Agile testing for Autosar Product

Name of Testers:
1. Anupam
2. Asha
3. Harshvardhan
4. Hrushikesh
5. Manish
6. Krishna
7. Tahir

Version: 1.7
Created: 05/28/2024
Last Updated: 05/28/2024
Status: Complete

Revision Sheet

Version	Date	Author	Description of Change
1.0	05/28/2024	Krishna	Initial Draft: Sprint Planning
1.1	05/28/2024	Asha	Revised Draft: Test-Driven Development (TDD)
1.2	05/28/2024	Tahir	Revised Draft: Continuous Integration (CI)
1.3	05/28/2024	HarshVardhan	Revised Draft: Daily Stand-Ups
1.4	05/28/2024	Anupam	Revised Draft: Incremental Testing
1.5	05/28/2024	Manish	Revised Draft: System Testing
1.6	05/28/2024	Hrushi	Revised Draft: Review and Retrospective
1.7	05/28/2024	Group 1	Final Draft: Example User Story Testing Flow and Tools and Frameworks

Table of Contents

7. System Testing
   - End-to-End Testing
   - Functional Safety Testing

8. Review and Retrospective
   - Test Coverage Analysis
   - Sprint Retrospective

9. Example User Story Testing Flow
   - User Story: Implement Diagnostic Communication
     - Requirement
     - Test Cases
     - TDD
     - Integration Tests
     - System Tests

10. Conclusion

Introduction

Agile testing for an AUTOSAR (Automotive Open System Architecture) product involves integrating testing activities into the agile development lifecycle, focusing on iterative and incremental delivery.

Here's a demonstration of how agile testing can be applied to an AUTOSAR product:

1. Sprint Planning

- Requirement Analysis: Review AUTOSAR requirements, functional safety requirements, and other system constraints.

- Test Planning: Define testing goals for the sprint. Identify which AUTOSAR modules (e.g., BSW, RTE, SW-C) will be developed and tested.

- User Stories and Test Cases: Break down requirements into user stories. Create corresponding test cases for each user story, focusing on both unit tests and integration tests.

2. Test-Driven Development (TDD)

- Writing Tests First: For each user story, write test cases before developing the code. For AUTOSAR, this could include:

  - Unit Tests: For individual SW-C (software components).

  - Integration Tests: For interactions between different AUTOSAR layers, such as communication between SW-Cs and BSW modules.

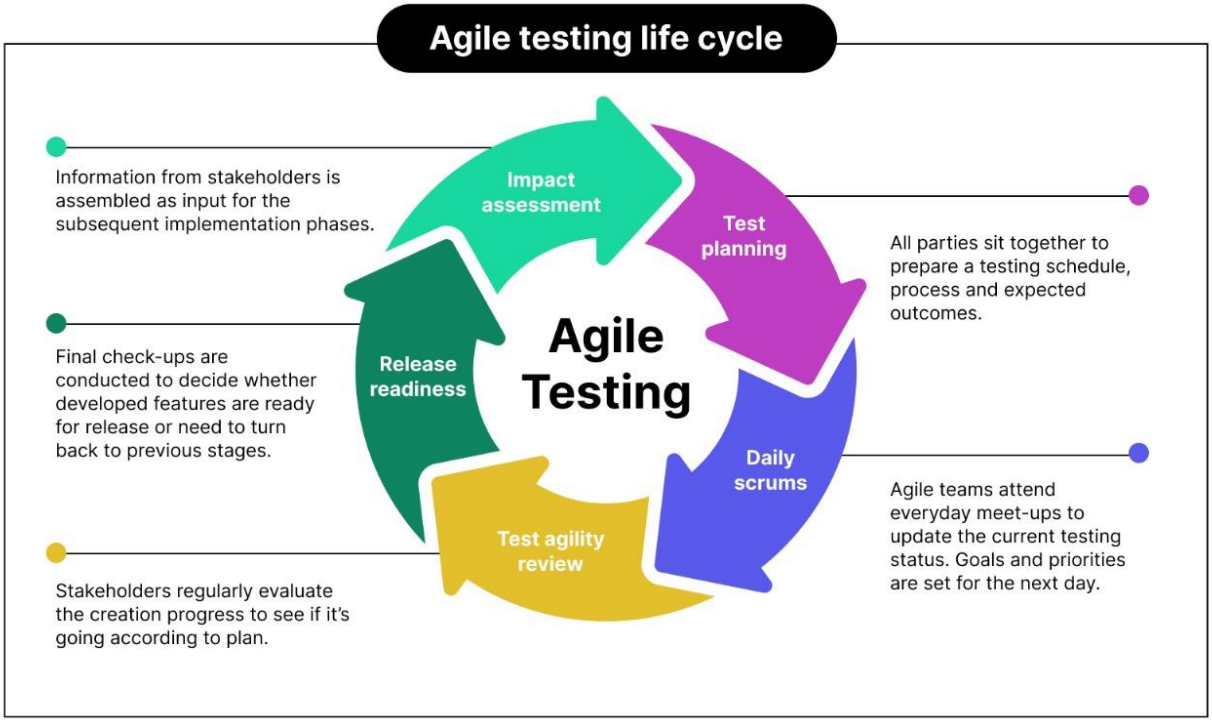
- Example: If a user story involves creating a CAN communication module, write tests to verify message transmission, reception, and error handling.

3. Continuous Integration (CI)

- Automated Builds and Tests: Set up a CI pipeline to automatically build the AUTOSAR software and run tests on every code commit.

- Tool Integration: Use tools like Jenkins, GitLab CI, or Azure DevOps to automate the process. Integrate with AUTOSAR tools like Vector CANoe for simulation and testing.

- Static Analysis: Integrate tools for static code analysis (e.g., Polyspace, QAC) to ensure compliance with coding standards and detect potential issues early.



4. Daily Stand-Ups

- Progress Tracking: Discuss testing progress, any issues encountered, and adjust the plan as necessary.

- Defect Reporting: Report and prioritize defects found during testing. Use a defect tracking tool (e.g., JIRA) to manage them.

5. Incremental Testing

- Unit Testing: Developers write unit tests for their components. Use frameworks like Google Test or VectorCAST.

- Integration Testing: As modules are integrated, perform integration tests to verify communication and interaction. Use tools like dSPACE or ETAS for hardware-in-the-loop (HIL) testing.

- Example: If integrating a new sensor input module, test how it interacts with existing control modules in both simulated and real environments.

6. System Testing

- End-to-End Testing: Perform system-level tests to validate the entire AUTOSAR stack. This includes testing in a simulated vehicle environment and on actual hardware.

- Functional Safety Testing: Conduct tests to ensure the system meets safety requirements as per ISO 26262.

7. Review and Retrospective

- Test Coverage Analysis: Review test coverage reports to ensure all parts of the code are adequately tested.

- Sprint Retrospective: Discuss what went well in the testing process and what can be improved. Adjust the testing strategy accordingly.

Example User Story Testing Flow

User Story: Implement Diagnostic Communication

1.Requirement:

The system must support diagnostic communication over CAN.

2. Test Cases:

- Verify that diagnostic messages can be received and decoded.

- Verify that the system can respond with appropriate diagnostic data.

- Test handling of erroneous diagnostic messages.

3. TDD:

- Write unit tests for the diagnostic module functions.

- Develop the diagnostic module.

- Run unit tests to validate functionality.

4. Integration Tests:

- Integrate the diagnostic module with the CAN communication stack.
- Test diagnostic communication in a simulated environment using Vector CANoe.

5. System Tests:

- Test the complete diagnostic communication process on HIL setup.
- Validate against real vehicle scenarios.

Conclusion

By incorporating agile testing practices, AUTOSAR product development can achieve higher quality, quicker feedback cycles, and better alignment with evolving requirements.

THANK YOU