

A Comparative Analysis of the Effectiveness and Efficiency of Manual and Automated Testing in Software Quality Assurance

Amara Seviany Agustin¹, Nuur Wachid Abdul Majid^{1*}, Md Baharuddin Bin Abdul Rahman²

¹Information Systems and Technology Education, Universitas Pendidikan Indonesia, Bandung, Indonesia

²Pusat Pengkajian Ilmu Pendidikan, Universiti Sains Malaysia, 11800 Minden Penang, Malaysia

Abstract

Purpose: This study aims to examine respondents' perceptions of the effectiveness, bottlenecks, and reliability of automated and manual testing, as well as their perceived contributions to bug detection and software product quality.

Methods/Study design/approach: This study employed a quantitative survey approach involving 105 respondents selected through purposive sampling. The questionnaire used a five-point Likert scale. Method-specific analyses involved 90 respondents for each testing approach, while the comparative analysis involved 75 respondents experienced in both methods. Data were analyzed using descriptive statistics, the Wilcoxon Signed-Rank Test, and the Friedman Test.

Result/Findings: Automated testing received positive assessments for regression testing and repetitive execution (88.9% agreement), while manual testing received positive assessments for UI/UX evaluation and complex test scenarios (81.1% agreement). Respondents identified script maintenance and debugging as automated-testing bottlenecks, whereas manual-testing challenges included execution time, test coverage, documentation maintenance, and sprint time constraints. Flaky tests and human error were also reported as reliability concerns. Among respondents experienced in both methods, 88.0% agreed that automated testing was more accurate in detecting repetitive functional bugs, while 89.3% agreed that combining both approaches could help prevent escaped defects.

Novelty/Originality/Value: This study provides a perception-based evaluation of automated and manual testing across effectiveness, bottlenecks, and reliability. The findings suggest that respondents perceive complementary roles for the two approaches, with automated testing positively assessed for repetitive activities and manual testing for exploratory and interface-related evaluation. These insights may inform software-testing strategies, although actual defect detection performance and software product quality were not directly measured.

Keywords: Automated Testing, Manual Testing, Software Quality Assurance, Bottleneck, Escaped Defects

Received September 2026 / **Revised** September 2026 / **Accepted** September 2026

This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).



INTRODUCTION

The speed of digital system distribution has become a necessity, but accelerating the release cycle will be counterproductive if functional stability and product quality assurance are marginalized [1], [2]. To maintain a balance between release acceleration and system reliability, testing and quality control play an essential role. Software Quality Assurance (SQA) plays a crucial role that focuses not only on functional defect detection but also implements a systematic damage prevention mechanism before the system is deployed to the production environment [3], [4]. To realize this quality assurance role, software testing activities become the main operational instruments executed. In current industrial practices, the two main approaches most used to evaluate software quality are manual testing and automated testing [5], [6].

Manual testing, particularly through the exploratory testing approach, provides profound advantages in evaluating intuitive aspects such as user interface and user experience (UI/UX), as well as handling dynamic and complex test cases [7], [8]. However, as the application scale and sprint iteration frequency increase, manual testing faces major challenges related to time efficiency, result consistency due to potential human error, and high resource dependency when executing tests repetitively [9], [10].

On the other hand, automated testing emerges as a primary solution to accelerate regression testing and repetitive test execution within the Continuous Integration and Continuous Deployment (CI/CD) scheme [11], [12], [13]. Despite offering execution speed, the implementation of this automation faces significant technical challenges [14]. Designing, debugging, and maintaining the test suite as the application evolves

*Corresponding author.

Email addresses: nuurwachid@upi.edu (Majid)

DOI: 10.15294/rji.v4i2.64137

demand substantial time and effort [15] . Additionally, there is a risk of flaky tests, which are inconsistent test failures without any modifications to the source code [16], [17], [18], [19] . These challenges may create bottlenecks in the software release process, particularly when organizations rely exclusively on one testing approach [20], [21].

A few researchers have focused on the individual benefits and technical challenges of either manual or automated testing independently. However, there have been limited studies concerned with comprehensively and empirically comparing the effectiveness, efficiency, and bottlenecks of both methods concurrently within an agile software development environment. Therefore, this research intends to bridge this gap by empirically evaluating the comparative performance of these two testing methods. The objectives of this research are specifically categorized into three main focuses: (1) to analyze the effectiveness, bottlenecks, and reliability of Automated Testing implementation; (2) to analyze the effectiveness, bottlenecks, and reliability of Manual Testing implementation; and (3) to compare the effectiveness of both methods in detecting bugs to prevent escaped defects in the production environment.

Through this survey-based quantitative approach, the research results are expected to provide theoretical contributions by enriching the literature regarding software testing evaluation. Furthermore, this study aims to offer practical guidelines for QA Engineers, Test Leads, and software developers in determining the most optimal and efficient proportion of manual and automated testing implementation [22] .

METHODS

This study employed a quantitative approach using a survey method to obtain and analyze data systematically. The use of questionnaire-based surveys is one of the approaches that can be utilized in quantitative research to collect and analyze data in a structured manner [23] . In the context of software testing, surveys of practitioners have been used to identify perceptions regarding the factors that influence test flakiness [24] . Furthermore, research on testing practitioners indicates that the manual testing process can be evaluated based on aspects such as fault detection likelihood, test feedback time, and test creation effort [25], [26] . Meanwhile, in automated testing, effective automation strategies, a good testing environment, and test script quality are among the practices related to improving test automation maturity [27], [28] . Based on these foundations, this study measures the perceptions and experiences of practitioners regarding the effectiveness, efficiency, bottlenecks, and reliability (flakiness) in the implementation of manual testing and automated testing.

The target population of this study comprised two main groups: technology industry practitioners, including QA Engineers, QA Analysts, Test Leads, and Software Developers who perform self-testing, as well as academics or students from information technology-related fields with direct experience in software testing. Respondents were recruited using a non-probability sampling technique with a purposive sampling approach to ensure their relevance to the research objectives. Purposive sampling has also been applied by Lima et al. [29] in a survey involving software-testing professionals, while Cui et al. [30] examined participant selection in crowdsourced software-testing activities.

Respondents were selected based on their practical experience in designing or executing manual test cases or automated test scripts and their direct involvement in software development and testing activities, particularly within sprint or Agile environments. This approach aligns with Rainer et al. [31], who emphasized the importance of recruiting appropriate and credible participants in empirical software engineering research. The inclusion of academic and industry participants is also consistent with Barraod et al. [32], who involved both groups in research concerning test case quality in Agile environments. A total of 105 respondents participated in this study, comprising 15 respondents experienced exclusively in automated testing, 15 experienced exclusively in manual testing, and 75 experienced in both approaches. Accordingly, the automated-testing and manual-testing analyses each included 90 eligible respondents, while the comparative analysis involved 75 respondents with experience in both methods.

Data collection was conducted through an online questionnaire distributed using Google Forms. The questionnaire employed a five-point Likert scale, ranging from 1 (Strongly Disagree) to 5 (Strongly Agree), to measure respondents' perceptions of manual and automated testing. The use of a five-point Likert scale in software-testing research was also applied by Barraod et al. [32] in their investigation of test case quality in Agile environments.

The questionnaire was adapted from the study by Haas et al. [26], which investigated manual and automated testing processes in industrial practice. The original instrument consisted primarily of process-oriented questions concerning testing activities, test suite execution and maintenance effort, operational bottlenecks, and flaky test cases. In the present study, relevant questions were selected, reformulated into declarative statements, and adjusted to the research objectives and five-point Likert response format. These modifications enabled respondents to express their level of agreement with statements concerning testing effectiveness, operational challenges, and reliability rather than providing the descriptive process information requested in the original questionnaire.

The automated-testing items were adapted from questions concerning automated test activities, test suite maintenance effort, process bottlenecks, and flaky automated test cases. The testing-activity question was reformulated to assess the perceived effectiveness of automation for regression testing and repetitive execution. The maintenance-effort question was adapted to assess the time and effort required to maintain automated test scripts, while the general bottleneck question was specified to address difficulties in designing and debugging scripts. The original question concerning flaky automated test cases was reformulated into a statement about inconsistent test results. For manual testing, questions concerning manual test activities, the overall test suite, execution cycles, maintenance effort, and process bottlenecks were adapted into statements addressing exploratory testing for UI/UX and complex cases, difficulty achieving full test coverage, relative execution time, test case documentation maintenance, and time constraints during sprints. The indicator concerning human error was developed for the present study to assess a potential source of inconsistent manual-testing results; it was not treated as a direct reproduction of the original question concerning flaky manual test cases. In addition, the two comparative items concerning repetitive functional bug detection and the perceived contribution of combining both testing methods to preventing escaped defects were developed specifically to address this study's comparative research objective. The adapted and newly developed questionnaire items are presented in Table 1.

Table 1. Adaptation of the Testing Evaluation Questionnaire

Category	Aspect / Indicator	Questionnaire Item
Automated Testing Evaluation	Effectiveness	Automated testing is highly effective in handling regression testing activities and repetitive test execution on applications.
	Efficiency	Maintaining automated test scripts (test suite maintenance) requires high time and effort when feature/code changes occur.
	Bottleneck	The process of designing and debugging automated test scripts often becomes a bottleneck in smooth feature releases.
	Reliability	Automated test scripts in our project frequently produce inconsistent reports or often fail (flaky test cases).
Manual Testing Evaluation	Effectiveness	Manual exploratory testing is much more effective in evaluating user interfaces (UI/UX) and complex UI cases.
	Efficiency (Scope)	Executing the entire scope of test scenarios manually in every release cycle is very difficult and consumes a lot of resources.
	Efficiency (Time)	The time required to complete all manual scenarios is relatively longer compared to automated testing.
	Efficiency (Maintenance)	Maintaining and updating manual test case documents (test cases) requires high time and effort.
	Bottleneck	The limited execution time for manual testing often becomes a major bottleneck in sprints.
Overall Effectiveness Comparison	Reliability	Manual testing is susceptible to potential human errors, which cause execution results to be less consistent.
	Bug Detection	Automated testing is more accurate in detecting repetitive functional bugs compared to manual testing.
	Defect Prevention	The combination of manual and automated methods can significantly prevent bugs from escaping into the production environment (escaped defects).

The questionnaire items were subjected to expert review by the research supervisor to assess their suitability for the research objectives and the software-testing context. Internal consistency was subsequently assessed using Cronbach's alpha for the two multi-item bottleneck groups. The reliability results are presented in Table 2. Both groups demonstrated acceptable internal consistency, with Cronbach's alpha values of 0.777 for automated-testing bottlenecks and 0.915 for manual-testing bottlenecks. Cronbach's alpha was not calculated for the single-item indicators or the two comparative statements, as they assess distinct aspects of testing. Therefore, the reported reliability coefficients apply specifically to the two bottleneck groups rather than to the questionnaire.

Table 2. Internal Consistency Reliability of the Bottleneck Indicators

Indicator Group	Number of Items	Valid N	Cronbach's Alpha	Corrected Item-Total Correlation
Automated Testing Bottlenecks	2	90	0.777	0.636
Manual Testing Bottlenecks	4	90	0.915	0.793 – 0.821

The collected quantitative data were processed using statistical analysis software. Descriptive statistical analysis was used to summarize the data through mean values, medians, standard deviations, percentages, and frequency distributions. Since the questionnaire responses were measured using a five-point Likert scale, nonparametric statistical tests were applied for inferential analysis. The Wilcoxon Signed-Rank Test was used to determine whether respondents' assessments of the effectiveness and reliability of Automated Testing and Manual Testing differed significantly from the neutral value of 3. The Wilcoxon Signed-Rank Test was also used to compare the two bottleneck indicators of Automated Testing. Meanwhile, the Friedman Test was used to compare the four bottleneck indicators of Manual Testing because the indicators were assessed by the same respondents. For the comparative analysis, the Wilcoxon Signed-Rank Test was used to evaluate respondents' assessments of the superiority of Automated Testing in detecting repetitive functional bugs and the ability of combining Automated Testing and Manual Testing to prevent escaped defects. All hypothesis tests were conducted at a 5% significance level ($\alpha = 0.05$).

RESULT AND DISCUSSION

Respondent Profile

This study involved a total of 105 respondents with diverse backgrounds in software development and testing. The respondents comprised 15 individuals with experience exclusively in automated testing, 15 with experience exclusively in manual testing, and 75 with experience in both methods. Based on this composition, the automated-testing and manual-testing analyses each included 90 eligible respondents, whereas the comparative analysis involved only the 75 respondents experienced in both approaches. The respondent profile was further analyzed based on professional roles, length of involvement in software testing, organizational characteristics, and testing methods experienced.

Respondent Profile Based on Role

The respondent profile distribution is presented in Figure 1.

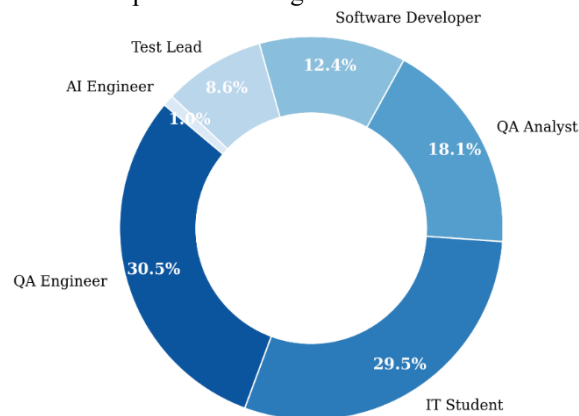


Figure 1. Respondent Profile Distribution

As illustrated in Figure 1, the respondents are predominantly comprised of QA Engineers (30.5%) and IT Students (29.5%), followed by QA Analysts (18.1%). This composition indicates that the majority of the participants are directly and actively engaged in software development and testing activities. The inclusion of respondents holding positions from QA Analysts to Test Leads provides a comprehensive perspective across various levels of involvement and strategic decision-making in the software testing lifecycle.

Respondent Profile Based on Organizational Characteristics

Respondents' organizational characteristics were analyzed based on the institutional affiliations reported in the questionnaire. The organizations were classified into two categories: higher education institutions and companies or corporate organizations. As presented in Table 3, 65 respondents (61.90%) were affiliated with companies or corporate organizations, while 40 respondents (38.10%) were affiliated with higher

education institutions. This distribution provides an overview of the organizational backgrounds represented in the sample and helps contextualize respondents' perceptions of manual and automated testing.

Table 3. Respondent Distribution by Organizational Type

Organizational Type	Frequency	Percentage
Higher Education Institutions	40	38,10%
Companies / Corporate Organizations	65	61,90%
Total	105	100%

Respondent Profile Based on Length of Involvement in Software Testing

The respondent distribution by length of involvement is presented in Table 4.

Table 4. Respondent Profile Based on Length of Involvement

Length of Involvement	Frequency	Percentage
< 1 year	26	24,76%
1 - 3 years	37	35,24%
3 - 5 years	12	11,43%
> 5 years	30	28,57%

The distribution in Table 4 demonstrates that this study incorporates participants with a highly diverse range of experience. It encompasses entry-level testers (under one year) to highly experienced practitioners (over five years). This diverse level of expertise ensures that the study captures robust perceptions based on varying degrees of exposure to software testing challenges and environments.

Respondent Profile Based on Experienced Testing Methods

The respondents' experience with different software testing methods is presented in Table 5.

Table 5. Respondent Profile Based on Experienced Testing Methods

Testing Method	Frequency	Percentage
Automated Testing Only	15	14,29%
Manual Testing Only	15	14,29%
Both Methods	75	71,43%

The vast majority of respondents (71.43%) possess hands-on experience utilizing both automated and manual testing methods. The predominance of participants with dual-method experience is highly relevant to the study's core objective, particularly in facilitating a valid comparative analysis.

Analysis of Automated Testing Implementation

The automated-testing analysis was conducted using responses from 90 participants with relevant experience, comprising 15 respondents experienced exclusively in automated testing and 75 respondents experienced in both automated and manual testing. This analysis focused on respondents' perceptions of automated testing effectiveness, operational bottlenecks, and reliability.

Effectiveness of Automated Testing

Descriptive statistical analysis reveals that the effectiveness of automated testing for regression testing and repetitive execution yielded a mean score of 4.33 (SD = 1.071). A total of 27 respondents (30.0%) agreed, and 53 respondents (58.9%) strongly agreed. Consequently, an overwhelming 88.9% of respondents provided a positive assessment of automated testing's efficacy in these scenarios, aligning with findings on automated UI validations [33]. Furthermore, the Wilcoxon Signed-Rank Test demonstrated a highly significant result ($Z = -7.002$, $p < 0.001$). Statistically, 80 respondents scored above the neutral threshold (3), confirming that practitioners significantly acknowledge the superiority and effectiveness of automated testing for regression.

Bottlenecks in Automated Testing

The indicators used to assess bottlenecks in automated testing are presented in Table 6.

Table 6. Indicators of Bottlenecks in Automated Testing

Bottleneck Indicator	Mean	Median	Std. Deviation	Agree + Strongly Agree
Test script maintenance requires high time and effort	4.09	4.00	1.167	80.0%
Designing and debugging scripts create release bottlenecks	3.92	4.00	1.220	71.1%

Bottlenecks in automated testing were analyzed through two primary indicators: script maintenance and the design/debugging phase. Script maintenance obtained a mean of 4.09 (80.0% agreement), which is consistent with the recognized challenges of sequential feature modifications [34]. Designing and debugging yielded a mean of 3.92 (71.1% agreement). However, the Wilcoxon Signed-Rank Test resulted in $Z = -1.673$ and $p = 0.094$. Since $p > 0.05$, there is no statistically significant difference between the two indicators. Thus, both script maintenance and the debugging process are considered equally significant bottlenecks.

Reliability of Automated Testing

The reliability of automated testing was assessed based on the consistency of automated test results, as presented in Table 7.

Table 7. Reliability Assessment of Automated Testing

Statistic	Result
Mean	3.71
Median	4.00
Mode	5
Std. Deviation	1.326
Agree + Strongly Agree	64.4%

Regarding reliability, the indicator concerning "flaky tests" or inconsistent automated test results obtained a mean of 3.71 (Table 7). Notably, 64.4% of respondents agreed or strongly agreed that automated testing is prone to producing inconsistent results. The Wilcoxon Signed-Rank Test confirmed this perception significantly ($Z = -4.679$, $p < 0.001$). This signifies the persistent challenge of test flakiness, driving recent initiatives in adaptive test healing utilizing graph neural networks and language models [35], [36].

Analysis of Manual Testing Implementation

The manual-testing analysis included 90 respondents with relevant experience, consisting of 15 respondents experienced exclusively in manual testing and 75 respondents experienced in both testing methods. The analysis examined respondents' perceptions of manual testing effectiveness, operational bottlenecks, and reliability.

Effectiveness of Manual Testing

The effectiveness of manual testing in evaluating UI/UX and handling complex and dynamic test cases is presented in Table 8.

Table 8. Effectiveness of Manual Testing

Statistic	Result
Number of Respondents	90
Mean	4.13
Median	4.00
Std. Deviation	1.073
Agree + Strongly Agree	81.1%

The effectiveness of manual testing in evaluating UI/UX and complex dynamic test cases obtained a mean of 4.13 (Table 8). A significant majority (81.1%) of respondents agreed on its efficacy in these areas, an assertion supported by recent studies on visual inspection and exploratory tasks [37], [38], [39]. The Wilcoxon Signed-Rank Test yielded $Z = -6.675$ with $p < 0.001$, proving that respondents significantly value manual testing when handling complex interfaces and exploratory scenarios.

Bottlenecks in Manual Testing

The indicators used to assess bottlenecks in manual testing are presented in Table 9.

Table 9. Indicators of Bottlenecks in Manual Testing

Bottleneck Indicator	Mean	Median	Std. Deviation	Agree + Strongly Agree
Difficulty in executing entire test coverage	4.03	4.00	1.096	81.1%
Longer relative execution time	4.13	4.00	1.104	82.2%
Maintenance of manual test case documents	4.02	4.00	1.060	81.1%
Execution time constraints during sprints	4.07	4.00	1.100	84.4%

Descriptive analysis revealed that all four indicators of manual testing bottlenecks received relatively high ratings, with mean scores ranging from 4.02 to 4.13 (Table 9). To examine whether respondents rated these bottlenecks differently, a Friedman Test was conducted using 90 complete responses. The test revealed no statistically significant differences among the four indicators ($\chi^2(3) = 2.235$, $p = 0.525$). Although longer execution time obtained the highest descriptive mean (4.13) and mean rank (2.63), the results do not provide sufficient evidence that it was rated significantly differently from the other bottlenecks.

Reliability of Manual Testing

The reliability of manual testing was assessed based on the risk of human error leading to inconsistent outcomes, as presented in Table 10.

Table 10. Reliability Assessment of Manual Testing

Statistic	Result
Number of Respondents	90
Mean	4.10
Median	4.00
Std. Deviation	1.152
Agree + Strongly Agree	84.4%

The reliability of manual testing was scrutinized based on the risk of human error leading to inconsistent outcomes. This indicator achieved a mean of 4.10, with 84.4% of respondents affirming the risk (Table 10). The Wilcoxon Signed-Rank Test further validated this concern ($Z = -6.092$, $p < 0.001$), emphasizing that human error remains a formidable threat to the consistency and reliability of manual testing.

Comparative Analysis of Automated and Manual Testing

The comparative analysis involved 75 respondents with experience in both automated and manual testing, focusing on automated testing accuracy for repetitive functional bugs and the combined methods' potential to prevent escaped defects. Respondents' agreement on testing effectiveness is illustrated in Figure 2.

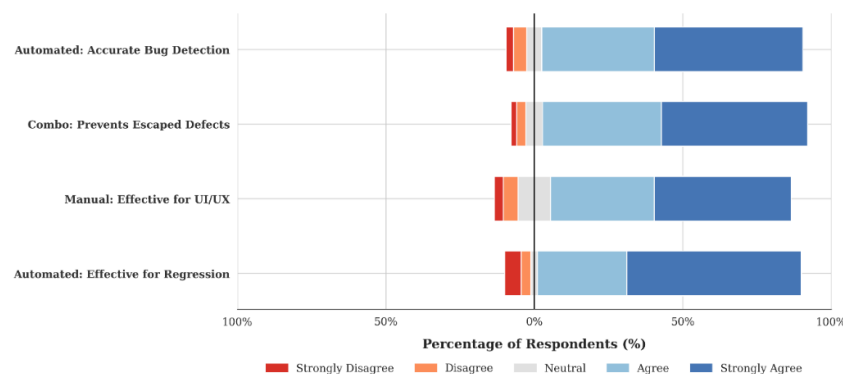


Figure 2. Distribution of Agreement on Testing Effectiveness

When comparing the two methods, 88.0% of respondents agreed or strongly agreed that automated testing is more accurate than manual testing in detecting recurring functional bugs (Mean = 4.17, SD = 1.070). This corroborates studies emphasizing optimization techniques in GUI testing and search-based detection [40], [41], [42]. As visualized in Figure 2, the diverging distribution highlights a strong consensus among practitioners favoring automation for repetitive accuracy. The Wilcoxon Signed-Rank Test corroborated this consensus ($Z = -6.069$, $p < 0.001$), indicating a highly significant statistical tendency.

The goal of SQA is to prevent escaped defects in the production environment. Analysis shows an overwhelming 89.3% of respondents affirming that a strategic combination of manual and automated

testing is highly capable of mitigating escaped defects (Mean = 4.23, SD = 1.021). The Wilcoxon Signed-Rank Test proved this to be highly significant ($Z = -6.241$, $p < 0.001$). This complements contemporary literature promoting hybrid and concolic testing architectures to eliminate hidden concurrency bugs [43], [44], [45], [46], [47]. The comparative distribution of testing bottlenecks is illustrated in Figure 3.

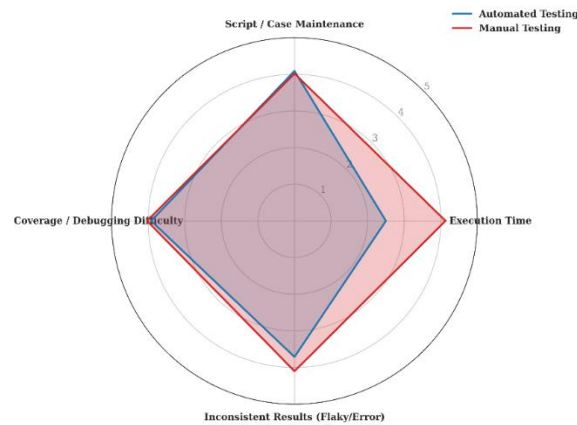


Figure 3. Comparative Footprint of Testing Bottlenecks

Figure 3 provides a visual overview of the challenges associated with automated and manual testing, covering script or test case maintenance, execution time, inconsistent results, and coverage or debugging difficulties. These categories represent different questionnaire indicators for each testing approach and should not be interpreted as directly equivalent measures. The survey results indicated that respondents perceived script maintenance and debugging as operational challenges in automated testing, while manual testing was associated with difficulties in achieving full test coverage, longer execution time, test case documentation maintenance, and sprint time constraints. Respondents also identified flaky tests and human error as reliability concerns in automated and manual testing, respectively. Although the figure illustrates the different challenges associated with the two approaches, no direct statistical comparison of their overall bottleneck levels was conducted. Therefore, the visualization should be interpreted as a descriptive overview of respondents' perceptions rather than evidence that one testing method experiences significantly greater challenges than the other.

CONCLUSION

This study examined respondents' perceptions of the effectiveness, bottlenecks, and reliability of automated and manual testing, as well as their perceived contributions to bug detection and software product quality. Automated testing received positive assessments for regression testing and repetitive execution, with 88.9% of respondents expressing agreement and a statistically significant Wilcoxon Signed-Rank Test result relative to the neutral value ($p < 0.001$). Respondents also identified script maintenance and debugging as operational bottlenecks, although no statistically significant difference was found between these two indicators ($p = 0.094$). Furthermore, 64.4% of respondents agreed that flaky tests represented a reliability concern ($p < 0.001$). Manual testing received positive assessments for UI/UX evaluation and complex test scenarios, with 81.1% agreement ($p < 0.001$). Its reported operational challenges included longer execution times, difficulties achieving full test coverage, test case maintenance, and sprint time constraints. The Friedman Test revealed no statistically significant differences among these four bottleneck indicators ($\chi^2(3) = 2.235$, $p = 0.525$). In addition, 84.4% of respondents agreed that human error could affect the consistency of manual testing results ($p < 0.001$). Among respondents experienced in both approaches, 88.0% agreed that automated testing was more accurate in detecting repetitive functional bugs ($p < 0.001$), while 89.3% agreed that combining manual and automated testing could help prevent escaped defects ($p < 0.001$). Overall, the findings suggest that respondents perceived complementary roles for the two testing approaches: automated testing was positively assessed for repetitive testing activities, whereas manual testing was positively assessed for exploratory and interface-related evaluation. These findings provide perception-based insights that may inform software-testing strategies, although actual differences in defect detection performance and software product quality were not directly measured. This study has several limitations. The use of purposive sampling restricts the statistical generalizability of the findings to the broader population of software-testing practitioners. Although the sample included respondents from different professional and organizational backgrounds and with varying levels of testing experience, geographic information was not systematically collected. Furthermore, the findings were based on self-

reported questionnaire responses rather than objective project-level measurements of test execution time, defect detection rates, or escaped defects. Consequently, the results should be interpreted as reflecting respondents' perceptions rather than as direct evidence of differences in the actual performance of the two testing methods. Future research should involve larger and more geographically diverse samples across different organizational contexts and incorporate objective testing metrics to further evaluate the effectiveness, efficiency, reliability, and combined use of automated and manual testing.

REFERENCES

- [1] D. Kato, "Quality Control Methods Using Quality Characteristics in Development and Operations," *Journal of Software Quality Control*, pp. 232–243, 2024.
- [2] G. Kumari, M. K. Shukla, N. Kumar, R. Kumar, M. S. Adhikari, and S. Singh, "Predicting Test Automation Failures using Machine Learning for Proactive Quality Engineering," in *6th International Conference on Inventive Computation and Information Technologies, ICICIT 2026 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., 2026, pp. 1199–1203. doi: 10.1109/ICICIT69063.2026.11634232.
- [3] A. N. Pribadi, S. I. Nuraini, and R. Anggitaningsih, "Peran Quality Assurance Dalam Upaya Penginputan Data Kualitas Benih," *Jurnal Penelitian Nusantara*, vol. 1, no. 2, pp. 189–196, 2025.
- [4] S. Reine De Reanzi and P. Ranjit Jeba Thangaiyah, "A survey on software test automation return on investment, in organizations predominantly from Bengaluru, India," *International Journal of Engineering Business Management*, vol. 13, 2021, doi: 10.1177/18479790211062044.
- [5] A. R. Rambe and H. Prihantoro, "Pengujian Otomatis Aplikasi Mobile dengan Teknik Black-box Menggunakan Appium (Studi Kasus: Pengembangan Aplikasi Jala Mobile)," *Jurnal Automata*, vol. 3, no. 2, pp. 1–6, 2022.
- [6] C. Zhai, H. Yu, and Z. Liang, "Design and Implementation of Automated Testing for Medical Web Platform Based on Cypress," *China Medical Devices*, vol. 39, no. 3, pp. 14–19, 2024, doi: 10.3969/j.issn.1674-1633.2024.03.003.
- [7] J. Itkonen, M. V. Mäntylä, and C. Lassenius, "The role of the tester's knowledge in exploratory software testing," *IEEE Transactions on Software Engineering*, vol. 39, no. 5, pp. 707–724, 2013, doi: 10.1109/TSE.2012.55.
- [8] A. Lama, "A Comparative Study of Manual and Automated Software Testing Strategies in Agile and DevOps Environments," *International Journal of Computer Application*, vol. 15, no. 4, pp. 64–68, 2025, doi: 10.5281/zenodo.15844142.
- [9] K. Ekasakti and D. N. Pratomo, "Analisis Evaluasi Kinerja Pengujian Manual Dan Otomatis Pada Sistem Informasi Properti (Propertio.id)," *Journal of Internet and Software Engineering*, vol. 6, no. 2, pp. 88–95, 2025, doi: 10.22146/jise.v6i2.16934.
- [10] J. Yu, "Design and Application on Agile Software Exploratory Testing Model," in *Proceedings of 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference, IMCEC 2018*, B. Xu, Ed., Institute of Electrical and Electronics Engineers Inc., 2018, pp. 2082–2088. doi: 10.1109/IMCEC.2018.8469368.
- [11] A. Chatterjee, A. Bala, M. Shah, and A. H. Nagappa, "CTAF: Centralized Test Automation Framework for multiple remote devices using XMPP," in *INDICON 2018 - 15th IEEE India Council International Conference*, Institute of Electrical and Electronics Engineers Inc., 2018. doi: 10.1109/INDICON45594.2018.8987182.
- [12] F. D. Wicaksono and S. Rani, "Rancang Bangun Automation Test Journey pada E-Commerce (Studi Kasus: Marketplace PT. Tokopedia)," *AUTOMATA*, vol. 3, no. 2, Aug. 2022. [Online]. Available: <https://journal.uui.ac.id/AUTOMATA/article/view/24157>
- [13] B. Manchuri, "Energy-Efficient Test Automation in Banking: Towards Sustainable CI/CD Pipelines," in *Lecture Notes in Networks and Systems*, J. C. Bansal, P. Jamwal, and S. Hussain, Eds., Springer Science and Business Media Deutschland GmbH, 2026, pp. 225–242. doi: 10.1007/978-3-032-22914-4_18.
- [14] B. S. Ahmed, A. Gargantini, and M. Bures, "An Automated Testing Framework for Smart TV apps Based on Model Separation," in *Proceedings - 2020 IEEE 13th International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2020*, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 62–73. doi: 10.1109/ICSTW50294.2020.00026.
- [15] C. Paduraru, M. Cernat, and A.-N. Staicu, "A Unified Framework for Automated Testing of Robotic Process Automation Workflows Using Symbolic and Concolic Analysis †," *Machines*, vol. 13, no. 6, 2025, doi: 10.3390/machines13060504.
- [16] P. Dai, X. Ma, Y. Zhao, and Y. Gong, "Heterogeneous Graph Transformer with Multi-View Representation Learning for Flaky Test Detection," *Computers*, vol. 15, no. 6, 2026, doi: 10.3390/computers15060372.
- [17] M. Elgazzar, E. Hossny, and F. A. Omara, "A survey of Detecting Flakiness in Automated Test Regression Suite," in *21st International Learning and Technology Conference: Reality and Science Fiction in Education, L and T 2024*, A. Sarirete, E.-A. M. M. F., P. M. Elkafrawy, Z. Balfagih, and T. Brahimi, Eds., Institute of Electrical and Electronics Engineers Inc., 2024, pp. 330–336. doi: 10.1109/LT60077.2024.10469624.
- [18] M. Gruber and G. Fraser, "A Survey on How Test Flakiness Affects Developers and What Support They Need To Address It," in *Proceedings - 2022 IEEE 15th International Conference on Software Testing, Verification and Validation, ICST 2022*, 2022, pp. 82–92. doi: 10.1109/ICST53961.2022.00020.
- [19] O. Parry, G. M. Kapfhammer, M. Hilton, and P. McMin, "A survey of flaky tests," *ACM Transactions on Software Engineering and Methodology*, vol. 31, no. 1, 2021, doi: 10.1145/3476105.
- [20] L. Ditasari and T. Raharjo, "Generative AI-Based Software Testing Implementation Model in Agile Development Methodology: Systematic Literature Review," *Journal of Software Engineering*, vol. 8, no. 3, pp. 1530–1544, 2026.
- [21] A. Sezgin, G. Ozkan, and E. Cosgun, "Leveraging Large Language Models in Software Testing: A Review of Applications and Challenges," in *ISDFS 2025 - 13th International Symposium on Digital Forensics and Security*, Institute of Electrical and Electronics Engineers Inc., 2025. doi: 10.1109/ISDFS65363.2025.11011986.
- [22] U. K. A. Sethupathy, "DESIGNING A LEAN-AGILE & DEVOPS MATURITY MODEL FOR CONTINUOUS IMPROVEMENT," *Social Science Forum*, vol. 8, no. 7, pp. 86–93, 2024, doi: 10.10118/SSFJ/2408710.
- [23] M. Lu, T. Cui, Z. Huang, H. Zhao, T. Li, and K. Wang, "A Systematic Review of Questionnaire-Based Quantitative Research on MOOCs," *The International Review of Research in Open and Distributed Learning*, vol. 22, no. 2, pp. 285–313, 2021, doi: 10.19173/irrodl.v22i2.5208.
- [24] A. Ahmad, O. Leifler, and K. Sandahl, "Empirical analysis of practitioners' perceptions of test flakiness factors," *Software Testing Verification and Reliability*, vol. 31, no. 8, pp. 1–24, 2021, doi: 10.1002/stvr.1791.

- [25] I. Dobles, A. Martinez, and C. Quesada-Lopez, "Comparing the effort and effectiveness of automated and manual tests: An industrial case study," in *Iberian Conference on Information Systems and Technologies, CISTI*, A. Rocha, I. Pedrosa, M. P. Cota, and R. Goncalves, Eds., IEEE Computer Society, 2019. doi: 10.23919/CISTI.2019.8760848.
- [26] R. Haas, D. Elsner, E. Juergens, A. Pretschner, and S. Apel, "How can manual testing processes be optimized? developer survey, optimization guidelines, and case studies," in *ESEC/FSE 2021 - Proceedings of the 29th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 1281–1291. doi: 10.1145/3468264.3473922.
- [27] B. Wang, L. Zhu, D. Sun, W. Wang, S. Song, and S. Peng, "Towards an Automated Testing Framework for IoT Devices," in *Journal of Physics: Conference Series*, Institute of Physics, 2023. doi: 10.1088/1742-6596/2493/1/012023.
- [28] Y. Wang, M. V. Mäntylä, Z. Liu, J. Markkula, and P. Raulamo-jurvanen, "Improving test automation maturity: A multivocal literature review," *Software Testing Verification and Reliability*, vol. 32, no. 3, pp. 1–33, 2022. doi: 10.1002/stvr.1804.
- [29] D. L. Lima, R. De Souza Santos, G. P. Garcia, S. S. Da Silva, C. Franca, and L. F. Capretz, "Software Testing and Code Refactoring: A Survey with Practitioners," in *Proceedings - 2023 IEEE International Conference on Software Maintenance and Evolution, ICSME 2023*, 2023, pp. 500–507. doi: 10.1109/ICSME58846.2023.00064.
- [30] Q. Cui, J. Wang, G. Yang, M. Xie, Q. Wang, and M. Li, "Who Should Be Selected to Perform a Task in Crowdsourced Testing?," in *Proceedings - International Computer Software and Applications Conference*, S. Reisman, S. I. Ahamed, C. Demartini, T. Conte, L. Liu, W. Claycomb, M. Nakamura, E. Tovar, S. Cimato, L. C.-H., H. Takakura, Y. J.-J., T. Akiyama, Z. Zhang, and K. Hasan, Eds., IEEE Computer Society, 2017, pp. 75–84. doi: 10.1109/COMPSAC.2017.265.
- [31] A. Rainer and C. Wohlin, "Recruiting credible participants for field studies in software engineering research," *Inf. Softw. Technol.*, vol. 151, p. 107002, 2022. doi: 10.1016/j.infsof.2022.107002.
- [32] S. O. Barraood, H. Mohd, and F. Baharom, "An initial investigation of the effect of quality factors on Agile test case quality through experts' review," *Cogent Eng.*, vol. 9, no. 1, 2022. doi: 10.1080/23311916.2022.2082121.
- [33] B. Jiang, "The Application of Vue.js Framework Technology in Multidomain Automated Testing Systems," *Automatic Control and Computer Sciences*, vol. 59, no. 4, pp. 455–466, 2025. doi: 10.3103/S0146411625700610.
- [34] R. Almeida, S. Nogueira, and A. Sampaio, "Combining sequential feature test cases to generate sound tests for concurrent features," *Sci. Comput. Program.*, vol. 250, 2026. doi: 10.1016/j.scico.2025.103414.
- [35] N. Mani and S. Attaranasl, "Enhancing Adaptive Test Healing with Graph Neural Networks for Dependency-Aware Decision Making," in *Proceedings - 2025 IEEE International Conference on Artificial Intelligence Testing, AITest 2025*, Institute of Electrical and Electronics Engineers Inc., 2025, pp. 126–133. doi: 10.1109/AITest66680.2025.00023.
- [36] N. Mani and S. Attaranasl, "Adaptive Test Healing using LLM/GPT and Reinforcement Learning," in *2025 IEEE International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2025*, A. R. Fasolino, S. Panichella, A. Aleti, and A. Mesbah, Eds., Institute of Electrical and Electronics Engineers Inc., 2025, pp. 9–16. doi: 10.1109/ICSTW64639.2025.10962516.
- [37] Z. Khaliq, S. U. Farooq, and D. A. Khan, "A deep learning-based automated framework for functional User Interface testing," *Inf. Softw. Technol.*, vol. 150, 2022. doi: 10.1016/j.infsof.2022.106969.
- [38] W. Lu, A. Senchenko, A. Sayle, A. Hindle, and C.-P. Bezemmer, "How Far Can VLMs Go for Visual Bug Detection? Studying 19,738 Keyframes from 41 Hours of Gameplay Videos," in *FSE Companion 2026 - Proceedings of the 34th ACM International Conference on the Foundations of Software Engineering*, S. H. Tan and F. Khomh, Eds., Association for Computing Machinery, Inc, 2026, pp. 345–350. doi: 10.1145/3803437.3805208.
- [39] A. Myrria *et al.*, "Leveraging Template Matching for Efficient Android Screenshot Visual Inspection," in *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*, Institute of Electrical and Electronics Engineers Inc., 2026. doi: 10.1109/ICCE67443.2026.11449809.
- [40] A. Perera, "Using Defect Prediction to Improve the Bug Detection Capability of Search-Based Software Testing," in *Proceedings - 2020 35th IEEE/ACM International Conference on Automated Software Engineering, ASE 2020*, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 1170–1174. doi: 10.1145/3324884.3415286.
- [41] S. Wongkampoo and S. Kiattisin, "Atom-Task Precondition Technique to Optimize Large Scale GUI Testing Time based on Parallel Scheduling Algorithm," in *ICSEC 2017 - 21st International Computer Science and Engineering Conference 2017, Proceeding*, Institute of Electrical and Electronics Engineers Inc., 2018, pp. 229–232. doi: 10.1109/ICSEC.2017.8443913.
- [42] X. Wu *et al.*, "Widget Detection-based Testing for Industrial Mobile Games," in *Proceedings - International Conference on Software Engineering*, IEEE Computer Society, 2023, pp. 173–184. doi: 10.1109/ICSE-SEIP58684.2023.00021.
- [43] L. Borzacchiello, E. Coppa, and C. Demetrescu, "FUZZOLIC: Mixing fuzzing and concolic execution," *Comput. Secur.*, vol. 108, 2021. doi: 10.1016/j.cose.2021.102368.
- [44] V. Chawla, "Enhancing Security and Performance in PyTorch: A Hybrid Fuzzing Approach," in *Proceedings of the 1st International Symposium on Parallel Computing and Distributed Systems, PCDS 2024*, C. W. Tan and T. H. Teo, Eds., Institute of Electrical and Electronics Engineers Inc., 2024. doi: 10.1109/PCDS61776.2024.10743720.
- [45] D. Parygina, A. Vishnyakov, and A. Fedotov, "Strong Optimistic Solving for Dynamic Symbolic Execution," in *Proceedings - 2022 Ivannikov Memorial Workshop, IVMEM 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 43–53. doi: 10.1109/IVMEM57067.2022.9983965.
- [46] M. Staron and S. Abrahão, "Hybrid Classical-AI Systems for Software Testing and Bug Fixing," *IEEE Softw.*, vol. 42, no. 6, pp. 106–110, 2025. doi: 10.1109/MS.2025.3597698.
- [47] J. Yang and W. K. Chan, "Dynamic Testing Against Hidden Concurrency Bugs through Abstraction and Projection," in *Proceedings - International Computer Software and Applications Conference*, S. Reisman, S. I. Ahamed, L. Liu, D. Milojicic, W. Claycomb, M. Matskin, H. Sato, M. Nakamura, S. Cimato, C. H. Lung, Z. Zhang, and Z. Zhang, Eds., IEEE Computer Society, 2016, pp. 600–601. doi: 10.1109/COMPSAC.2016.102.