SOFTWARE TESTING METRICS

By tracking testing metrics, you can quickly spot problems, improve your testing and ensure your software is top-notch. The key metrics are:

- Project progress metrics
- Test progress metrics
- Product quality metrics
- Defect metrics
- Cost metrics
- Risk metrics
- Coverage metrics

01 PROJECT PROGRESS METRICS

They give you a clear snapshot of where the project stands, ensuring on-time delivery and highlighting areas to optimize.

Task completion

The ratio of finished tasks to total planned tasks.

$$\frac{\text{No. of tasks completed}}{\text{Total tasks}} \times 100\%$$

Resource usage

Resources used compared to those allocated.

$$\frac{\text{Logged hours by testers}}{\text{Allocated hours for testing}} \times 100\%$$

Defects open-close rate

Compares the rate of opened vs. closed defects.

$$\frac{\text{No. of defects opened}}{\text{No. of defects closed}}$$

Rework effort rate

Measures effort to fix defects or retest.

$$\frac{\text{Total hours spent on rework}}{\text{Total hours spent on initial testing}}$$

Test effort

Total hours spent on test planning, design, execution, and defect tracking.

Planning hours + design hours + execution hours + defect tracking hours

Test environment setup progress

Shows the progress in setting up the test environment.

Often qualitative, based on a checklist of environment components.

02 TEST PROGRESS METRICS

They monitor the execution time and effectiveness of tests to ensure they align with your project goals.

Defect discovery rate

Defects discovered in a given time frame (e.g., daily or weekly).

Test execution time

Total time taken to execute a particular test or set of tests.

Passed/failed

Total number of test cases that passed or failed.

Test pass rate

Percentage of test cases passed in a cycle.

$$\frac{\text{No. of passed test cases}}{\text{Total test cases run}} \times 100\%$$

Number of test cases run/not run

Total test cases executed against the total planned.

Test case implementation progress

Shows the progression of test cases from design to completion

$$\frac{\text{No. of implemented test cases}}{\text{Total test cases designed}} \times 100\%$$

Test environment preparation progress

Reflects test environment readiness; typically binary (ready/not ready).

$$\frac{\text{No. of environment components set up}}{\text{Total environments components required}} \times 100\%$$

03 PRODUCT QUALITY METRICS

They gauge how well your software performs, ensuring it meets user expectations and keeps a competitive edge.

Mean time to failure (MTTF)

Average time between system failures.

$$\frac{\text{Total operational tme}}{\text{No. of failures}}$$

Availability

Percentage of app uptime or availability.

$$\frac{\text{Operational time}}{\text{Total time}} \times 100\%$$

Response time

Average system response time to a user's action or request.

$$\frac{\sum \text{response times for all request}}{\text{Total No. of requests}}$$

Customer satisfaction

How closely the product aligns with customer expectations.

Often derived from an average score on feedback forms or surveys.

Customer found defects (CFD)

Defects reported by customers after release.

Total post - release defects reported by customers.

System reliability

How likely it is for a system to run without failing.

$$\frac{\text{MTTF} + \text{MTTR}}{\text{MTF}}$$

MTTF = Mean time to failure
MTTR = Mean time to repair

04 DEFECT METRICS

They're used to quantify and assess the quality of a software product by measuring the defects or issues found during the development and testing phases.

Defect density

Number of defects per size of the software module.

$$\frac{\text{No. of defects}}{\text{Size of the software (KLOC or function points)}}$$

Defect detection percentage

Ratio of defects found in testing vs. after release.

$$\frac{\text{Defects found in testing}}{\text{Total defects}}$$

Defect reopen rate

The rate at which closed/fixed defects are reopened due to incorrect fixes or similar issues.

$$\frac{\text{No. of reopened defects}}{\text{Total number of defects fixed}} \times 100\%$$

Open Defect Age

Average time defects stay open.

$$\frac{\sum \text{days each defect remains}}{\text{No. of open defects}}$$

Number and Priorities of Defects Found/Fixed

Total number of defects categorized by their priority (e.g., critical, high, medium, low) and their status (found or fixed)

Closed Defect Age

Average time to fix defects.

$$\frac{\text{Days to fix defects}}{\text{Closed defects}}$$

05 COST METRICS

These metrics show how software testing affects the financial side of things.

Cost of downtime

The financial impact during periods when the software or system is not operational due to defects or other reasons.

Organizational cost of quality (CoQ)

Overall cost for maintaining and assessing product quality, encompassing prevention, appraisal, and failure costs.

Cost of testing

Sum of all testing - related expenses, including resources, tools, and infrastructure.

Failure cost

Costs associated with defects found after the product release. These can include patching, hotfixes, support costs, and potential compensation to customers.

Cost per defect

Average cost for each detected defect.

$$\frac{\text{Total No. of defects detected}}{\text{Total cost of testing}}$$

06 RISK METRICS

Understanding and monitoring risks is essential for you to address the most critical issues in the right way.

Residual risk level

The level of risk remaining after mitigation efforts.

Inherent risk – Risk mitigation effect

Risk exposure

Measures the potential impact of a risk when it materializes.

Probability of occurrence × Potential impact

07 COVERAGE METRICS

Coverage metrics reveal untested parts, ensuring thorough checks and better software.

Requirements coverage

Percentage of requirements with test cases.

$$\frac{\text{No. of requirements with test cases}}{\text{Total requirements}} \times 100\%$$

Code coverage

Percentage of code tested by the application.

$$\frac{\text{Lines of code executed by test}}{\text{Total lines of code}} \times 100\%$$

Branch coverage

Measures the percentage of traversed code branches (e.g., if, else).

$$\frac{\text{No. of executed branches}}{\text{Total number of branches}} \times 100\%$$

These metrics help you:

- ✓ Optimize time, manpower, and budget usage
- ✓ Track testing progress and environment readiness
- ✓ Measure app speed, uptime, and reliability
- ✓ Spot and fix flaws quickly
- ✓ Understand and address risks
- ✓ Balance testing costs with product quality benefits

Need quality assurance for your product? — Learn more →