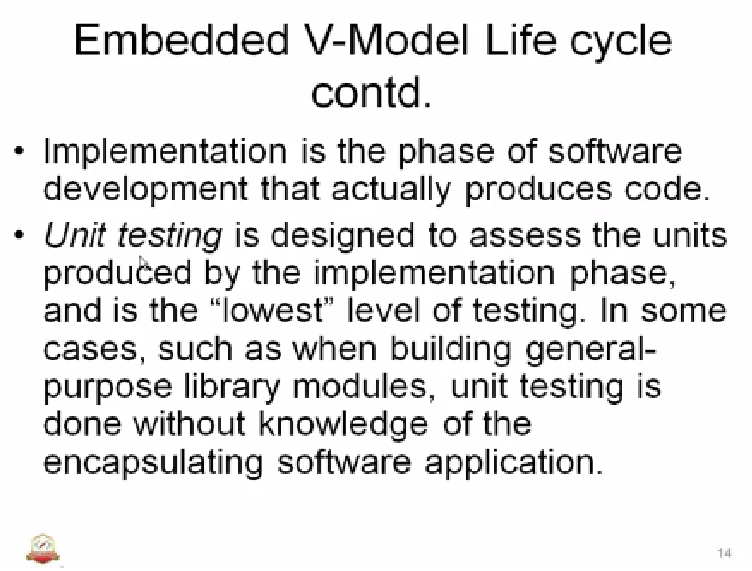


Embedded V-Model Life cycle contd.

- The *low level design* phase of software development specifies the structure and behavior of subsystems, each of which is intended to satisfy some function in the overall architecture.
- *Integration testing* is designed to assess whether the interfaces between modules (defined below) in a given subsystem have consistent assumptions and communicate correctly.

13

So next the low level design phase of software development specifies the structure and behavior of subsystems, each other of which is intended to satisfy some function in the overall architecture, that are we have different sub modules, we have different structure and behavior all will be detailed in it. So each sub memory will have its own defined of the function.
The integration testing is designed to assess whether the interfaces between modules in given subsystem have consistent assumptions and communicate correctly.
(Refer Slide Time: 50:31)



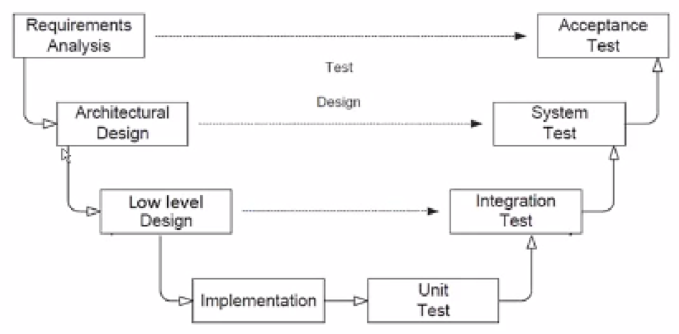
Embedded V-Model Life cycle contd.

- Implementation is the phase of software development that actually produces code.
- *Unit testing* is designed to assess the units produced by the implementation phase, and is the "lowest" level of testing. In some cases, such as when building general-purpose library modules, unit testing is done without knowledge of the encapsulating software application.

14

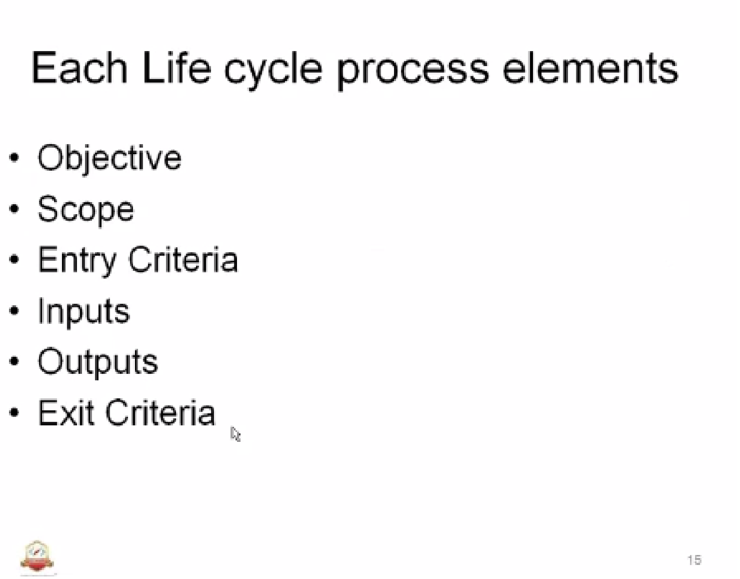
So integration testing is on part which will identify all the low level limit design and it is consistent and they interface each other correctly communicate, and in some cases implementation is the phase of software development that actually produces code.
So, the unit testing is designed to assess the units produced by the implementation phase, and is the lowest level of testing and in some cases, such as when building general purpose library modules, unit testing is done without knowledge of the encapsulating software application.
(Refer Slide Time: 52:10)

Embedded V-Model Life cycle



10

So this will go and an and with different logical process so V life cycle will have a procedure which will be helpful for the next session, so we will go through these criteria in next session of course, so what are those life cycle elements we know that each life cycle will have a criteria in this module, so what is the mandatory things we need to have in the life cycle process.
(Refer Slide Time: 53:00)

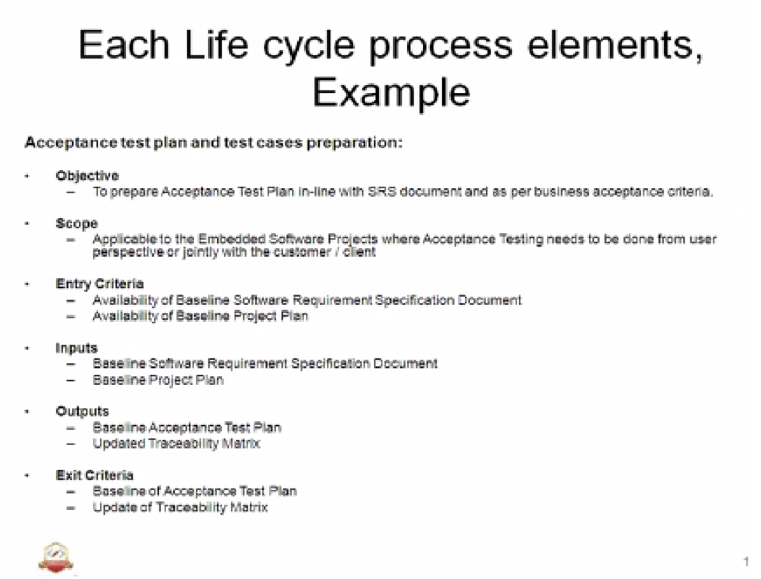


Each Life cycle process elements

- Objective
- Scope
- Entry Criteria
- Inputs
- Outputs
- Exit Criteria

15

There must be objective, scope, entry criteria, inputs, and output, and the exit criteria, so you will take an example for each of the life cycle process.
(Refer Slide Time: 53:20)



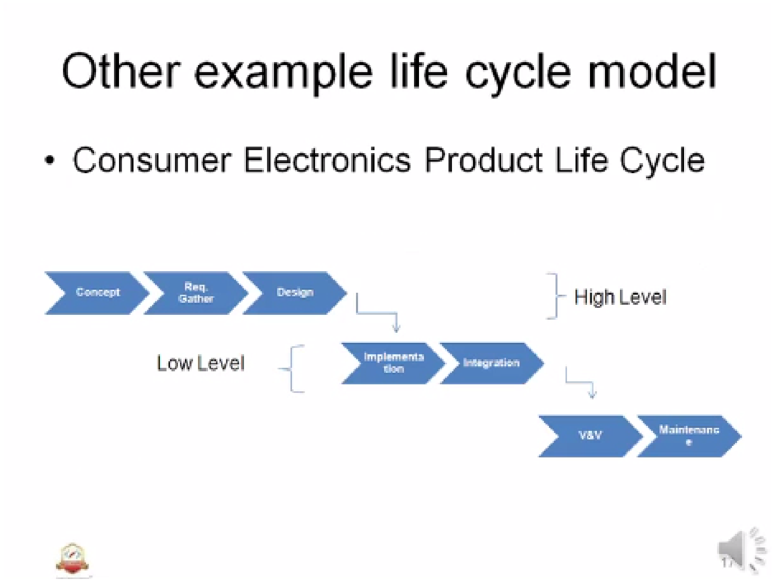
Each Life cycle process elements, Example

Acceptance test plan and test cases preparation:

- **Objective**
  - To prepare Acceptance Test Plan in-line with SRS document and as per business acceptance criteria.
- **Scope**
  - Applicable to the Embedded Software Projects where Acceptance Testing needs to be done from user perspective or jointly with the customer / client
- **Entry Criteria**
  - Availability of Baseline Software Requirement Specification Document
  - Availability of Baseline Project Plan
- **Inputs**
  - Baseline Software Requirement Specification Document
  - Baseline Project Plan
- **Outputs**
  - Baseline Acceptance Test Plan
  - Updated Traceability Matrix
- **Exit Criteria**
  - Baseline of Acceptance Test Plan
  - Update of Traceability Matrix

1

(Refer Slide Time: 53:26)



Other example life cycle model

- Consumer Electronics Product Life Cycle

So basically these are some of the elements that are mandatorily goes on for embedded software life cycle objective, scope, entry criteria, inputs, outputs, exit criteria, so those are required.