## 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.



### Scope Change (SC)

This metric indicates how stable the scope of testing is.



## Manual Testing Metrics

### Test Case Productivity (TCP)

* This metric gives how productivity are you writing the test cases for game.
* The value of this metric can be compared with previous release for getting better picture



**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

**TC Productivity Trend**

### Test Execution Summary

Get the summary of how much test case pass and how much test case failed

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)

* Provides the number of defects accepted during execution of the game.
* The value of this metric can be compared with previous release for getting better picture



**Defect Acceptance Trend**

### Defect Rejection (DR)

* + Provides the number of defects rejected during execution of the game.
  + The value of this metric can be compared with previous release for getting better picture
  + Depend on what people you have to set the %



### 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.



It 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. how much test case have been tested.



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 Efficiency (TE)

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

It also include the defects missed out during testing 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

### Defect Severity Index (DSI)

This metric determine the quality of the game under test and at the time of release.



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

1. **DSI for All Status defect(s):** - This value gives the quality of the game during unreleased.
2. **DSI for Open Status defect(s):** - This value gives the quality of the game after in a time of release.



**Defect Severity Index Trend**

If using a graph:

* 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 in the game.



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

When making online game. 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

When making online game. 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 show the quality of the Performance testing team in meeting the requirements.



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.

Requirement met during PT = 4

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

Requirement not met after Signoff of PT = 1

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

Performance Testing Efficiency is 80%

### 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.



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**

Common Metrics for all types of