SE 3501

SOFTWARE VALIDATION AND VERIFICATION

M. Mangong Clement

SE 3501

SOFTWARE VALIDATION AND VERIFICATION

Welcome!

This course is design to provide a comprehensive check of a software system against its specification and to ensure you understand the process in verifying and validating a software produced.

M. Mangong Clement

I. Basic introduction to Software V&V

❖ Summary

✓ V & V is the process of investigating that a software system **satisfies specifications** and standards and it fulfills the required purpose.

➢ *Satisfies specifications and standards (verification)*

➢ *Fulfills the required purpose (validation)*

I. Basic introduction to Software V&V

❖ Verification

- ✓ Are you building the product right?
- ✓ It verifies whether the developed product fulfills the requirements that we have without bugs.
- ✓ Verification is ***static testing***
- ✓ Activities involved : *Inspections, Reviews, walkthroughs , Desk-checking*

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Peer-reviews

- Easiest and an informal way of reviewing the document or program/ software for the purpose of finding faults.
- Give document or program/software to others to review it so as to give their views about the quality of the product and also expect them to find the fault in the program/software.

I. Basic introduction to Software V&V

- **Verification :** Methods of verification

✓ Peer-reviews

- Activities include the SRS document verification, SDD verification ,and program verification
- The reviewer may prepare a short report on their observation or findings.

I. Basic introduction to Software V&V

- **Verification :** Methods of verification
- ✓ Peer-reviews - Example
- ➢ Do a peer-reviews of your program of a previously written code.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Peer-reviews : Advantages

- You can expect good results without spending any significant resources.
- It is very efficient and significant in its nature.

I. Basic introduction to Software V&V

- **Verification :** Methods of verification

✓ Peer-reviews : disadvantages

- It leads to bad results if the reviewer doesn't have sufficient knowledge.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through Method

- More formal and systematic type of verification method
- The author of the software document presents the document to other persons which can range from 2 to 7.
- Participants are not expected to prepare anything
- The author is responsible for preparing the meeting.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through Method

- Document is distributed to all participants
- At the time of the meeting of the walk-through
- The author introduces the content in order to make them familiar with it and all the participants are free to ask their doubts.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through Method - example

- Do a walk-through demonstration of a program develop.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through Method: Advantages

- It may help us to find potential faults
- It may also be used for sharing documents with others.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through Method: disadvantages

- The author may hide some critical areas and unnecessarily emphasize some specific areas of his/her interest.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

- There are three phases in verification testing

1. Requirement verification
2. Design verification
3. Code verification

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

1. Requirement verification

- The process of verifying and confirming that the requirements are complete, clear and *correct*.
- Before the mobile app goes for design, the testing team verifies business requirements or customer requirements for their correctness and completeness.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

1. Requirement verification

Assignment (1)

- ***How can you test the completeness , clarity, and correctness of a requirement?***

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

2. Design verification

- The process of checking if the design of the software meets the design specification by **providing evidence**.
- The test team checks if layouts , prototypes, navigational charts, architectural designs and database logical models meet the requirements.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

3. Code verification

- The process of checking the code for its completeness, correctness and consistency.
- The testing team checks if construction artifacts such as a source code , user interfaces, and databases physical model of the mobile app meet the design specification.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Walk-through of a mobile App

3. Code verification

- Assignment (2)

How can you test the completeness, correctness and consistency of a source code?

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Inspections

- The most structured and most formal method type
- A team of 3 to 6 participants is constituted which is led by a moderator.
- Group members participate openly,
- After the meeting, a **final report** is prepared after incorporating necessary suggestions by the moderator.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Inspections

- Assignment(3)

How do you prepare a **final report** after a software inspection meeting?

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Inspections : Advantages

- Very effective for finding potential faults or problems in the documents like SRS, SDD,…
- The critical inspections may also help in finding fault and improve the documents which can or in preventing the propagation of fault in the software development life cycle process.

I. Basic introduction to Software V&V

❖ **Verification :** Methods of verification

✓ Inspections : disadvantages

- They take time and require discipline
- It requires more cost and also needs skilled testers.

I. Basic introduction to Software V&V

❖ **Verification :** Other verification methods

✓ Formal verification

- It involves mathematically proving that the requirements are complete and consistent, and that the system will meet the requirements.

I. Basic introduction to Software V&V

- **Verification :** Other verification methods
- ✓ Formal verification

- Assignment (4)

How can we mathematically proof that requirements are complete and consistent, and that the system will meet the functional and non-functional requirements?

I. Basic introduction to Software V&V

❖ **Verification :** Other verification methods

✓ Prototyping verification

- In involves creating a working prototype of the system and testing it to see if it meets the requirements.

I. Basic introduction to Software V&V

- **Verification :** Other verification methods

✓ Acceptance Testing

- It involves testing the system with real users to see if it meets their needs and requirements.

I. Basic introduction to Software V&V

❖ **Verification :** Other verification methods

✓ User feedback

- It involves gathering feedback from the users and incorporating their suggestions and feedback into the requirements.

I. Basic introduction to Software V&V

❖ **Verification :** Other verification methods

✓ Black-box-testing

- It involves testing the system without any knowledge of its internal structure or implementation, to see if it meets the requirements.

I. Basic introduction to Software V&V

❖ **Verification :** Other verification methods

✓ Model-based verification

- It involves creating a model of the system and simulating it to see if it meets the requirements

I. Basic introduction to Software V&V

❖ Validation

- ✓ Are you building the right product?
- ✓ A process of checking whether the software satisfies the customer.
- ✓ Validation is dynamic testing
- ✓ Activities involved : Black box testing, white box testing, unit testing, integration testing.

I. Basic introduction to Software V&V

❖ Requirement Validation

- ✓ A process of checking the requirement defined for development, define the system that the customer really wants.
- ✓ We do requirement validation to check issues related to the requirements.

I. Basic introduction to Software V&V

❖ Requirement Validation

✓ We perform the following checks

- Completeness checks
- Consistency checks
- Validity checks
- Realism checks
- Ambiguity checks
- verifiability

I. Basic introduction to Software V&V

❖ **Validation of a mobile application**

✓ It checks the functionality, usability and performance of the mobile application.

I. Basic introduction to Software V&V

❖ Requirement Validation

- ✓ Output of req. validation is the list of problems and agreed on actions of detected problems

II. Software Testing

❖ What is Software Testing?

- ✓ A process that consists of all test life cycle activities.
- ✓ Life cycle activities like static and dynamic testing, planning, preparation and evaluation of software products.
- ✓ To determine that the software products satisfy customers requirements and are fit for customers use.
- ✓ To find software defects or failure in advance.

II. Software Testing

❖ What is Software Testing?

✓ It can be divided into two parts.

1. Testing as a process
2. Objectives of testing

II. Software Testing

❖ What is Software Testing?

1. Testing as a process

- ✓ Testing is a process
- ✓ Testing is both static and dynamic
- ✓ Planning
- ✓ Preparation
- ✓ Evaluation
- ✓ Software products and related work products

II. Software Testing

❖ What is Software Testing?

2. Objectives of testing

- ✓ Determine that software meets product requirements
- ✓ Determine that software is fit for use
- ✓ Find defects/bugs in software

II. Software Testing

❖ Testing is a process

- ✓ Testing is not a standalone activity
- ✓ It’s a series of activities

II. Software Testing

❖ Testing is a process - Example

- ✓ Software testing is done in all phases of SDLC
- ✓ Testers start with static testing which includes reviewing the documents like software requirements Specification (SRS) , High level Design, and in the later phases of SDLC when testers gets the working build he installs the software and validates that the software functions as per the end user requirements.

II. Software Testing

❖ **Testing is both Static and Dynamic**

✓ Testing both static and dynamic verification

- In initial phases of SDLC static testing is done
- Technical Reviews
- Walkthrough
- Static code analysis

II. Software Testing

❖ Testing is both Static and Dynamic

- ✓ Testing both static and dynamic verification
- Once the build is available to tester, he starts dynamic testing and validates that software meets customer requirements.
- Unit, Integration, system and acceptance testing

II. Software Testing

❖ Planning

✓ Test planning is the most important part of testing

- You need to plan for:

➢ What you want to achieve?

➢ Who will do what?

➢ Time frame of testing

➢ Control the test process

➢ Prepare test summary reports

II. Software Testing

❖ Planning

- ✓ Once the plan has been finalized you need to do test preparation
- ➢ Prepare test cases
- ➢ Prepare test environment
- ➢ Prepare test data

II. Software Testing

❖ Evaluation

✓ While test execution you also need to evaluate the software and make sure that:

- ➢ It meets the exit criteria
- ➢ It is easy to use (Usability testing)
- ➢ Meets end user requirements

II. Software Testing

❖ Software products and related work products

✓ Software testing is not just about testing the software code. Its requires testing all the related documents like

- ➢ Software requirements document(SRS)
- ➢ Design document
- ➢ Quick reference guide
- ➢ Training materials
- ➢ User guides , installation guide

II. Software Testing

❖ Objectives of Testing

1. Determine that software meets end user requirements

- Software Testing also checks products against requirements.
- The design document is reviewed to make sure that it meets requirements and validation of software is done to ensure it meets the design and requirements.
- Testing ensures that product meets its specification and helps stakeholders to make release decisions.

II. Software Testing

❖ Objectives of Testing

2. Determine that software is fit for use

- Software Testing also demonstrates that the software is fit for use
- Testing is done to ensure that software is fit for use for the end users who will be using the software
- It ensures that the software does what end users expect it to do

II. Software Testing

❖ Objectives of Testing

3. Find Defects/Bugs in software

- Software testing detects the defects/bugs in software
- Fixing those defects improves the quality of software
- By doing root-cause analysis of defects found by testing, improves the software development process.

II. Software Testing

❖ Why is testing necessary?

Software Testing is necessary because

- A defect in software can cause harm to person, environment or company
- A defect can cause loss of money, time or business
- Testing improves software quality
- Testing reduces the risk

II. Software Testing

❖ Why do we test something?

- To ensure that it is ok
- Testing is necessary because we are all human beings and human beings make mistakes during development.
- Some human errors do not impact much on our day to day life and can be ignored, however some errors are so severe that they can break the whole system or software.
- In such situations you need to take care that such errors are caught well in advance before deploying the system/software in production environment

II. Software Testing

❖ Software systems context

- An error in your persona blog does not impact any one else
- An error in a business website may put oof the company as it looks unprofessional
- Net banking websites or ATMs should be thoroughly tested to maintain bank credibility
- Air traffic control system is also very critical for testing

II. Software Testing

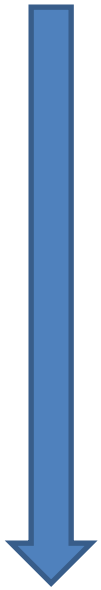
❖ Causes of software defects

- People may make mistake during requirement gathering
- People may make mistake during design
- People may make mistake during coding.

Due to these mistakes they can be flaws in software and these flaws are known as defects or bugs.

II. Software Testing

- **What is cost of defects?**
  - **Requirements**
  - **Design**
  - **Coding**
  - **Testing**
  - **Production**



Cost of finding defects fixing the defects increase over time

II. Software Testing

❖ Role of testing in software development, maintenance and operations

- Testing is important in development and maintenance to identify defects/bugs
- Reduces risk of software failure in operational environment
- Improves quality of software
- Testing is also required as part of contractual agreement or legal requirements (for software which has high risk associated)

II. Software Testing

❖ Testing and Quality

- Testing helps to measure software quality
- Testing provides confidence in software based on number of defects found
- Well designed tests uncover most of the defects in software and if the test pass, it gives more confidence in software quality
- Testing helps to find defects and software quality improves when those defects are fixed.

II. Software Testing

❖ What is software quality?

➢ Quality : The degree to which a component, system or process meets specified requirements and/or user/customers needs and expectations

1. Software quality for developers and testers is that it meets specifications ,is technically good and has few defects.
2. Software quality for other stakeholders may be different- They also need value for money.

II. Software Testing

❖ What is software quality?

➢ Different viewpoints for software quality can be

- Attributes of products
- Fit for use
- Good development processes
- Value for money

II. Software Testing

❖ What is root cause analysis?

1. The finding of real reason for the failure
2. If the software you are testing fails then you do root cause analysis to find the actual cause of that failure.

II. Software Testing

❖ How much testing is enough?

- Exhaustive testing is impossible

- Risk assessment is done to decide how much testing you need to do.

- Testing efforts is based on the risk associated with the different modules of the software.

II. Software Testing

❖ Summary

- Why is testing necessary
- Causes of software defects
- When do defects arise?
- What is the cost of defects?
- Role of testing in software development maintenance and operations
- Testing and Quality
- What is root cause analysis?
- How much testing is enough?

II. Software Testing

❖ The Testing Principles (common for all testing)

1. Testing shows the presence of defect
2. Exhaustive testing is impossible
3. Early Testing
4. Defect clustering
5. Pesticide paradox
6. Testing is context dependent
7. Absence of error fallacy

II. Software Testing

❖ The Testing Principles (common for all testing)

1. Testing shows the presence of defect

- Testing can show that defects are presence but cannot prove that there are no defect in the software
- Testing only reduces the probability of undiscovered defect remaining in the system but even if no defect are found, it is not proof of correctness

II. Software Testing

❖ The Testing Principles (common for all testing)

1. Testing shows the presence of defect

➢ Example: if you are executing test cases and you did not find any defect in many test cases that you executed, you still can not say that there is no bugs in the software, as soon as you find one bug you can say that software is not defect free.

II. Software Testing

❖ The Testing Principles (common for all testing)

2. Exhaustive testing is impossible

➢ Example : How many tests would you need to do to completely test a one-digit numeric field?

II. Software Testing

❖ The Testing Principles (common for all testing)

2. Exhaustive testing is impossible –Example 1

- There are 10 possible valid numeric values
- Invalid Scenario: There are 26 uppercase alpha characters, 26 lower case.
- At least some special and punctuation characters as well as blank value. So there would be approx. 68-70 tests for the example of a one-digit field.

II. Software Testing

❖ The Testing Principles (common for all testing)

2. Exhaustive testing is impossible –Example2

- If you take an example where one screen has 15 input fields. How many test will you need?
- Each having 5 possible values, to test all the valid input combinations you would need (5 to the power 15) 30 517 578 125 test.

II. Software Testing

❖ The Testing Principles (common for all testing)

3. Early testing

- Testing activities should start as early as possible in the software development cycle and should focus on defined objectives
- Early test design and review activities because its cheap to find and fix defects in initial phases.

II. Software Testing

❖ The Testing Principles (common for all testing)

4. Defect clustering

- Most of the testers have noticed that defects tend to cluster
- Defect clustering can happen because an area of the code is complex and tricky
- Changing software and other products tends to cause knock-on defects.
- Testers should use this information when making risk assessment for planning the test.

II. Software Testing

❖ The Testing Principles (common for all testing)

5. Pesticide paradox

- If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new bugs.
- You need to modify the test cases again and again , review them, and more test case to find you bug.

II. Software Testing

❖ The Testing Principles (common for all testing)

6. Testing is context dependent

- Testing is done differently in different contexts
  - For example, safety critical software is tested differently from and e-commerce site
  - Not all software system carry the same number of risk, so different approach is taken to test software in different context.

II. Software Testing

❖ The Testing Principles (common for all testing)

7. Absence-of-error fallacy

- Finding and fixing does not help is the system built is unusable and Does not fulfill the users needs and expectations.
- The customers who buys the software do not bother about number of defects until software directly affects them and is unstable to use.
- They are more interested in software fulfilling their requirements and does what they want.

II. Software Testing

❖ The Testing Principles (common for all testing)

1. Testing shows the presence of defect
2. Exhaustive testing is impossible
3. Early Testing
4. Defect clustering
5. Pesticide paradox
6. Testing is context dependent
7. Absence of error fallacy

II. Software Testing

❖ Fundamental Test process

- It starts through the planning and continues till the test closure.
- Test planning should consider the time spent on planning and test closure activities
- Project and test plans should include time spent on test planning, test designing , preparing for execution and evaluating status.

II. Software Testing

- **Fundamental Test Process (5 steps)**

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Planning and control
2. Analysis and Design
3. Implementation and execution
4. Evaluating exit criteria and reporting
5. Test closure activities

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control

➢ Before setting **goals and objectives for** testing in the planning phase, we need to

✓ Understand the **goals** and **objectives** of {customers, stakeholders and the project}

✓ Understand the ***risk*** which testing is intended to address.

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control

➢ Major task of test Planning

✓ Identify the objectives of testing based on the scope and risk of the project

✓ Determine the test approach

✓ Implement the required test resources

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control

➢ Major task of test Planning

✓ Scheduling test analysis and design tasks, test implementation , execution and evaluation.

✓ Determine the exit criteria

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control - Major task in planning

➢ Understand the objectives of testing based on the scope and risks of project.

- Decide which components, systems or other products are in the test scope

- Decide the business, product, project and technical risks which need to be address

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control- *Major task in planning(1)*

a. Understand the objectives of testing based on the scope and risks of project.

➢ D***ecide the objectives of testing(Why testing?)***

1. To ***uncover*** *defects*
2. To **verify** that software *meets requirement*.
3. To ***demonstrate*** if the software is *fit for use*.

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control- *Major task in planning(1)*

a. Understand the objectives of testing based on the scope and risks of project.

Testing Objectives{ ***project Scope, Project Risk***}

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control- *Major task in planning(2)*

b. Determine the test approach

1. What needs to be tested?
2. **How testing** will be carried out?
3. What test technique will be used?
4. To what extend of test coverage is required?
5. Who is involved and when?
6. Decide test deliverable's to be produced(Test cases, test data)

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control- *Major task in planning(3)*

c. Implement the test policy and/or test strategy

- If the organization test policy and strategy exists, then during planning ensure that testing adheres to those policy/ strategy.

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control- *Major task in planning(4)*

d. Determine the required test resources

- Define the required resources for testing like testers , hardware and software etc.

II. Software Testing

❖ Fundamental Test process (5 steps)

1.Test Planning and control-Major task in planning(5)

- **Scheduling test analysis and design tasks, test implementation, execution and evaluation**
- You need to prepare the schedule for all the tasks so that tracking can be done and the process is captured.

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control - Major task in planning

➢ **Determine the exit criteria**

- Criteria set to find out when to finish testing.
- The task that must be completed for the test level before we can exit the test phase

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control- Test control

- After test planning we need to measure and control the process
- It is an ongoing activity of comparing actual progress against plan
- Test control reports that status of test progress including any deviations form the actual plan
- Test control monitors the testing throughout the project

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control-

Major task Test control

- Measure and analyze results of reviews and testing
- Monitor and document test progress coverage and exit criteria
- Provide information on testing
- Initiate corrective actions
- Make release decisions

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control- Major task Test control

➢ Measure and analyze results of reviews and testing

- Track test pass/fail percentage
- Track tests remaining

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control- Major task Test control

➢ Monitor and document test progress coverage and exit criteria

- Track how many tests executed
- What is the testing outcome(number of test passed/failed)
- Risk assessment of test outcome

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control- Major task Test control

➢ Provide information on testing

- Provide regular test progress reports to stakeholders.

➢ **Initiate corrective actions**

- Putting more efforts in debugging
- Prioritizing defects

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Test Planning and control- Major task Test control

➢ Make release decisions

- Based on information gathered during testing decision are made – like to continue testing, stop testing, release software or not to release.

II. Software Testing

❖ Fundamental Test process (5 steps)

2. Test Analysis and Design

➢ Task

- Review the test basis
- Identifying the test conditions
- Designing the test
- Evaluate testability of the requirements and system
- Designing the test environment

II. Software Testing

- Fundamental Test process (5 steps)

3. Test implementation and execution

- It takes conditions and make them into test cases
- Build test environment where test execution needs to be done.

II. Software Testing

- Fundamental Test process (5 steps)

4. Evaluating exit criteria and reporting

- An activity where test execution is assessed against the define objectives
- This should be done in every test level
- The criteria is set based on risk assessment

II. Software Testing

❖ Fundamental Test process (5 steps)

4. Evaluating exit criteria and reporting

➢ Major task

- Check the test logs against the exit criteria specified in the plan.
- Assess if more tests are needed or if the exit criteria specified should be changed.
- Writing a test summary report for stakeholders

II. Software Testing

- Fundamental Test process (5 steps)

5. Test closure activities

- Collect data from completed test activities to consolidate experience.

II. Software Testing

❖ Fundamental Test process (5 steps)

5. Test closure activities

➢ Major Tasks

- Check which planned deliverables have been delivered.
- Finalize and archive testware for later use
- Handover testware to the maintenance organization
- Evaluating of the testing and analysis lessons learned for future releases and projects.

II. Software Testing

❖ Fundamental Test process (5 steps)

5. Test closure activities

➢ Major Tasks

- Check which planned deliverables have been delivered.
- Finalize and archive testware for later use
- Handover testware to the maintenance organization

II. Software Testing

❖ Fundamental Test process (5 steps)

1. Planning and control
2. Analysis and Design
3. Implementation and execution
4. Evaluating exit criteria and reporting
5. Test closure activities

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

i. Clear objectives

ii. Independent testing

iii. Feedback on defects

iv. Clear and courteous communication

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

i. Clear objectives

ii. Independent testing

iii. Feedback on defects

iv. Clear and courteous communication

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

❑ Independent testing

✓ Mindset is different if you are building or developing something

✓ Mindset is different if you are testing

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

❑ Independent testing

✓ Work positively to build something

✓ Work to look for issues/defects in the product.

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

❑ Independent testing - levels

- ✓ Testing done by developers
- ✓ Another person in team (another developer)
- ✓ Another person form difference organization
- ✓ Testing done by different organizations

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

❑ Feedback on defects

✓ Tips for reporting defects or failures

- Communicate defects/failures in neutral way
- Explain that by finding the issue earlier we can deliver better software.

II. Software Testing

❖ The Psychology of testing

➢ Factors that influence **testing**

❑ **Clear and courteous communication**

✓ Start with collaboration not battles

✓ Improve interpersonal and communication skills

II. Software Testing

❖ Test Levels in testing through the SDLC

- ❑ Component / unit testing
- ❑ Integration testing
- ❑ System testing
- ❑ Acceptance testing

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- The smallest testable part of the software system
- Done to verify that the lowest independent entities in any software are working fine.

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- The smallest part is isolated form the remainder code and tested to determine whether it works correctly
- Unverified blocks can be replace by STUBS and DRIVERS

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- A STUBS is called form the software components

✓ Suppose module A calls functions from module B and C which are not yet ready

✓ The developer will write a dummy module which simulate B and C and return values to module A

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- A DRIVERS call a component to be tested
- Suppose you have module B and C ready, but module A which calls module B and C is not ready.

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- The developer writes a dummy piece of code for module A and which takes case of the call for module A
- This dummy piece of code is known as DRIVERS

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Component / unit testing

- This type of testing may include testing of functional and non functional characteristics
- It is typically done by developers
- Defect are typically fixed as soon as they are found
- No formal defect logging process is followed

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing

- Test interfaces between components
- Testers concentrate only on the integration
- Testers should build integration test in the order required for most effective integration testing

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing

- It test interaction to different parts of the system like, file system, system hardware, OS, interfaces between other systems

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing -Levels

- Component integration testing
- System integration testing

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing -Levels

- Component integration testing

✓ Tests the interactions between software components

✓ Done after component/unit testing

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing -Levels

- system integration testing

✓ Tests the interaction between different systems

✓ Done mostly after system testing is complete

✓ Done to identify the cross-platform issues which may arise after integrating the system to others

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing - Approach

- Big bang integration testing
- Incremental integration testing

✓ Top-down approach

✓ Button-up approach

✓ Functional incremental approach

II. Software Testing

❖ Test Levels in testing through the SDLC

❑ Integration testing - Approach

- Big bang integration testing

✓ Integrate all components or system are integrated simultaneously, after which everything is tested as a whole

Welcome!

This course is design to provide a comprehensive check of a software system against its specification and to ensure you understand the process in verifying and validating a software produced.

QUESTIONS