DOI:10.5937/IIZS26271R

METRICS FOR EVALUATING SOFTWARE TESTING QUALITY

Research paper

Branislava Radu[1], Ljubica Kazi[1]
[1]University of Novi Sad, Technical Faculty "Mihajlo Pupin", Zrenjanin, Serbia

Abstract: Software testing quality is an important aspect of developing reliable software systems and a large number of different metrics are used to evaluate it. The aim of this paper is to present a preliminary overview of metrics used to assess various aspects of testing, including the testing process, efficiency, test case quality and test code quality. Different groups of indicators are considered, with particular emphasis on coverage, defect detection and mutation testing metrics. The literature review indicates the limitations of individual metrics, particularly code coverage and highlights the need to combine multiple indicators for a more comprehensive assessment of testing quality.
Key words: testing quality, testing metrics, test case, software testing

INTRODUCTION

Software testing represents a significant part of the software development process. As software systems become increasingly large and complex, thorough testing and the application of appropriate metrics can contribute to greater confidence in software reliability, while testing metrics provide quantitative indicators for evaluating testing activities and processes(Singh et al., 2008).The development of technology has led to the increasingly widespread use of software solutions in business, prompting companies to improve development and testing processes in order to increase their efficiency and respond to increasingly complex business requirements (Barraood et al., 2023). The assessment of testing effectiveness is most commonly based on indicators such as code coverage, defect detection rate and test execution time. However, these indicators do not provide a comprehensive view of testing quality, as each addresses only a specific aspect of the process. (Do & Morales, 2026)In addition to selecting appropriate metrics, the quality of the tests themselves is an important factor in testing effectiveness, as it can affect the reliability of testing results and the quality of the final software product (Bowes et al., 2017).Testing aims to detect defects and ensure an adequate level of quality before the software product is released, but in practice it faces various challenges, including changing requirements. Under such conditions, test case quality represents an important aspect of testing effectiveness, while existing metrics for its measurement are not always sufficiently clear and do not enable a straightforward assessment of different test case quality characteristics (Barraood et al., 2023).

The aim of this paper is to present the metrics used to evaluate software testing quality and to examine them in relation to the different aspects of testing that they assess. Particular attention is given to metrics related to the testing process, efficiency and testing outcomes, as well as metrics for test case and test code quality. In addition to identifying and classifying these metrics, their limitations are also discussed, particularly regarding the extent to which an individual indicator can provide a comprehensive view of testing quality.

The paper is organized as follows. The second section presents the theoretical foundations of software testing, including basic concepts related to test cases and approaches to their identification. The third section provides an overview of related research on metrics for the testing process, efficiency and testing outcomes, as well as metrics for test case and test code quality. The fourth section discusses selected metrics identified in the reviewed studies and summarizes them according to the aspects of testing quality they assess. Finally, the paper presents conclusions and directions for future research.

[1]Corresponding author,branislava.radu@tfzr.rs, ORCID: 0009-0004-3337-8445
[2] ljubica.kazi@tfzr.rs, ORCID: 0000-0001-8089-3029

BACKGROUND

A test case represents a fundamental element of the testing process and defines how a particular functionality or property of the software is verified. Its structure includes an identifier, a description of the testing objective, preconditions, input data, expected outputs and postconditions and may also contain additional information about test execution. Particular attention should be given to expected outputs, whose precise definition can, in some cases, be complex and may require the involvement of domain experts. Test case execution involves establishing the preconditions, entering the specified input data, observing the actual results obtained and comparing them with the expected results. Based on this comparison, it is determined whether the test case has been successfully executed, after which the postconditions are checked, if necessary. Given their role in a systematic and repeatable testing process, test cases represent important work artifacts that are created, reviewed, executed, maintained and stored throughout the software life cycle(Jorgensen, 2014).
When identifying test cases, two fundamental approaches can be applied: specification-based and code-based approaches. The former is based on the expected behavior of the software and does not require knowledge of how the system is implemented. In contrast, code-based testing relies on the software implementation when selecting test cases, while coverage metrics are used to assess the extent of testing. However, neither approach alone enables complete verification of the software, because implementation-based testing cannot reveal behavior that is specified but not implemented, while specification-based testing alone cannot detect additional behavior that exists in the implementation but is not covered by the specification(Jorgensen, 2014).

METRICS FOR EVALUATING THE TESTING PROCESS AND EFFICIENCY

This section presents metrics used to evaluate different aspects of the software testing process.
Singh et al. (2008) examine various metrics for assessing the quality and effectiveness of the testing process, emphasizing that no single metric is sufficient for its comprehensive evaluation. Based on data from the NASA Metrics Data Program (MDP) repository, metrics such as Quality of Code (QC), Quality of Product (QP) and Test Effectiveness (TE) are analyzed. QC relates the number of weighted defects to the size of the product, whereas QP considers the weighted defects delivered to users in relation to the product size. The effectiveness of defect removal across different phases is determined by the ratio of defects of a particular type detected in the corresponding phase to the total number of such defects, while TE is based on the ratio of closed defects to the total number of reported defects. In this way, these metrics enable the monitoring of product quality and the effectiveness of the testing process in detecting and resolving defects (Singh et al., 2008).
In addition to the aforementioned indicators, (Singh et al., 2008)also considers metrics related to testing time, cost and efficiency, as well as measures of software size and reliability, test session efficiency, testing focus and software maturity. The study also covers metrics of subprogram complexity, test coverage, computer resource utilization and structural characteristics of modules, such as the number of procedures, conditions and loops. A specific group of indicators concerns the analysis of defects according to their cause, module, priority, type, detection method, detection phase and severity. The study also considers indicators of development efficiency, test automation and testing costs, including average testing efficiency, the contribution of requirements reviews, design reviews and program reviews, as well as the time required to verify fixes.The authors of paper also propose the Error Focus and Test Session Efficiency (TSE) metrics. Error Focus measures the proportion of reported errors that are successfully resolved and subsequently closed, while TSE relates the number of weighted defects identified during a specific testing phase to the number of added or modified lines of source code (Singh et al., 2008).

A broader overview of software metrics is presented in(Lee & Chang, 2013), where metrics are classified according to their significance, measurement method and area of application. In the context of testing, the metrics include test case development productivity, defect acceptance and rejection, defects caused by inadequate corrections, defect density in test cases, testing efficiency and effectiveness, as well as the ratio of testing time and costs to development time and costs and product size. In addition, the efficiency of individual testing phases, scripting and automation productivity and system performance expressed through response time, throughput and processor and memory utilization are considered (Lee & Chang, 2013).
The evaluation of such indicators is further addressed in (Zhang et al., 2022), which focuses not only on defining test suite effectiveness metrics but also on their reliability. The authors propose the ASSENT (evAluating teSt Suite EffectiveNess meTrics) framework for the systematic evaluation of metrics and investigate the extent to which their values correspond to the actual effectiveness of test suites. Particular attention is given to inconsistent findings in previous studies regarding the relationship between mutation score and the ability to detect real defects. The authors identify the choice of ground truth, reference test suites and agreement assessment method as possible sources of these differences (Zhang et al., 2022).
In an experiment conducted on projects from the Defects4J dataset, seven metrics were compared: statement coverage, branch coverage, mutation score, Certain Operator Selection (COS), Random Mutant Selection (RMS), Subsuming Mutation Score (SMS) and CMS (Clustering Mutation Score). A rank correlation measure was used to assess the agreement between metric values and the ground truth. When real defects were used as the ground truth, MS and SMS showed the highest accuracy in this experiment, while Statement coverage (SC), Branch Coverage (BC), COS and RMS were insensitive to improvements in test suite effectiveness in certain cases. When mutants were used as an alternative ground truth, all analyzed metrics were overestimated by more than 20% on average, with mutation score-based metrics being more overestimated than code coverage metrics. The authors therefore highlight the importance of selecting the ground truth and the method of generating test suites when evaluating metrics and recommend systematically validating new indicators within the ASSENT framework (Zhang et al., 2022).

METRICS FOR EVALUATING TEST CASE QUALITY

This section presents metrics and characteristics used to evaluate test case and test code quality.

In addition to metrics that assess the testing process as a whole, a significant body of research focuses on the quality of individual test cases and test code. The systematic review presented in research found that there is a lack of studies specifically addressing test case quality, while most of the existing literature focuses on individual characteristics, such as test case reuse, generation and selection, readability, understandability, test smells, design defect analysis and the relationship between coverage and effectiveness. Based on the analyzed studies, 16 quality characteristics were identified, including readability, understandability, simplicity or complexity, correctness, independence, reusability and maintainability (Barraood et al., 2023).
The same study identified a total of 50 metrics used to assess different aspects of test case quality. Statement coverage, code coverage and branch coverage are among the most frequently used, while mutation score appears in a smaller number of studies. Other metrics considered include modified condition coverage, Weighted Methods per Class (WMC), Depth of Inheritance Tree (DIT), Number of Children (NOC), Lack of Cohesion of a TestMethod (LCT) and the number of assertions, as well as indicators of the structural characteristics of tests, readability, execution time, cost, number of test methods and number of detected faults (Barraood et al., 2023).

Individual metrics can be associated with different quality characteristics. Code, branch and statement coverage metrics are used to assess coverage and effectiveness, while mutation score and the number of detected faults can be used to evaluate the ability of tests to detect defects. WMC, DITand NOCmetrics are related to the complexity and maintainability of test code, whereas textual metrics enable the assessment of readability and understandability. Execution time, cost and the number of test cases are associated with testing efficiency (Barraood et al., 2023).For some metrics, such as mutation score and the number of faults detected by test cases, it is not clearly specified which quality characteristics they measure. Therefore, further analysis is needed to determine their relationship with the corresponding quality characteristics.

Testing efficiency is addressed through metrics such as event-pair coverage, cost in hours, number of test methods and test case execution time. The ability to detect faults is assessed using the number of faults detected by a test case, composite code coverage, all-pairs results and mutation killing ratios. Other metrics include the number of test items, constants, variables, preceding test cases and applications that can use a test case, as well as the number of successful user acceptance tests, the total number of detected defects and the number of defects per line of code(Barraood et al., 2023).

A similar approach to test case quality is presented in (Barraood et al., 2018), in which a systematic literature review identified 30 metrics and indicators. These include test team experience, the quality of documented test cases, error and defect density, test case understandability, variability and independence, as well as test case size, similarity, cohesion and coupling. A specific group consists of coverage criteria, including method, statement, branch and condition coverage, as well as adequacy criteria based on coverage and defects. Other indicators considered include mutation analysis, historical fault detection, modified method coverage, defect detection capability and rate, defect detection time, test suite size and McCabe's cyclomatic complexity (Barraood et al., 2018).

The review in (Barraood et al., 2018) also highlights the limitations of coverage as a standalone indicator. Although different types of coverage are frequently used to assess the quality and completeness of a test suite, some of the analyzed studies do not recommend using coverage as the sole measure of test suite effectiveness. Therefore, coverage is combined with other indicators, while some studies consider mutation analysis as a complement to coverage because it enables the assessment of the ability of tests to detect changes in the program. One of the analyzed studies also considers test case quality in terms of design errors and the level of knowledge of the people who create them, indicating that insufficient understanding of relevant knowledge may limit test case quality. The identified metrics are used in various contexts, including software reliability and maintenance, productivity, reuse, test case selection, generation and prioritization, as well as test suite diagnosis (Barraood et al., 2018).

In addition to individual metrics, the quality of test code is also discussed in the literature through principles related to its structure, understandability, reliability and maintainability. Important characteristics include test simplicity, readability and understandability, clear responsibility of an individual test, focus on system behavior, test independence and maintainability. These principles can be associated with measurable characteristics of test code, such as test size, the number of `assert` statements, complexity, the amount of duplicated code, execution time and the way test dependencies are used. Particular emphasis is placed on the need for tests to cover both expected and unexpected scenarios, produce consistent results when repeatedly executed and be sufficiently simple to maintain (Bowes et al., 2017).

Test code quality is also associated with the presence of test smells, which may indicate problems in the structure, readability and maintainability of tests. Test suite quality can also be assessed by combining the number of assertions with test code structure metrics. These approaches further highlight the multidimensional nature of test quality, as different metrics and characteristics address different aspects of test code(Bowes et al., 2017).

The relationship between different characteristics of test code and its quality is also examined in (Grano et al., 2020), where unit test quality is considered in terms of behavioral,

structural and execution characteristics. Behavioral characteristics include self-validation, scope, efficiency and fault diagnosability, with code coverage and mutation score identified as important indicators. Structural characteristics include test size and design, readability, maintainability, independence and reusability, while execution characteristics include execution time, reliability and the conditions of the execution infrastructure (Grano et al., 2020).

In the second part of the study (Grano et al., 2020), the extent to which existing metrics correspond to developers' perceptions of unit test quality was examined. A total of 199 valid ratings provided by 70 developers for 10 tests were analyzed, considering mutation score, line coverage, complexity, coupling, test code size, test smells, assertion density and readability. Due to the high correlation between line coverage and mutation score, only mutation score was included in the final model. In the analyzed dataset, mutation score showed the strongest positive association with higher test quality ratings. Higher coupling between objects, the presence of the Assertion Roulette smell and a higher number of static calls (NOSI) were associated with lower ratings, whereas assertion density showed a positive association with quality ratings. Size, complexity and readability did not show a statistically significant association with developers' ratings, although readability was identified as an important characteristic of a high-quality test in the qualitative part of the study (Grano et al., 2020).

The findings of these studies particularly highlight the need to combine different indicators, which is further examined in(Jain et al., 2023).The authors analyze the limitations of using code coverage and mutation score individually to assess test suite adequacy. Coverage is a commonly used approximation of test adequacy, but it measures the execution of program code rather than necessarily the extent to which its behavior has been tested(Jain et al., 2023).

Based on an analysis of Java projects, the authors of (Jain et al., 2023) identify a positive but imperfect relationship between coverage and mutation score, with considerable variability at higher coverage levels. As an additional indicator, they propose the oracle gap, defined as the difference between source code coverage and mutation score. This approach suggests that the difference between two metrics can provide useful information about aspects of a test suite that are not visible when only one of the metrics is considered(Jain et al., 2023).

In addition to identifying potential metrics, it is important to examine whether characteristics that are commonly considered indicators of high-quality test cases are actually associated with their effectiveness. This question is addressed by the authors of (Lucrédio et al., 2023), who analyze characteristics such as size, complexity, atomicity, independence, readability, exception handling and executed code. In addition to existing metrics, such as lines of code, McCabe's cyclomatic complexity and readability, the study also uses metrics defined as part of the research, including the number of commits related to bug fixes, the number of assertions, coupling metrics and indicators of exception handling(Lucrédio et al., 2023).

The empirical analysis in (Lucrédio et al., 2023) did not confirm most of the hypotheses concerning the characteristics of high-quality test cases. The number of assertions, independence, exception handling and test case size did not show a significant relationship with test case effectiveness in detecting faults. A relationship between test case size and cyclomatic complexity was confirmed, while test case footprint and readability showed relationships contrary to expectations. Based on these findings, the authors indicate that widely accepted assumptions about test case quality are not necessarily supported by empirical data. The study identifies the selection of analyzed projects and the possibility of errors related to the implementation of metrics and the tools used as limitations, which restricts the generalizability of the findings (Lucrédio et al., 2023).

DISCUSSION

This section discusses the metrics identified in the reviewed studies for evaluating the effectiveness and efficiency of the testing process, as well as the quality of test cases and test code. Selected metrics from the analyzed studies are summarized in Tables 1 and 2.

Table 1 presents an overview of the selected metrics for evaluating testing process effectiveness and efficiency.

Table 1. *Metrics for Evaluating Testing Process Effectiveness and Efficiency*

Study	Metric	Description
Singh et al. (2008)	Test Effectiveness (TE)	Based on the ratio of closed defects to the total number of reported defects. It enables monitoring of the effectiveness of the testing process in detecting and resolving defects.
Singh et al. (2008)	Defect Removal Effectiveness	Determined by the ratio of defects of a particular type detected in the corresponding phase to the total number of such defects.
Singh et al. (2008)	Error Focus	Measures the proportion of reported errors that are successfully resolved and subsequently closed.
Singh et al. (2008)	Test Session Efficiency (TSE)	Relates the number of weighted defects identified during a specific testing phase to the number of added or modified lines of source code.

Table 2 presents an overview of the selected metrics for assessing test case and test code quality.

Table 2. *Metrics for Evaluating Test Case Quality*

Study	Quality Aspect	Metrics	Quality Aspect Assessed
Barraood et al. (2023)	Coverage and Effectiveness	Statement Coverage, Branch Coverage, Code Coverage	Coverage and effectiveness of tests.
Barraood et al. (2023)	Defect Detection	Mutation Score, Number of Detected Faults	The ability of tests to detect defects.
Barraood et al. (2018)	Test Case Characteristics	Test Case Size, Test Case Similarity, Test Case Cohesion, Test Case Coupling	Characteristics and quality of individual test cases.
Barraood et al. (2023, 2018)	Complexity and Maintainability	WMC, DIT, NOC	Complexity and maintainability of test code.

The reviewed studies indicate that a wide range of metrics has been proposed to evaluate different aspects of software testing. Coverage metrics are among the most frequently considered, while defect detection and mutation score are also widely addressed. Other metrics assess test case characteristics, as well as test code complexity and maintainability, while several studies examine the relationship between individual metrics and testing quality. However, individual metrics have limitations when used in isolation. No single metric provides a comprehensive assessment of testing quality, and code coverage alone does not fully reflect test effectiveness or quality. For some metrics, their relationship with specific quality characteristics remains unclear, while for some commonly assumed test case quality characteristics, there is insufficient consistent empirical evidence. Therefore, further research is needed to better relate individual metrics to specific quality characteristics and to examine combinations of different indicators for a more comprehensive assessment of testing quality.

CONCLUSION

This paper presents a preliminary overview of metrics used in the literature to evaluate software testing quality. Metrics related to the testing process and testing efficiency, as well as metrics for test case and test code quality, were considered. The review of the studies included indicates that testing quality can be viewed from different perspectives, including coverage, defect detection capability, mutation score, test structure and characteristics, testing time and testing costs.

It was particularly observed that no single metric can provide a complete picture of testing quality. Code coverage represents an important indicator of testing scope, but on its own, it does not provide information about the extent to which tests verify system behavior. Similar limitations were identified for other metrics, while empirical studies show that some characteristics commonly associated with test case quality are not always confirmed as indicators of their effectiveness. Therefore, a more comprehensive assessment requires consideration of multiple interconnected indicators.

Future research should focus on a more detailed analysis of the relationships between individual metrics and testing quality characteristics, as well as on examining combinations of different indicators for a more comprehensive assessment of testing quality. Particular attention could be given to combining different metrics and defining a set of indicators that would enable a more comprehensive assessment of testing quality across different software projects.

ACKNOWLEDGEMENT

This research was supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia, with a scholarship for PhD students in the year 2026.

REFERENCES

Barraood, S. O., Mohd, H., Baharom, F., & Omar, M. (2018). Quality factors of test cases: A systematic literature review. In Knowledge Management International Conference (KMICe) 2018, 25–27 July 2018, Miri, Sarawak, Malaysia

Barraood, S. O., Mohd, H., Baharom, F., & Almogahed, A. (2023). Systematic literature review on test case quality characteristics and metrics. In *2023 3rd International Conference on Emerging Smart Technologies and Applications (eSmarTA) (pp. 1–8). IEEE. https://doi.org/10.1109/ESMARTA59349.2023.10293544

Barraood, S. O., Mohd, H., Baharom, F., & Almogahed, A. (2023). Verifying agile black-box test case quality measurements: Expert review. IEEE Access, 11, 106987–107003. https://doi.org/10.1109/ACCESS.2023.3320576

Bowes, D., Hall, T., Petrić, J., Shippey, T., & Turhan, B. (2017). How good are my tests? In 2017 IEEE/ACM 8th Workshop on Emerging Trends in Software Metrics (WETSoM) (pp. 9–14). IEEE. https://doi.org/10.1109/WETSoM.2017.2

Do, P., & Morales, E. (2026, August 6). Development of quality metrics for measuring the efficiency and reliability of automated software testing processes.

Grano, G., De Iaco, C., Palomba, F., & Gall, H. C. (2020). Pizza versus pinsa: On the perception and measurability of unit test code quality. In 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 336–347). IEEE. https://doi.org/10.1109/ICSME46990.2020.00040

Jain, K., Kalburgi, G. T., Le Goues, C., & Groce, A. (2023). Mind the gap: The difference between coverage and mutation score can guide testing efforts. In 2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE) (pp. 102–113). IEEE. https://doi.org/10.1109/ISSRE59848.2023.00036

Jorgensen, P. C. (2014). Software testing: A craftsman's approach (4th ed.). CRC Press.

Lee, M. C., & Chang, T. (2013). Software measurement and software metrics in software quality. International Journal of Software Engineering and Its Applications, 7(4), 15–34.

Lucrédio, D., Vincenzi, A. M. R., de Almeida, E. S., & Ahmed, I. (2023). Test case quality: An empirical study on belief and evidence. arXiv. https://doi.org/10.48550/arXiv.2307.06410

Singh, Y., Kaur, A., & Suri, B. (2008). An empirical study of product metrics in software testing. In M. Iskander (Ed.), Innovative techniques in instruction technology, e-learning, e-assessment, and education (pp. 64–72). Springer. https://doi.org/10.1007/978-1-4020-8739-4_12

Zhang, P., Wang, Y., Liu, X., Yang, Y., Li, Y., Chen, L., Wang, Z., Sun, C.-A., & Zhou, Y. (2022). Test suite effectiveness metric evaluation: What do we know and what should we do? arXiv. https://doi.org/10.48550/arXiv.2204.09165