COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD CAMPUS

Assignment 01

Software Testing

Submitted by:

Arfah Ali FA21-BSE-080-6A

Submitted To:

Sir Mukhtiar Zamin

Table of Contents

Introduction

The concept of a test plan is fundamental in the field of software testing. It outlines the strategy, resources, scope, and schedule for the testing activities to ensure that a software product meets its requirements and is free of defects. Over time, the format and complexity of test plans have evolved, influenced by changes in software development methodologies, technological advancements, and industry standards. This assignment aims to explore the evolution of test plans, from their informal beginnings to the adoption of structured templates and modern practices such as Continuous Integration (CI) and Continuous Deployment (CD) on platforms like GitHub.

Early Beginnings: Informal Test Plans (1990s)

In the early days of software development, testing was often an ad-hoc process. Test plans were informal and undocumented, relying heavily on the knowledge and expertise of individual testers. These plans lacked standardization and were prone to inconsistencies, making it difficult to ensure comprehensive test coverage. Informal test plans typically included basic elements such as:

Test Plan for Desktop Calculator Application (1990s)

Objective: Ensure basic arithmetic operations (addition, subtraction, multiplication, division) work correctly.

Scope: Manual testing of calculator functions on a Windows 3.1 environment.

Resources:

- Tester: John Doe
- Environment: Windows 3.1
- Tools: Basic calculator application

Schedule:

- Start Date: 01/01/1995
- End Date: 01/15/1995

Test Cases:

TC ID	Description	Input Data	Expected Result	Actual Result	Status
TC001	Verify addition of two positive numbers	5 + 3	8	8	Pass

TC002	Verify subtraction of two positive numbers	10 - 4	6	6	Pass
TC003	Verify multiplication of two positive numbers	7 * 6	42	42	Pass
TC004	Verify division of two positive numbers	20 / 5	4	4	Pass

Informal Test Plan Template (1990s)

- **Title**: Test Plan for [Application Name]
- **Objective**: [Objective of the test plan]
- **Scope**: [Scope of testing]
- **Resources**:
  - **Tester**: [Tester Name]
  - **Environment**: [Testing environment]
  - **Tools**: [Tools used]
- **Schedule**:
  - **Start Date**: [Start Date]
  - **End Date**: [End Date]
- **Test Cases**: [Detailed test cases with TC ID, description, input data, expected result, actual result, status]

IEEE 829 Standard for Test Documentation (2000s)

As software development became more complex, the need for standardized test plans became evident. The IEEE 829 standard, also known as the Standard for Software Test Documentation, was introduced to provide a comprehensive framework for creating test plans. The IEEE 829 test plan template includes the following sections:

Test Plan for Online Banking System (2000s)

Test Plan Identifier: TP-OBS-2005

Introduction: This test plan outlines the testing activities for the online banking system. The objective is to ensure the functionality, security, and performance of the system. Testing will cover user login, fund transfer, account statement generation, and security features.

Test Items:

- User login module
- Fund transfer module

- Account statement generation module
- Security features

Features to be Tested:

- User authentication
- Fund transfer between accounts
- Generation of account statements
- Data encryption and session management

Features Not to be Tested:

- Non-core banking services (e.g., loan applications, investment services)

Approach: The overall approach includes functional testing, security testing, and performance testing. Testing will be manual and automated where applicable.

Item Pass/Fail Criteria: Test items will pass if they meet the expected results outlined in the test cases. Any deviations will be logged as defects.

Suspension Criteria and Resumption Requirements: Testing will be suspended if critical defects are found that block further testing. Testing will resume once the defects are resolved.

Test Deliverables:

- Test plan document
- Test cases
- Test execution reports
- Defect logs

Testing Tasks:

- Develop test cases
- Execute test cases
- Log defects
- Re-test resolved defects
- Generate test reports

Environmental Needs:

- Web-based application environment
- Internet Explorer 6
- Test data for user accounts and transactions

Responsibilities:

- Test Manager: Alice Brown
- Test Engineer: Jane Smith

- Security Analyst: Bob White

Staffing and Training Needs:

- Test engineers trained in banking domain testing
- Security analysts trained in data encryption and security testing

Schedule:

- Start Date: 05/01/2005
- End Date: 06/30/2005

Risks and Contingencies:

- Risk: Delayed delivery of test environment
- Contingency: Use backup environment

Approvals:

- Test Manager: [Signature]
- Project Manager: [Signature]

Test Cases:

TC ID	Description	Input Data	Expected Result	Actual Result	Status
TC101	Verify user login with valid credentials	Username: user1, Password: pass123	Login successful, user dashboard displayed	Login successful	Pass
TC102	Verify fund transfer between accounts	From Account: 12345, To Account: 67890, Amount: $100	Transfer successful, balance updated	Transfer successful	Pass
TC103	Verify generation of account statements	Account: 12345, Date Range: 01/01/2005 - 05/01/2005	Statement generated with correct details	Statement generated	Pass
TC104	Verify security features such as encryption and session management	N/A	Data encrypted, session expires after inactivity	Data encrypted, session expires	Pass

IEEE 829 Test Plan Template

- **Test Plan Identifier**: [Unique Identifier]
- **Introduction**: [Overview of the test plan, including scope, objectives, and assumptions]
- **Test Items**: [List of software items to be tested]

- **Features to be Tested**: [Specific features and functionalities to be tested]
- **Features Not to be Tested**: [Any features excluded from testing]
- **Approach**: [Overall approach and strategy for testing]
- **Item Pass/Fail Criteria**: [Criteria for determining whether test items pass or fail]
- **Suspension Criteria and Resumption Requirements**: [Conditions for suspending and resuming testing]
- **Test Deliverables**: [Documents and other deliverables related to testing]
- **Testing Tasks**: [Specific tasks and activities involved in testing]
- **Environmental Needs**: [Hardware, software, and other environmental requirements]
- **Responsibilities**: [Roles and responsibilities of team members]
- **Staffing and Training Needs**: [Staffing requirements and any necessary training]
- **Schedule**: [Detailed schedule for testing activities]
- **Risks and Contingencies**: [Potential risks and contingency plans]
- **Approvals**: [Sign-offs and approvals required for the test plan]

Agile and Modern Test Plans (2010s)

With the advent of Agile methodologies, the approach to test planning shifted. Agile emphasizes flexibility, collaboration, and iterative development, leading to the creation of lightweight and dynamic test plans. These plans are often integrated into user stories and sprints, with a focus on continuous feedback and improvement. Key characteristics of Agile test plans include:

Agile and Modern Test Plans (2010s)

Test Plan for Social Media App (2010s)

Objective: Ensure the functionality and performance of core social media features, such as posting, commenting, and liking.

Scope: Testing includes user registration, post creation, commenting, liking, and notifications.

User Stories:

- As a user, I want to register with my email and password so that I can create an account.
- As a user, I want to create posts with text and images so that I can share content with others.
- As a user, I want to comment on posts so that I can engage with other users.
- As a user, I want to like and unlike posts so that I can express my preferences.
- As a user, I want to receive notifications for new comments and likes so that I am informed about interactions.

Iteration-Based Planning: Test plans are created for each sprint, focusing on the user stories planned for that iteration.

Collaboration: Close collaboration between developers, testers, and stakeholders to ensure that the test plans align with the development goals.

Continuous Integration: Frequent integration and testing of code changes to catch issues early.

Automation: Emphasis on automated testing to ensure quick feedback.

Resources:

- Tester: Emily Brown
- Environment: Mobile application, Android and iOS
- Tools: Social media app

Schedule:

- Start Date: 03/01/2015
- End Date: 04/30/2015

Test Cases:

TC ID	Description	Input Data	Expected Result	Actual Result	Status
TC201	Verify user registration with email verification	Email: test@example.com, Password: securePass1	Registration successful, verification email sent	Registration successful	Pass
TC202	Verify post creation with text and images	Post: "Hello world!", Image: test.jpg	Post created successfully	Post created	Pass
TC203	Verify commenting on posts	Post ID: 123, Comment: "Nice!"	Comment added to post	Comment added	Pass
TC204	Verify liking and unliking posts	Post ID: 123	Post liked, like count updated, post unliked, like count updated	Post liked, post unliked	Pass
TC205	Verify notifications for new comments and likes	Post ID: 123, Like, Comment	Notification received	Notification received	Pass

Agile Test Plan Template (2010s)

- **Title**: Test Plan for [Application Name]
- **Objective**: [Objective of the test plan]
- **Scope**: [Scope of testing]
- **User Stories**: [User stories aligned with the test plan]
- **Iteration-Based Planning**: [Test plans for each iteration or sprint]
- **Collaboration**: [Collaboration strategy between developers, testers, and stakeholders]
- **Continuous Integration**: [Approach for frequent integration and testing]

- **Automation**: [Emphasis on automated testing]
- **Resources**:
  - **Tester**: [Tester Name]
  - **Environment**: [Testing environment]
  - **Tools**: [Tools used]
- **Schedule**:
  - **Start Date**: [Start Date]
  - **End Date**: [End Date]
- **Test Cases**:
  - **TC ID**: [ID]
  - **Description**: [Description]
  - **Input Data**: [Input data]
  - **Expected Result**: [Expected result]
  - **Actual Result**: [Actual result]
  - **Status**: [Status]

Continuous Integration and Continuous Deployment (CI/CD) on GitHub (2015-2020s)

The rise of DevOps practices has further transformed test planning. CI/CD pipelines automate the process of building, testing, and deploying software, enabling rapid and reliable delivery. Test plans in a CI/CD context are integrated into the pipeline configuration and managed through version control systems like GitHub. Key elements of CI/CD test plans include:

Test Plan for Project Management Tool (2015-2020s)

Objective: Ensure the continuous integration and deployment of the project management tool with automated testing.

Scope: Testing includes user authentication, project creation, task management, and integration with third-party tools.

Pipeline Configuration: Test plans are defined as part of the CI/CD pipeline configuration using tools like GitHub Actions or Jenkins.

Automated Tests: Extensive use of automated tests (unit, integration, end-to-end) to ensure code quality.

Continuous Monitoring: Monitoring and reporting tools to track test results and performance metrics.

Version Control: Test plans and related scripts are version-controlled in repositories.

Infrastructure as Code (IaC): Infrastructure and test environments are defined and managed as code.

Resources:

- Tester: Michael Green
- Environment: Cloud-based application, GitHub, Jenkins
- Tools: Project management tool, CI/CD pipeline

Schedule:

- Start Date: 01/01/2018
- End Date: 12/31/2018

Test Cases:

TC ID	Description	Input Data	Expected Result	Actual Result	Status
TC301	Verify user authentication with OAuth	OAuth Provider: Google	Authentication successful, user dashboard displayed	Authentication successful	Pass
TC302	Verify project creation and editing	Project Name: "New Project"	Project created successfully, project details editable	Project created, editable	Pass
TC303	Verify task creation, assignment, and completion	Task: "Finish report", Assignee: user2	Task created, assigned, and marked as completed	Task created, assigned, completed	Pass
TC304	Verify integration with third-party tools like Slack and Trello	Integration Settings: Enabled	Notifications sent to Slack, tasks synced with Trello	Notifications sent, tasks synced	Pass

CI/CD Test Plan Template (2015-2020s)

- **Title**: Test Plan for [Application Name]
- **Objective**: [Objective of the test plan]
- **Scope**: [Scope of testing]
- **Pipeline Configuration**: [CI/CD pipeline configuration details]
- **Automated Tests**: [Details of automated tests used]
- **Continuous Monitoring**: [Monitoring and reporting tools]
- **Version Control**: [Version control strategy]
- **Infrastructure as Code (IaC)**: [Infrastructure management as code]
- **Resources**:
    - **Tester**: [Tester Name]
    - **Environment**: [Testing environment]
    - **Tools**: [Tools used]
- **Schedule**:

  - **Start Date**: [Start Date]
  - **End Date**: [End Date]
- **Test Cases**:
  - **TC ID**: [ID]
  - **Description**: [Description]
  - **Input Data**: [Input data]
  - **Expected Result**: [Expected result]
  - **Actual Result**: [Actual result]
  - **Status**: [Status]

Other Evolved Test Plans

In addition to the above-mentioned test plans, several other templates and frameworks have emerged over time, each catering to specific needs and contexts:

- **V-Model Test Plans:** Based on the V-Model development lifecycle, these plans emphasize the parallel development and testing phases, ensuring each development activity is matched with a corresponding testing activity.
- **Risk-Based Test Plans:** Focus on identifying and prioritizing risks, ensuring critical functionalities are tested first.
- **Behavior-Driven Development (BDD):** Uses natural language to define test scenarios, promoting better understanding and collaboration among team members.

Summary

The evolution of test plans from informal checklists to comprehensive, automated, and integrated documents has significantly improved software quality. Early test plans were simple and manual, lacking formal documentation and detailed test cases. The introduction of standards like IEEE 829 brought structure and comprehensiveness, while Agile methodologies added flexibility and continuous feedback. The integration of CI/CD pipelines revolutionized testing with automation and seamless workflow integration. Evolved test plans, including V-Model, risk-based, and BDD approaches, further enhanced traceability, risk mitigation, and collaboration. Adapting test plans to new methodologies and technologies is crucial for continuous improvement in software quality.

References

- IEEE 829-2008 Standard for Software and System Test Documentation
- Agile Testing: A Practical Guide for Testers and Agile Teams by Lisa Crispin and Janet Gregory

- Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation by Jez Humble and David Farley
- GitHub Actions Documentation
- Jenkins Documentation
- Various industry whitepapers and case studies on test planning and CI/CD practices.