Software Testing Metrics, Tools and Standards

Chapter Syllabus

- Software Testing Metrics
- Software Testing Tools
  - Open Source and Proprietary/ Commercial
- Software Testing Standards
  - ISO/IEEE/IEC

Chapter Objectives

- After successful completion of this chapter, students will be able to
  - Explain software testing metrics
  - Understand when and why to apply testing metrics
  - Describe software testing tools
  - Understand the advantages and limitations of commonly used testing tools
  - Explain software testing standards

Software Testing Metric

- In software testing, metric is a quantitative measure of the degree to which a system, system component or process possesses a given attribute.
- In other words, metrics helps estimating the progress and quality of a software testing effort.
- Software testing metrics improve the effectiveness and efficiency of a software testing process.

…Metrics

"You can't control what you can't measure"

*(Tom DeMacro, 1982 p. 3)**

- Software test metrics is used for monitoring and controlling processes and products.
- It helps to drive the project towards our planned goals without deviation.

**An American software engineer, author and consultant on software engineering topics.*

Why test metrics?

- **Quality Assurance:** Metrics help in determining the quality of the software product and testing processes.
- **Decision-Making:** Provides a basis for making informed decisions about the software release.
- **Process Improvement:** Helps in identifying areas that need improvement in the testing process.
- **Risk Management:** Assists in understanding and mitigating risks associated with the software.

Software Test Metrics

- Software test metrics used in the process of test preparation, execution and defect analysis phases include:

Test Design Phase

1. Test Case Preparation Productivity
2. Test Design Coverage

Test Execution Phase

3. Test Execution Productivity
4. Test Execution Coverage
5. Test Cases Passed
6. Test Cases Failed
7. Test Cases Blocked
8. Test Effort Ratio

Defect Analysis Metrics

8. Error Discovery Rate
9. Defect Fix Rate
10. Defect Density
11. Defect Leakage
12. Defect Removal Efficiency

1. Test Case Preparation Productivity

- It is used to calculate the number of test cases prepared and the effort spent for the preparation of test cases.

- **Formula:**
  - $Test\ case\ preparation\ productivity = \frac{Number\ of\ Test\ cases}{Effort\ spent\ for\ Test\ Case\ Preparation}$

- **E.g.:**
  - No. of Test cases = 240
  - Effort spent for Test case preparation (in hours) = 80   hours
  - Test Case preparation productivity = 240/80 = **30 test cases/hour**
  - Asap, higher test case productivity is recommended

2. Test Design Coverage

- It helps to measure the percentage of test case coverage against the number of requirements

- **Formula:**
  - $Test\ Design\ Coverage = \left(\frac{Total\ No.of\ Requirements\ Mapped\ to\ Test\ Cases}{Total\ No.of\ Requirements}\right) \times 100$

- **E.g.:**
  - Total number of requirements: 78
  - Total number of requirements mapped to test cases: 69
  - Test design coverage = (69/78)*100 = **88.5%**

3. Test Execution Productivity

- It determines the number of Test Cases that can be executed per hour

- **Formula:**
  - $Test\ Execution\ Productivity = \frac{No.of\ Test\ Cases\ Executed}{Effort\ Spent\ for\ Execution\ of\ Test\ Cases}$

- **E.g.:**
  - No. of Test cases executed = 180
  - Effort spent for execution of test cases = 10 hours
  - Test Execution Productivity = 180/10 = **18 test cases/hour**

4. Test Execution Coverage

- It is used to measure the number of test cases executed against the number of test cases planned to be executed.
- **Formula:**
  - $Test\ Execution\ Coverage = \left(\frac{Total\ No.of\ Test\ Cases\ Executed}{Total\ No.of\ Test\ Cases\ Planned\ to\ be\ Executed}\right) \times 100$
- **E.g.:**
  - Total no. of test cases planned to be executed = 240
  - Total no. of test cases executed = 180
  - Test Execution Coverage = (180/240)*100 = **75%**

5. Test Cases Passed

- It measures the percentage of test cases passed during test execution. Also known as pass rate
- **Formula:**
  - $Test\ Cases\ Passed = \left(\frac{Total\ No.of\ Test\ Cases\ Passed}{Total\ No.of\ Test\ Cases\ Executed}\right) \times 100$
- **E.g.:**
  - Total number of test cases passed = 80
  - Total number of test cases executed = 90
  - Test Cases Passed = (80/90)*100 = 88.8 ≈ **89%**

6. Test Cases Failed

- It measures the percentage of test cases failed the test
- **Formula:**
  - $Test\ Cases\ Failed = \left(\frac{Total\ No.of\ Test\ Cases\ Failed}{Total\ No.of\ Test\ Cases\ Executed}\right) \times 100$
- **E.g.:**
  - Total no. of test cases failed = 10
  - Total no. of test cases executed = 90
  - Test Cases Failed= (10/90)*100 = 11.1 ≈ **11%**

7. Test Cases Blocked

- It measures the percentage of test cases blocked during the test
- **Formula:**
  - $Test\ Cases\ Blocked = \left(\frac{Total\ No.of\ Test\ Cases\ Blocked}{Total\ No.of\ Test\ Cases\ Executed}\right) \times 100$
- **E.g.:**
  - Total number of test cases blocked = 5
  - Total number of test cases executed = 90
  - Test Cases Blocked = (5/90)*100 = 5.5 ≈ **6%**

8. Test Effort Ratio

- Measures the ratio of time spent on testing versus the total project time
- **Formula:**
  - $Test\ \text{Effort Ratio} = \left(\frac{\text{Time Spent on Testing}}{Total\ \text{Project Time}}\right) \times 100$

  - - **E.g.**
  - A project has a total duration of = 1000 hours
  - Time spent for test = 400 hours
  - Test Effort Ratio = (400/1000)*100 = **40%**

Defect Analysis Metrics

- Software test metrics used in the process of defect analysis phase of the STLC includes:

1. Error Discovery Rate
2. Defect Fix Rate
3. Defect Density
4. Defect Leakage
5. Defect Removal Efficiency

1. Error Discovery Rate

- It determines the effectiveness of the test cases.
- **Formula:**
  - $Error\ Discovery\ Rate\ (EDR) = \left(\frac{Total\ No.of\ Defects\ Found}{Total\ No.of\ Test\ Cases\ Executed}\right) \times 100$
- **E.g.:**
  - Total number of defects found = 60
  - Total number of test cases executed = 240
  - Error Discovery Rate = (60/240)*100 = **25%**

2. Defect Fix Rate

- It helps to know the quality of a build in terms of defect fixing.

- **Formula:**
  - $Defect\ Fix\ Rate = \left(\frac{Total\ No.of\ Defects\ Reported\ as\ Fixed - Total\ No.of\ Defects\ Reopened}{Total\ No.of\ Defects\ Reported\ as\ Fixed + Total\ No.of\ New\ Bugs\ Due\ to\ Fix}\right) \times 100$

- **E.g.:**
- Total number of defects reported as fixed = 10
- Total number of defects reopened = 2
- Total number of new Bugs due to fix = 1
- Defect Fix Rate = ((10 – 2)/(10 + 1))*100 = (8/11)100 = 72.7 ≈ **73%**

Reopening is performed when the QA team decides that the defect still needs to be fixed.

3. Defect Density

- Defect density is the number of confirmed defects detected in the software or a component during a defined period of development or operation, divided by the size of the software.
- It is defined as the ratio of defects to requirements.
- Defect density determines the stability of the application.
- **Formula:**
  - $Defect\ Density = \dfrac{Total\ No.of\ Defects\ Identified}{Actual\ Size\ of\ Software\ Product}$

…Defect Density

- **E.g.:** Suppose you have a software product which has been integrated from 4 modules and you found the following bugs in each of the modules.
  - Module 1 = 20 bugs
  - Module 2 = 30 bugs
  - Module 3 = 50 bugs
  - Module 4 = 60 bugs
- And the total line of code for each module is
  - Module 1 = 1200 LoC
  - Module 2 = 3023 LoC
  - Module 3 = 5034 LoC
  - Module 4 = 6032 LoC
- Then, we calculate defect density as
  - Total bugs = 20+30+50+60 = 160
  - Total Size= 15289 LOC
  - Defect Density = 160/15289

    = 0.01046504 defects/LoC

    = **10.5 defects/KLoC**

4. Defect Leakage

- It is used to review the efficiency of the testing process before User Acceptance Testing (UAT).
- **Formula:**
- $Defect\ Leakage = \left(\frac{Total\ No.of\ Defects\ Found\ in\ UAT}{Total\ No.of\ Defects\ Found\ Before\ UAT}\right) \times 100$
- **E.g.:**
  - Number of defects found in UAT = 20
  - Number of defects found before UAT = 120
  - Defect Leakage = (20 /120) * 100 = 16.6 ≈ **17%**

5. Defect Removal Efficiency

- It allows us to compare the overall defect removal efficiency.
- It includes defects found pre and post-delivery.
- **Formula:**

$$Defect\ Removal\ Efficiency = \left(\frac{Total\ No. of\ Defects\ Found\ Pre\_Delivery}{Total\ No. of\ Defects\ Found\ Pre\_Delivery + Total\ No. Defects\ Found\ Post\_Delivery}\right) \times 100$$

- **E.g.:**
  - Total no. of defects found pre-delivery = 80
  - Total no. of defects found post-delivery = 10
  - Defect Removal Efficiency = ((80) / ((80) + (10)))*100

    = (80/90)*100 = 88.8 ≈ **89%**

Software Testing Tools

Manual Testing Vs Automated Testing

- Software testing is carried out using manual and automated software testing tools
- Tester should have the perspective of an end-user and to ensure all the features are working as mentioned in the requirement document.
- In this process, testers execute the test cases and generate the reports manually, without using any automation tools.

...Manual

- Advantages:
  - It can be done on all kinds of applications
  - It is preferable for short life cycle products
  - Newly designed test cases should be executed manually
  - Application must be tested manually before it is automated
  - It is preferred in the projects where the requirements change frequently and for the products where the GUI changes constantly
  - It is cheaper in terms of initial investment compared to automation testing
  - It requires less time and expense to begin productive manual testing
  - There is no necessity to the tester to have knowledge on automation Tools

…Manual

- Limitations:
  - Manual testing is time-consuming mainly while doing regression testing.
  - In the long run, expensive over automation testing

Automated Testing

- Automated testing is the process of testing a software using automated tools.
- Automated testing means running the software programs that carry out the execution of test cases automatically and produce the test results without any human intervention.
- In this process, executing the test scripts and generating the results are performed automatically by automated tools.

…Automated

- Advantages:
  - Automation testing is faster in execution
  - It is cheaper compared to manual testing in a long run
  - It is more reliable
  - It is more powerful and versatile
  - It is mostly used for regression testing
  - It does not require human intervention.
  - It helps to increase the test coverage

…Automated

- Limitations:
  - It is recommended only for stable products
  - Automation testing is expensive initially
  - Most of the automation tools are expensive
  - It has some limitations such as handling captcha, fonts, color
  - Huge maintenance in case of repeated changes in the requirements
  - Not all the tools support all kinds of testing.

…Automated

- Now days we can get lots of software testing tools in the market.
- Selection of tools is totally based on the project requirements
  - Proprietary/commercial tools or
  - Free tools/open source tools
- Off course, free testing tools may have some limitation in the features list of the product, so it’s totally based on what are you looking for & is that your requirement fulfill in free version or go for paid software testing tools.

…Automated

- The tools are divided into different categories as follows:

1. Test Management Tools
2. Functional Testing Tools
3. Non-Functional Testing Tools
    - Performance

Test Management Tools

Open-Source Tools

- TET (Test Environment Toolkit)
- TETware
- Test Manager
- RTH (Requirements & Testing Hub)

Proprietary/Commercial Tools

- HP Quality Center/ALM
- QA Complete
- T-Plan Professional
- Automated Test Designer (ATD)
- Testuff
- SMARTS
- QAS.TCS (Test Case Studio)
- PractiTest
- Test Manager Adaptors
- SpiraTest
- TestLog
- ApTest Manager
- DevTest

Functional Testing Tools

Open-Source Tools

- Selenium
- Soapui
- Watir
- HTTP::Recorder
- WatiN
- Canoo WebTest
- Webcorder
- Solex
- Imprimatur
- SAMIE
- Swete
- ITP
- WET
- WebInject
- Katalon Studio

Proprietary/Commercial Tools

- QuickTest Pro
- Rational Robot
- Sahi
- SoapTest
- Badboy
- Test Complete
- QA Wizard
- Netvantage Functional Tester
- PesterCat
- AppsWatch
- Squish
- actiWATE
- liSA
- vTest
- Internet Macros
- Ranorex

Performance Testing Tools

Open-Source Tools

- JMeter
- FunkLoad

Proprietary/Commercial Tools

- WebLOAD Professional
- HP LoadRunner
- LoadStorm
- NeoLoad
- Loadtracer
- Forecast
- ANTS – Advanced .NET Testing System
- vPerformer
- Webserver Stress Tool
- preVue-ASCII
- Load Impact

1. Selenium

- It is the most widely used automated testing tool among all web application testing tools.
- Selenium can be executed in multiple browsers and operating systems.
- It is compatible with several programming languages and automation testing frameworks.
- With selenium, you can come up with very powerful browser-centered automation test scripts which are scalable across different environments.
- You can also create scripts using Selenium that is of great help for prompt reproduction of bugs, regression testing, and exploratory testing.

2. Unified Functional Testing (UFT) QTP

- Unified Functional Testing (UFT) tool given by Hewlett-Packard Enterprise is one of the best automation testing software for functional testing. It was previously known as QuickTest Professional (QTP).
- It brings developers & testers coming together under one umbrella and provides high-quality automation testing solutions.
- It makes functional testing less complex and cost-friendly.
- Its top features include Cross browser & multi-platform compatibility, Optimized distributed testing, multiple testing solutions, image-based object recognition and canvas – visual test flows.

3. HP Quality Center

- It is basically an integrated IT quality management software.
- Automated testing is one of its key features which constantly allows you to test earlier and quicker.
- Asset sharing and reusability allows QC to deliver bug-free and reliable applications.

4. Testim.io

- It leverages machine learning for the authoring, execution, and maintenance of automated test cases.
- We use dynamic locators and learn with every execution.
- The outcome is super fast authoring and stable tests that learn, thus eliminating the need to continually maintain tests with every code change.

5.TestComplete

- It allows all level of users to quickly create powerful, reusable and time-saving GUI automation tests for the web, mobile, and desktop applications.
- It lets you combine the recorded scripts and tests into a single framework which reduces the training cost and testing time.
- The best part of the tool is TestComplete Visualizer**,** which is a screenshot based feature allows you to modify previously recorded tests, quickly update your assertions and checkpoint and provide a visual recording of your test – all in one area.

6. Telerik Test Studio

- Telerik test studio is a comprehensive test automation solution.
- It is well suited for GUI, performance, load and API testing.
- It allows you to test desktop, mobile and web applications.
- Its main features include Point-and-click test recorder, support for coding languages like C# and VB.NET, central object repository and continuous integration with source control.

7. Katalon Studio

- Katalon Studio is a powerful test automation solution for mobile, Web, and API testing. And it is completely FREE!
- It provides a comprehensive set of features for test automation, including recording actions, creating test cases, generating test scripts, executing tests, reporting results, and integrating with many other tools in the software development lifecycle.
- Katalon Studio runs on both Windows and MacOS, supporting automated testing of iOS and Android apps, web applications on all modern browsers, and API services.
- It can integrate with tools such as JIRA, qTest, Kobiton, Git, and Slack.

Testing Standards

Software Testing Standards

- Software testing standards are a set of rules that are expected to be met in a particular product or service offered by a company.
- Compliance to these set of rules is important for a company as it defines the extent to which a company is performing it duty legally and is not engaged in any sort of malpractice.

...Standards

There are two types of test standards:

- **External Standards**
  - Familiarity with and adoption of industry test standards from organizations.
- **Internal Standards**
  - Development and enforcement of the test standards that testers must meet. It is set by the developer/ tester company itself.

…Standards

- Institutes and organizations setting standards in software testing include:
  1. Institute of Electrical and Electronics Engineers (IEEE)
  2. International Organization for Standardization (ISO)
  3. International Electrotechnical Commission (IEC)
  4. National Institute of Standards and Technology (NIST)

Standard	Description
IEEE 829	A standard for the format of documents used in different stages of software testing.
IEEE 1061	A methodology for establishing quality requirements, identifying, implementing, analyzing, and validating the process, and product of software quality metrics.
IEEE 1059	Guide for Software Verification and Validation Plans.
IEEE 1008	A standard for unit testing.
IEEE 1012	A standard for Software Verification and Validation.
IEEE 1028	A standard for software inspections.
IEEE 1044	A standard for the classification of software anomalies.
IEEE 1044-1	A guide for the classification of software anomalies.
IEEE 830	A guide for developing system requirements specifications.
IEEE 730	A standard for software quality assurance plans.
IEEE 1061	A standard for software quality metrics and methodology.
IEEE 12207	A standard for software life cycle processes and life cycle data.
BS 7925-1	A vocabulary of terms used in software testing.
BS 7925-2	A standard for software component testing.

ISO/IEC/IEEE 29119

- ISO/IEC/IEEE 29119 Software Testing is an internationally agreed set of standards for software testing that can be used within any organization.
- By implementing these standards, you will be adopting the only internationally-recognized and agreed standards for software testing, which will provide your organization with a high-quality approach to testing that can be communicated throughout the world.
- There are currently five standards:
  1. ISO/IEC 29119-1: Concepts & Definitions (published in 2013)
  2. ISO/IEC 29119-2: Test Processes (published in 2013)
  3. ISO/IEC 29119-3: Test Documentation (published in 2013)
  4. ISO/IEC 29119-4: Test Techniques (published in 2014)
  5. ISO/IEC 29119-5: Keyword Driven Testing (published in 2015)

kind thanks