Tutorial 3

COMP 354- Fall 23

Apekshaba Gohil Anam A. Shaikh Marzieh Adeli

apekshabagohil.it@gmail.com anams403@gmail.com marzieh.adeli@gmail.com

Quality Control

1. Technical Reviews

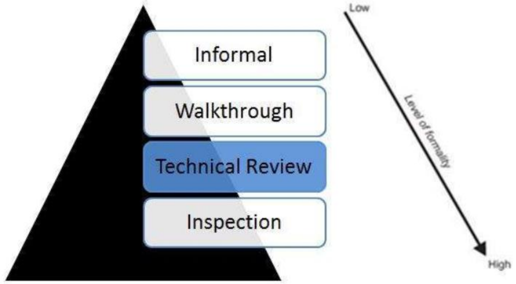
2. Component Testing

3. Integration Testing (bottom-up or top-down)

1. Technical Reviews

- Technical reviews **provide status and feedback on the products under review and the on on-going activities of a project.**

- A Technical review is a static white-box testing technique which is conducted to spot the defects early in the life cycle that cannot be detected by black box testing techniques.



Technical Review

- Technical Reviews are documented.

- It uses a defect detection process that has peers and technical specialist as part of the review process.

- It is usually led by trained moderator who is NOT the author.

- The report is prepared with the list of issues that needs to be addressed.

Component Testing

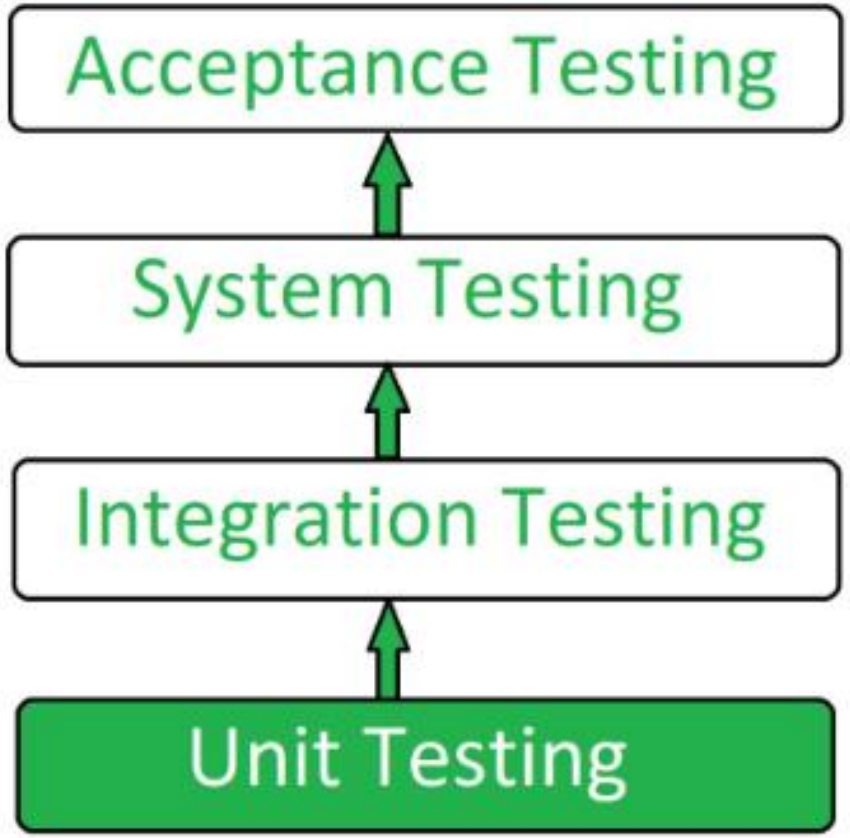
- The primary objective of component testing is **to validate the behavior of the individual component, as specified in the requirements document**.

- Components are the part of system application. It breaks the complexity of software into manageable parts.

- E.g. Model , View and Controller are individual components.

- To perform components testing on each of the components in our application we will perform the individual testing of the program based on Unit testing.

Unit Testing



Unit Testing

- **Unit Testing** is a software testing technique by means of which individual units of software i.e. group of computer program modules, usage procedures, and operating procedures are tested to determine whether they are suitable for use or not.
- Unit Testing is defined as a type of software testing where individual components of a software are tested.
- Unit Testing is performed by the developers during the development phase.
- It is the most lowest level of testing and performed in the early stages of software development.
- E.g. Unit Testing Tool is Junit in JAVA application.a

Objective of Unit Testing

1. To isolate a section of code.
2. To verify the correctness of the code.
3. To test every function and procedure.
4. To fix bugs early in the development cycle and to save costs.
5. To help the developers to understand the code base and enable them to make changes quickly.
6. To help with code reuse.

Integration Testing

- **Integration testing** is the process of testing the interface between two software units or modules.

- It focuses on determining the correctness of the interface.

- The purpose of integration testing is to expose faults in the interaction between integrated units.

Types of Integration Testing

1. Big-Bang Integration Testing

2. Bottom-Up Integration Testing

3. Top-Down Integration Testing

4. Mixed Integration Testing

1. Big-Bang Integration Testing

- It is the simplest integration testing approach, where all the modules are combined and the functionality is verified after the completion of individual module testing.

- In simple words, all the modules of the system are simply put together and tested.

- This approach is practicable only for very small systems.

- If an error is found during the integration testing, it is very difficult to localize the error as the error may potentially belong to any of the modules being integrated.

2. Bottom-Up Integration Testing

- Each module at lower levels is tested with higher modules until all modules are tested.
- The primary purpose of this integration testing is that each subsystem tests the interfaces among various modules making up the subsystem.
- This integration testing uses test drivers to drive and pass appropriate data to the lower-level modules.
- In this approach, several disjoint sub-systems can be tested together.

3. Top-Down Integration Testing

- First, high-level modules are tested and then low-level modules and finally integrating the low-level modules to a high level to ensure the system is working as intended.

- Top-down integration testing technique is used in order to simulate the behavior of the lower-level modules that are not yet integrated.

- No test-drivers are required.

- This technique is more stable and accurate at the aggregate level.

4. Mixed Integration Testing

- A mixed integration testing is also called sandwiched integration testing.

- Follows a combination of top down and bottom-up testing approaches.

- This sandwich or mixed approach overcomes this shortcoming of the top-down and bottom-up approaches.

- Very useful for very large projects having several sub-projects.

Quality Control

1. **Revision History**

2. **Technical Review**
    1. Preparations
    2. Role
    3. Issues
        1. Issue – Reviewed By, Severity Level, Results
        2. Review Metrics – Preparation, Assessment, Rework
        3. Review Outcome – Vote, How, What

Quality Control

3. Unit Tests

-Cover each and every unit/ module within your software product and write the appropriate test cases for the same.

Try to do it for each and every module of the application.

4. Integration Testing

- Explain which approach of integration testing have you adopted and why is it best according to you.
  1. Purpose
  2. Components being tested
  3. Steps to perform the test

Quality Control

5. Changes to the original design

- After performing technical reviews and testing which areas of the design requires changes in your Concordia Cart App.
- MVC Diagram explains the best how the individual components were integrated and how can they interface and interact.

6. **Contributions**

References

- https://www.worldscientific.com/doi/10.1142/9789813221888_0002#:~:text=%E2%80%9CA%20software%20component%20is%20a,composition%20by%20third%20parties.%E2%80%9D

- https://www.geeksforgeeks.org/difference-between-component-and-unit-testing/#:~:text=Component%20Testing%20involves%20testing%20each,performed%20by%20the%20testing%20team

- https://www.geeksforgeeks.org/software-engineering-integration-testing/?ref=lbp

- https://www.geeksforgeeks.org/unit-testing-software-testing/