Table of Contents

[1.0 Business need 2](#_Toc219181104)

[2.0 Software Metrics 2](#_Toc219181105)

[3.0 Importance of Metrics 2](#_Toc219181107)

[4.0 Point to remember 3](#_Toc219181108)

[5.0 Metrics Lifecycle 3](#_Toc219181109)

[6.0 Type of Software Testing Metrics 4](#_Toc219181111)

[6.1 Manual Testing Metrics 5](#_Toc219181112)

[6.1.1 Test Case Productivity (TCP) 5](#_Toc219181113)

[6.1.2 Test Execution Summary 5](#_Toc219181114)

[6.1.3 Defect Acceptance (DA) 6](#_Toc219181115)

[6.1.4 Defect Rejection (DR) 6](#_Toc219181116)

[6.1.5 Bad Fix Defect (B) 7](#_Toc219181117)

[6.1.6 Test Execution Productivity (TEP) 7](#_Toc219181118)

[6.1.7 Test Efficiency (TE) 8](#_Toc219181119)

[6.1.8 Defect Severity Index (DSI) 9](#_Toc219181120)

[6.2 Performance Testing Metrics 10](#_Toc219181121)

[6.2.1 Performance Scripting Productivity (PSP) 10](#_Toc219181122)

[6.2.2 Performance Execution Summary 11](#_Toc219181123)

[6.2.3 Performance Execution Data - Client Side 11](#_Toc219181124)

[6.2.4 Performance Execution Data - Server Side 12](#_Toc219181125)

[6.2.5 Performance Test Efficiency (PTE) 12](#_Toc219181126)

[6.2.6 Performance Severity Index (PSI) 13](#_Toc219181127)

[6.3 Automation Testing Metrics 13](#_Toc219181128)

[6.3.1 Automation Scripting Productivity (ASP) 13](#_Toc219181129)

[6.3.2 Automation Test Execution Productivity (AEP) 14](#_Toc219181130)

[6.3.3 Automation Coverage 14](#_Toc219181131)

[6.3.4 Cost Comparison 14](#_Toc219181132)

[6.4 Common Metrics for all types of testing 15](#_Toc219181133)

[6.4.1 Effort Variance (EV) 15](#_Toc219181134)

[6.4.2 Schedule Variance (SV) 15](#_Toc219181135)

[6.4.3 Scope Change (SC) 15](#_Toc219181136)

[7.0 Conclusion 17](#_Toc219181137)

[8.0 Reference 18](#_Toc219181138)

[9.0 About Author 18](#_Toc219181139)

Software Testing Metrics

# Business need

Increase in competition and leaps in technology have forced companies to adopt innovative approaches to assess themselves with respect to processes, products and services. This assessment helps them to improve their business so that they succeed and make more profits and acquire higher percentage of market.

***Metric*** is the cornerstone in assessment and also foundation for any business improvement

# Software Metrics

Metric is a standard unit of measurement that quantifies results. Metric used for evaluating the software processes, products and services is termed as Software Metrics.

### Definition of Software Metrics given by Paul Goodman: -

Software Metrics is a Measurement Based Technique which is applied to processes, products and services to supply engineering and management information and working on the information supplied to improve processes, products and services, if required.

# Importance of Metrics

* Metrics is used to improve the quality and productivity of products and services thus achieving Customer Satisfaction.
* Easy for management to digest one number and drill down, if required.
* Different Metric(s) trend act as monitor when the process is going out-of-control.
* Metrics provides improvement for current process.

# Point to remember

* Metrics for which one can collect accurate and complete data must be used.
* Metrics must be easy to explain and evaluate.
* Benchmark for Metric(s) varies form organization to organization and also from person to person.

# Metrics Lifecycle

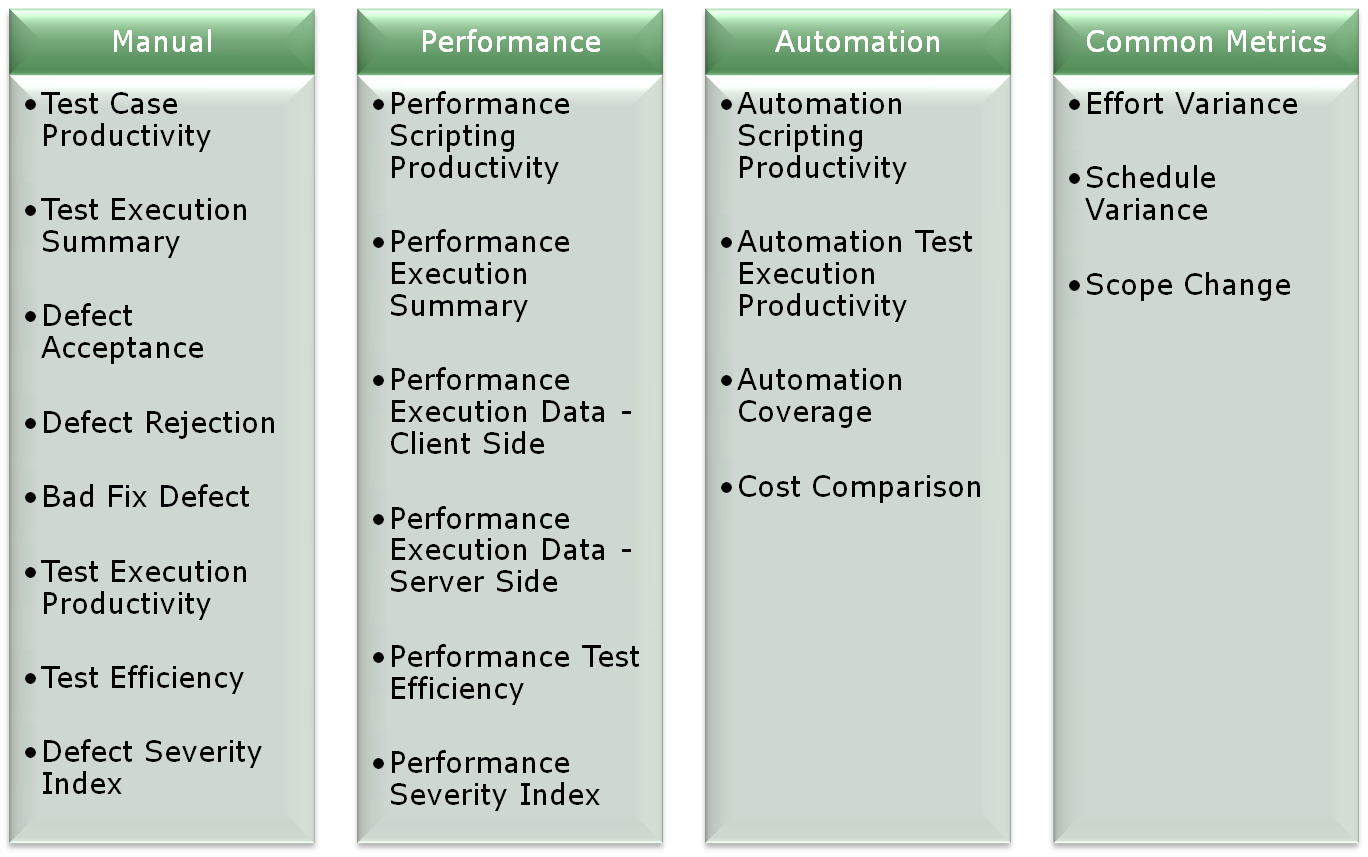
# The process involved in setting up the metrics:

# Type of Software Testing Metrics

Based on the types of testing performed, following are the types of software testing metrics: -

1. Manual Testing Metrics
2. Performance Testing Metrics
3. Automation Testing Metrics

Following figure shows different software testing metrics.



Let’s have a look at different type of metrics for each of them.

## Manual Testing Metrics

### Test Case Productivity (TCP)

This metric gives the test case writing productivity based on which one can have a conclusive remark.



**Example**

|  |  |
| --- | --- |
| **Test Case Name** | **Raw Steps** |
| XYZ\_1 | 30 |
| XYZ\_2 | 32 |
| XYZ\_3 | 40 |
| XYZ\_4 | 36 |
| XYZ\_5 | 45 |
| **Total Raw Steps** | **183** |

**Efforts** took for writing 183 steps is 8 hours.

TCP=183/8=22.8

Test case productivity = 23 steps/hour

One can compare the Test case productivity value with the previous release(s) and draw the most effective conclusion from it.

**TC Productivity Trend**

### Test Execution Summary

This metric gives classification of the test cases with respect to status along with reason, if available, for various test cases. It gives the statical view of the release. One can collect the data for the number of test case executed with following status: -

* Pass.
* Fail and reason for failure.
* Unable to Test with reason. Some of the reasons for this status are time crunch, postponed defect, setup issue, out of scope.

**Summary Trend**

One can also show the same trend for the classification of reasons for various unable to test and fail test cases.

### Defect Acceptance (DA)

This metric determine the number of valid defects that testing team has identified during execution.



The value of this metric can be compared with previous release for getting better picture

**Defect Acceptance Trend**

### Defect Rejection (DR)

This metric determine the number of defects rejected during execution.



This metric gives the percentage of the invalid defect the testing team has opened and one can control, if required, in future.

**Defect Rejection Trend**

### Bad Fix Defect (B)

Defect whose resolution give rise to new defect(s) are bad fix defect.

This metric determine the effectiveness of defect resolution process.



This metric gives the percentage of the bad defect resolution which needs to be controlled.

**Bad Fix Defect Trend**

### Test Execution Productivity (TEP)

This metric gives the test cases execution productivity which on further analysis can give conclusive result.



Where Te is calculated as,



Where,

Base Test Case = No. of TC executed atleast once.

T (1) = No. of TC Retested with 71% to 100% of Total TC steps

T (0.66) = No. of TC Retested with 41% to 70% of Total TC steps

T (0.33) = No. of TC Retested with 1% to 40% of Total TC steps

**Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **TC Name** | **Base Run Effort (hr)** | **Re-Run1 Status** | **Re-Run1 Effort (hr)** | **Re-Run2 Status** | **Re-Run2 Effort (hr)** | **Re-Run3 Status** | **Re-Run3 Effort (hr)** |
| XYZ\_1 | 2 | T(0.66) | 1 | T(0.66) | 0.45 | T(1) | 2 |
| XYZ\_2 | 1.3 | T(0.33) | 0.3 | T(1) | 2 |  |  |
| XYZ\_3 | 2.3 | T(1) | 1.2 |  |  |  |  |
| XYZ\_4 | 2 | T(1) | 2 |  |  |  |  |
| XYZ\_5 | 2.15 |  |  |  |  |  |  |

|  |  |
| --- | --- |
| Base Test Case | 5 |
| T(1) | 4 |
| T(0.66) | 2 |
| T(0.33) | 1 |
| Total Efforts(hr) | 19.7 |

In above example,

Te = 5 + ((1\*4) + (2\*0.66) + (1\*0.33))) = 5 + 5.65 = 10.65

Test Execution Productivity = (10.65/19.7) \* 8 = 4.3 Execution/day

One can compare the productivity with previous release and can have an effective conclusion.

**Test Execution Productivity Trend**

### Test Efficiency (TE)

This metric determine the efficiency of the testing team in identifying the defects.

It also indicated the defects missed out during testing phase which migrated to the next phase.



Where,

DT = Number of valid defects identified during testing.

DU = Number of valid defects identified by user after release of application. In other words, post-testing defect

**Test Efficiency Trend**

### Defect Severity Index (DSI)

This metric determine the quality of the product under test and at the time of release, based on which one can take decision for releasing of the product i.e. it indicates the product quality.



One can divide the Defect Severity Index in two parts: -

1. **DSI for All Status defect(s):** - This value gives the product quality under test.
2. **DSI for Open Status defect(s):** - This value gives the product quality at the time of release. For calculation of DSI for this, only open status defect(s) must be considered.



**Defect Severity Index Trend**

From the graph it is clear that

* Quality of product under test i.e. DSI – All Status = 2.8 (High Severity)
* Quality of product at the time of release i.e. DSI – Open Status = 3.0 (High Severity)

## Performance Testing Metrics

### Performance Scripting Productivity (PSP)

This metric gives the scripting productivity for performance test script and have trend over a period of time.



Where Operations performed is: -

1. No. of Click(s) i.e. click(s) on which data is refreshed.
2. No. of Input parameter
3. No. of Correlation parameter

Above evaluation process does include logic embedded into the script which is rarely used.

**Example**

|  |  |
| --- | --- |
| **Operation Performed** | **Total** |
| No. of clicks | 10 |
| No. of Input Parameter | 5 |
| No. of Correlation Parameter | 5 |
| **Total Operation Performed** | **20** |

Efforts took for scripting = 10 hours.

Performance scripting productivity =20/10=2 operations/hour

**Performance Scripting Productivity Trend**

### Performance Execution Summary

This metric gives classification with respect to number of test conducted along with status (Pass/Fail), for various types of performance testing.

Some of the types of performance testing: -

1. Peak Volume Test.
2. Endurance/Soak Test.
3. Breakpoint/Stress Test.
4. Failover Test

**Summary Trend**

### Performance Execution Data - Client Side

This metric gives the detail information of Client side data for execution.

Following are some of the data points of this metric -

1. Running Users
2. Response Time
3. Hits per Second
4. Throughput
5. Total Transaction per second
6. Time to first byte
7. Error per second

### Performance Execution Data - Server Side

This metric gives the detail information of Server side date for execution.

Following are some of the data points of this metric -

1. CPU Utilization
2. Memory Utilization
3. HEAP Memory Utilization
4. Database connections per second

### Performance Test Efficiency (PTE)

This metric determine the quality of the Performance testing team in meeting the requirements which can be used as an input for further improvisation, if required.



To evaluate this one need to collect data point during the performance testing and after the signoff of the performance testing.

Some of the requirements of Performance testing are: -

1. Average response time.
2. Transaction per Second.
3. Application must be able to handle predefined max user load.
4. Server Stability.

**Example**

Consider during the performance testing above mentioned requirements were met.

In production, average response time is greater than expected, then

Requirement met during PT = 4

Requirement not met after Signoff of PT = 1

PTE = (4 / (4+1)) \* 100 = 80%

Performance Testing Efficiency is 80%

**Performance Test Efficiency Trend**

### Performance Severity Index (PSI)

This metric determine the product quality based performance criteria on which one can take decision for releasing of the product to next phase i.e. it indicates quality of product under test with respect to performance.



If requirement is not met, one can assign the severity for the requirement so that decision can be taken for the product release with respect to performance.

**Example**

Consider, Average response time is important requirement which has not met, then tester can open defect with Severity as Critical.

Then Performance Severity Index = (4 \* 1) / 1 = 4 (Critical)

**Performance Severity Trend**

## Automation Testing Metrics

### Automation Scripting Productivity (ASP)

This metric gives the scripting productivity for automation test script based on which one can analyze and draw most effective conclusion from the same.



Where Operations performed is: -

1. No. of Click(s) i.e. click(s) on which data is refreshed.
2. No. of Input parameter
3. No. of Checkpoint added

Above process does include logic embedded into the script which is rarely used.

**Example**

|  |  |
| --- | --- |
| **Operation Performed** | **Total** |
| No. of clicks | 10 |
| No. of Input Parameter | 5 |
| No. of Checkpoint added | 10 |
| **Total Operation Performed** | **25** |

**Efforts** took for scripting = 10 hours.

ASP=25/10=2.5

Automation scripting productivity = 2.5 operations/hour

**Automation Scripting Productivity Trend**

### Automation Test Execution Productivity (AEP)

This metric gives the automated test case execution productivity.

Where Te is calculated as,



Evaluation process is similar to Manual Test Execution Productivity.

### Automation Coverage

This metric gives the percentage of manual test cases automated.



**Example**

If there are 100 Manual test cases and one has automated 60 test cases then Automation Coverage = 60%

### Cost Comparison

This metrics gives the cost comparison between manual testing and automation testing. This metrics is used to have conclusive ROI (return on investment).

Manual Cost is evaluated as: -

**Cost (M)** = Execution Efforts (hours) \* Billing Rate

Automation cost is evaluated as: -

**Cost (A)** = Tool Purchased Cost (One time investment) + Maintenance Cost

+ Script Development Cost

+ (Execution Efforts (hrs) \* Billing Rate)

If Script is re-used the script development cost will be the script update cost.

Using this metric one can have an effective conclusion with respect to the currency which plays a vital role in IT industry.

## Common Metrics for all types of testing

### Effort Variance (EV)

This metric gives the variance in the estimated effort.



**Effort Variance Trend**

### Schedule Variance (SV)

This metric gives the variance in the estimated schedule i.e. number of days.



**Schedule Variance Trend**

### Scope Change (SC)

This metric indicates how stable the scope of testing is.



Where,

Total Scope = Previous Scope + New Scope, if Scope increases

Total Scope = Previous Scope - New Scope, if Scope decreases

**Scope Change Trend for one release**

# Conclusion

***Metric*** is the cornerstone in assessment and foundation for any business improvement. It is a Measurement Based Technique which is applied to processes, products and services to supply engineering and management information and working on the information supplied to improve processes, products and services, if required. It indicates level of Customer satisfaction, easy for management to digest number and drill down, whenever required and act as monitor when the process is going out-of-control.

Following table summarize the Software testing metrics discussed in this paper:

|  |  |
| --- | --- |
| **Test Metric** | **Description** |
|  | |
| **Manual Testing Metrics** | |
| Test Case Productivity | Provides the information for the number of step(s) written per hour. |
| Test Execution Summary | Provides statical view of execution for the release along with status and reason. |
| Defect Acceptance | Indicates the stability and reliability of the application. |
| Defect Rejection | Provides the percentage of invalid defects. |
| Bad Fix Defect | Indicates the effectiveness of the defect-resolution process |
| Test Execution Productivity | Provides detail of the test case executed per day. |
| Test Efficiency | Indicates the testing capability of the tester in identifying the defect. |
| Defect Severity Index | Provides indications about the quality of the product under test and at the time of release. |
| **Performance Testing Metrics** | |
| Performance Scripting Productivity | Provides scripting productivity for performance test flow. |
| Performance Execution Summary | Provides classification with respect to number of test conducted along with status (Pass/Fail), for various types of performance testing. |
| Performance Execution Data - Client Side | Gives the detail information of Client side data for execution |
| Performance Execution Data - Server Side | Gives the detail information of Server side data for execution |
| Performance Test Efficiency | Indicates the quality of the Performance team in meeting the performance requirement(s). |
| Performance Severity Index | Indicates quality of product under test with respect to performance criteria. |
| **Automation Testing Metrics** | |
| Automation Scripting Productivity | Indicates the scripting productivity for automation test script. |
| Auto. Execution Productivity | Provides execution productivity per day. |
| Automation Coverage | Gives percentage of manual TC automated |
| Cost Comparison | Provides information on ROI |
| **Common metrics for all types of testing** | |
| Effort Variance | Indicates effort stability |
| Schedule Variance | Indicates schedule stability |
| Scope Change | Indicates requirement stability |

Thus, Metrics help organization to obtain the information it needs to continue to improve its processes, products and services and achieve the desired Goal as:

"**You cannot control what you cannot measure**" (Tom DeMarco)

# Reference

<http://www.prosci.com/metrics.htm>

<http://www.westfallteam.com/Papers/>

<http://www.stpmag.com/downloads/stp-0507_testmetrics.htm>

# About Author

A Cognizant India Test Engineer with over 3.5 years of proven Quality and Test management experience in the Financial Services sector.  Consistently developed and implemented new ideas and techniques which have lead to dramatic Quality improvement within the projects. Published two whitepapers: Customer Satisfaction through Quality Index and Sanity Testing which are available at: -

<http://www/projectperfect.com.au/info_quality_index.php> and

<http://www.stickyminds.com>

To reach email at: - [Lokesh.gulechha@cognizant.com](mailto:Lokesh.gulechha@cognizant.com)