Cogent Engineering

ISSN: 2331-1916 (Online) Journal homepage: www.tandfonline.com/journals/oaen20

The acceptance, use, and perceptions of metamorphic testing for a sample of open-source software developers

Brittany R. Hoard

To cite this article: Brittany R. Hoard (2025) The acceptance, use, and perceptions of metamorphic testing for a sample of open-source software developers, Cogent Engineering, 12:1, 2522652, DOI: 10.1080/23311916.2025.2522652

To link to this article: https://doi.org/10.1080/23311916.2025.2522652

COGENT ENGINEERING
2025, VOL. 12, NO. 1, 2522652
https://doi.org/10.1080/23311916.2025.2522652

COMPUTER SCIENCE | RESEARCH ARTICLE

OPEN ACCESS Check for updates

The acceptance, use, and perceptions of metamorphic testing for a sample of open-source software developers

Brittany R. Hoard

School of Technology, Northcentral University, San Diego, CA, USA

ABSTRACT
This article examines the use, acceptance, and perceptions of metamorphic testing (MT) for a sample of open-source software developers. The study findings, including descriptive data and the results of correlational analyses, are used to inform recommendations for practice and for future research. The guiding framework of this study is the unified theory of acceptance and use of technology (UTAUT). An online survey instrument was used to collect data from a sample of contributors to a variety of projects in a GitHub collection. The survey included quantitative questions designed to measure the following UTAUT constructs and variables as they pertain to metamorphic testing: performance expectancy (PE), effort expectancy (EE), behavioral intention (BI), and frequency of use (FoU). The PE and EE constructs were found to have moderate to strong positive correlations with the BI and FoU. The creation of interventions to educate developers about metamorphic testing is recommended. Future research could involve examining other factors that may affect the acceptance and use of metamorphic testing. In addition, tools that make MT easier to use, easier to learn, and more efficient could be developed. Tools that help the developer more easily identify metamorphic relations could be especially useful.

ARTICLE HISTORY
Received 29 December 2023
Revised 6 December 2024
Accepted 16 March 2025

KEYWORDS
Software testing; software verification; software quality; unified theory of acceptance and use of technology; metamorphic testing

SUBJECTS
Psychological Science; Work & Organizational Psychology; Computer Science (General); Software Engineering & Systems Development

1. Introduction

Software testing is an essential component of the software development life cycle. There are various challenges associated with testing software. One such challenge is the oracle problem, which is the difficulty inherent in testing software output for correctness due to the lack of a test oracle (Barr et al., 2015; Chen et al., 2018). Although the oracle problem does not apply to all scientific software, this problem is commonly encountered when testing scientific software due to the fact that many types of scientific software perform complex calculations or create complex models or simulations, where the correct output is often not evident. Another significant challenge for software testing is the possibility that defects remain in a software program regardless of the testing that has been performed on said program (Lidbury et al., 2015; Pourreza Shahri et al., 2019; Rao et al., 2013). These problems are significant because defects in software can have severe consequences for software users and researchers, including poor software performance, reduced precision or accuracy of software output, and retractions of research publications for which defective software was used (Kanewala & Chen, 2019). One significant defect in software used for scientific research led to the retraction of five research articles, one of which was highly cited by other researchers (Miller, 2006). Therefore, the development of software testing methods that can effectively address these challenges is important. One promising approach to this problem is metamorphic testing (MT) (Chen et al., 1998, 2003).

CONTACT Brittany R. Hoard B.Hoard0115@o365.ncu.edu, hoardbrittany@gmail.com School of Technology, Northcentral University, 9388 Lightwave Avenue, San Diego, 92123 CA, USA
†Present address: School of Technology, Western Governors University, 4001 South 700 East Suite 700, Salt Lake City, UT, USA

Supplemental data for this article can be accessed online at https://doi.org/10.1080/23311916.2025.2522652.

Recent literature has discussed researchers' experience with using MT for software testing (Ding et al., 2016; Kanewala & Chen, 2019; Lin et al., 2018). The researchers' findings suggest that MT can be highly effective at detecting software defects even when conventional testing methods have failed. However, the acceptance, use, and perceptions of MT among practitioners have not been studied. A better understanding of the acceptance and use of MT among software developers would enable industry and organizational leaders to take steps to increase the adoption of MT. Greater use of MT could reduce the number of undetected defects in finished software products.

1.1. Related work

This section discusses the related work that guided the study design and helped to shape the recommendations provided in Section 5.

1.1.1. Open-source software development and testing

Open-source software (OSS) communities tend to contain two types of developers: core developers and non-core developers. Core developers guide the development, management, and maintenance of an OSS project, are involved with the project over a more extended period than non-core developers, and are more active in contributing to the project than are non-core developers (Yamashita et al., 2015). Researchers have found that a small proportion of the contributors to an OSS project make most of its code contributions (Avelino et al., 2016). A study that had a sample size of thousands of projects employed multiple heuristics for determining which contributors were core developers (Yamashita et al., 2015). The study findings showed that most OSS projects have fewer than 16 core developers and that about 20% of the contributions of both core and non-core developers are to correct software defects (Yamashita et al., 2015).

There are differences between the quality assurance (QA) practices of OSS projects and closed-source software projects. A survey performed by Bahamdain (2015) found that OSS projects tend to employ informal and unstructured QA, risk assessment, and testing practices. Additionally, they observed that there is little planning for the development life cycle of the project. The discovery of software defects happens late in the project's development, and empirical data regarding the software quality is often not gathered. In a large-scale study of over 20,000 OSS projects, Kochhar et al. (2013) found that 38% of projects did not contain any test cases, that 85% of projects had fewer than 100 test cases, and that projects with test cases tended to be larger and have more contributors than those without test cases. They also noted that a high number of test cases does not guarantee that a software product will be free of defects.

1.1.2. Metamorphic testing

The MT method is a software testing method that involves checking for the satisfaction of metamorphic relations (MRs) among a set of software tests. These relations specify how a software program's output should change when the program's input changes in a specific way. The concept of MT was first introduced in a technical report (Chen et al., 1998). The current state of MT research has been surveyed in two recent articles (Chen et al., 2018; Segura et al., 2016). The key finding from these studies is that MT has significant advantages, including conceptual simplicity, straightforward implementation, and cost-effectiveness. Researchers have explained, using a testing theory framework, how programs without test oracles can be tested using MT (Kanewala & Chen, 2019). They noted that MT tests tend to be easy to implement once MRs are selected.

There are several challenges associated with the use of MT, including a lack of MT tools, difficulties with systematically identifying and selecting useful MRs, problems with generating effective test cases, and the lack of a standard method for describing MRs. These challenges could affect a developer's decision of whether or not to use MT for their software project(s). It should be noted that there are examples in which some of these challenges have been overcome for specific problems or scenarios. Some of these cases will be discussed throughout this section.

Researchers and practitioners may find MRs difficult to understand and use due to the lack of a commonly accepted method for describing them (Segura et al., 2017). There is high variability in the way

MRs are described, leading to miscommunications. While there have been proposed methods for formalizing MR descriptions, none have been widely adopted, which may be due to some stakeholders finding them difficult to comprehend (Hui & Huang, 2013). Furthermore, in a survey on MT research, it was found that important information about the MRs was often left out of publications (Segura et al., 2016).

The MT method can be challenging to use in practice because of a lack of MT tools. In a survey of 119 MT research papers, only two of them proposed a tool as a primary contribution to the literature (Segura et al., 2016). One example of a testing tool that incorporates MT is JFuzz, a tool for automated unit testing of Java classes (Zhu, 2015). Another MT tool is MTKeras, an MT framework specifically for machine learning applications (Liu et al., 2020). The applicability of these tools is limited to specific problem domains or types of software. The lack of publicly available and well-maintained MT tools has hindered the widespread adoption of MT (Segura et al., 2016).

The selection of appropriate MRs is a non-trivial process. Developing useful MRs requires knowledge of the problem domain, the properties of useful MRs, and the processes needed to construct MRs. Using all relevant MRs in testing can be inefficient, so prioritizing MRs based on their effectiveness is an important task. The automated generation of MRs is one potential solution to this challenge, but approaches for MR generation have been largely limited to numerical programs. The production of useful source test cases for MT use is essential since the test cases can affect the effectiveness of the MRs (Chen et al., 2004, 2018; Wu et al., 2005).

Researchers have addressed these challenges by proposing approaches for selecting MRs and for generating test cases. One such approach is the incremental creation of a hierarchy of MRs (Lin et al., 2018). In this approach, the results of an MR test are employed to generate additional MRs to locate software defects. A machine learning-based approach to predict MRs at the function level automatically was proposed and tested on programs with numerical inputs and outputs (Kanewala, 2014). The Java class testing tool JFuzz combines MT with a test case generation method called data mutation (Zhu, 2015). The METRIC framework is a category-choice framework that measures differences among test cases and the association of every MR with a set of choices and categories (Chen et al., 2016). The METRIC framework was developed to identify a set of useful MRs that adequately cover the code. One limitation of METRIC is its reliance on tester expertise for MR identification.

The amount of literature relating to MT has increased over time, suggesting a sustained growth in researcher interest and acceptance in MT (Chen et al., 2018; Segura et al., 2016). A survey of MT literature (Segura et al., Segura et al. 2016) found that the cumulative number of MT publications from 1998 to 2015 fit well to a quadratic function with a determination coefficient of 0.997, indicating polynomial growth. Similarly, Chen et al. (2018) noted the growing interest in MT by researchers and the use of MT for detecting defects in real-world programs.

One example of a recent successful application of MT to a real-world problem is MET-MAPF, which is an MT approach to solving the multi-agent path-finding problem (Zhang et al., 2024). This problem is a challenging optimization problem that requires the successful computation of collision-free paths for multiple agents from the agents' current location to their destination. Experimental results demonstrate that MET-MAPF, which generates test cases for 10 MRs specific to MAPF systems, can detect different types of failures depending on which MRs are used (Zhang et al., 2024). Similarly, MetaCDP, an MT approach for the quality assurance of containerized data pipelines, has successfully utilized a set of 10 MRs that pertain to the correctness and robustness of the data pipeline in order to expose several problematic behaviors (Ur Rehman et al., 2024). Another example of a real-world application of MT is Metamorphic Security Testing for Web-interactions, which is an approach for Web system security testing that integrates MT with test input generation strategies based on mutational fuzzing (Chaleshtari et al., 2023). With this approach, the oracle problem, which is inherent in security testing, is addressed via the specification of MRs that capture security properties. These MRs are then transformed into Java code and combined with input data to perform automated security testing.

1.2. Theoretical framework

The study's theoretical framework is the unified theory of acceptance and use of technology (UTAUT) (Venkatesh et al., 2003). The UTAUT was selected as the theoretical framework for this study because it is a widely used model among researchers to predict technology use and acceptance (Al-Mamary et al., 2015). Furthermore, researchers have found that the quality of the UTAUT is high and that it is robust and valid (Venkatesh & Thong, 2016; Walldén et al., 2016). Thus, the variables and constructs included in this study were drawn from the UTAUT.

The UTAUT was initially developed to explain why individuals decide whether to accept and use software system innovations. It is a unifying model that integrates elements of eight previous models of information technology (IT) acceptance. In this framework, the constructs of performance expectancy (PE), effort expectancy (EE), and social influence (SI) determine the behavioral intention (BI) of the new technology.

The UTAUT constructs relevant to this study are the PE, EE, and BI. The PE is the user's belief that the new technology will help improve their job performance or result in other favored job-related outcomes (Venkatesh et al., 2003). The EE represents the user's perceived ease of use and understanding of the new technology. The BI is the user's intention to use the new technology. In the UTAUT, the BI is used as a measure of user acceptance of the technology. The technology of interest is metamorphic testing (MT) in the context of this study.

Since its development, the UTAUT has been employed in a wide variety of contexts, including understanding the readiness of local governments to adopt smart city technology, students' acceptance of mobile learning systems, and the willingness of potential customers to use delivery services (Almaiah et al., 2019; Gunawan, 2018; Xiang & Wu, 2018). It has been used for studying processes and practices for managing and developing software (Anderson, 2019; Guardado, 2012). The UTAUT helps leaders understand the factors underlying the acceptance of new technologies so that effective interventions targeted at users who are less likely to adopt new technologies can be designed (Venkatesh & Thong, 2016).

In this study, we examine whether the UTAUT is useful for understanding the acceptance, use, and perceptions of a software testing method. Before this study was performed, the UTAUT had not been applied to the adoption of software testing methods.

1.3. Study significance

This study makes the following novel contributions:

- Characterizes the use and acceptance of MT among OSS developers.
- Characterizes the perception of MT among OSS developers by employing the UTAUT constructs of performance expectancy (PE) and effort expectancy (EE).
- Extends the generalizability of UTAUT by applying its constructs to the study of a software testing method.
- Quantifies the relationships between the PE and EE constructs and the acceptance and use of MT.
- Makes recommendations for increasing MT adoption based on the study results and a review of the literature.

The study results suggest that MT adoption may depend on developers' perception of MT as being useful to their job performance and easy to learn, understand, and use. In addition, the responses for the individual survey items inform recommendations for specific actions that can be taken to improve MT acceptance and use. For example, a majority of participants responded that they neither agree nor disagree that using the MT method enables them to accomplish tasks more quickly, which may indicate a lack of knowledge among participants about the efficiency of MT. Thus, developing techniques for using MT efficiently could be useful for increasing MT acceptance and use. It should be noted that selecting appropriate metamorphic relations and other challenges involved with incorporating MT into existing software development workflows could be significant hurdles to the efficient use of MT. Therefore, techniques to address these issues may be effective at increasing the use and acceptance of MT.

1.4. Research questions

Our correlational analysis addresses the following research questions (Hoard, 2021):

1. To what extent is there a relationship between the acceptance of metamorphic testing (MT) among OSS developers and the following constructs: (a) performance expectancy and (b) effort expectancy?
2. To what extent is there a relationship between the frequency of use of MT among OSS developers and the following constructs: (a) performance expectancy and (b) effort expectancy?

1.4.1. Hypotheses

Null hypotheses:

1. There is no statistically significant relationship between the acceptance of MT among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.
2. There is no statistically significant relationship between the frequency of use of MT among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.

Alternative hypotheses:

1. There is a statistically significant relationship between the acceptance of MT among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.
2. There is a statistically significant relationship between the frequency of use of MT among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.

It is important to note that these research questions do not presume anything about the level of experience that the study participants have with MT or the amount of knowledge that they have about MT. More information about this aspect of our study design can be found in Section 2.5.

Although the existence of positive relationships between these variables may seem intuitive, the findings of previous studies in which the UTAUT framework was applied to the study of other software development practices and processes have been inconsistent (Anderson, 2019; Guardado, 2012).

This article is organized as follows. The study population, sample, research instrument, variable definitions, study procedures, and data analysis are described in Section 2. The study findings are presented in Section 3. A discussion of the study findings, including potential threats to validity, can be found in Section 4. Finally, Section 5 contains conclusions and recommendations for practice and research.

2. Materials and methods

In this study, we applied a quantitative methodology, which is suitable for validating existing theories, testing hypotheses, and measuring known constructs (Choy, 2014; Creswell, 2014). We used a correlational research design for this study because the research questions pertain to relationships among variables (Creswell, 2014).

A cross-sectional survey method was used to measure the theoretical constructs and variables of interest. A survey method was appropriate for this study because answering the research questions required measuring the study participants' opinions, perceptions, and intentions (Creswell, 2014). Due to its cost-effectiveness and high accessibility to the target population, an online survey instrument was used to collect the sample data.

2.1. Population and sample

GitHub is the most widely used website for hosting open-source software projects. The number of GitHub users is approximately 50 million, although not all GitHub user accounts are active (GitHub, Inc., 2020a). Since this population is quite large, sampling from across the entire GitHub user population was infeasible for this study. As a result, we decided to select our target population from the contributors to a collection of GitHub projects for which MT might be especially relevant.

2.2. Recruitment of study participants

We decided to recruit our study participants from the set of contributors to the OSS projects in a specific GitHub collection. The size of this target population was approximately 4500 contributors (Hoard, 2021). The GitHub collection that we chose for this study is the 'Software in science' collection on the GitHub website (GitHub, Inc., 2020b). As of September 24, 2020, this collection consisted of 18 repositories that host scientific software projects in addition to other types of software projects that could potentially be useful to scientists, including mathematical software. It should be noted that MT is likely to be useful for testing mathematical software in addition to scientific software, as the correct output of a mathematical software program may be difficult to know due to the complexity of the calculations involved. The scientific projects in this collection include the following: a multibody dynamics/physics library, a software package for a particle detector, particle-in-cell simulation software, event-driven software for rockets, a library for astronomy and astrophysics, an ensemble sampler for Markov chain Monte Carlo, a bioinformatics platform, and a set of tools for biological sequencing data analysis. The non-scientific software projects in this collection include the following: a computer algebra system, a package manager for supercomputers, a command shell for interactive computing, an API for scientific journals, a general-purpose mathematical software system, a system for computational discrete algebra, a computer algebra system, a library for arbitrary-precision interval arithmetic, a set of tools for manipulating high-throughput sequencing data and formats, and a web application for creating interactive notebooks that contain equations, code, and visualizations.

We selected this collection of GitHub projects for our study because of the variety of OSS projects included in the collection and because the collection includes scientific software, for which MT is likely to be useful, due to this category of software often (but not always) facing the oracle problem. (It should be noted that MT can be used to test software for which the oracle problem is not a known issue, and that we did not determine whether the oracle problem is a known issue for these GitHub projects.) Limiting the pool of study participants to those who had contributed to at least one of the OSS projects in this GitHub collection may have introduced biases in the study findings. It could be that the use or acceptance of MT is overrepresented in this study because of the inclusion of scientific software contributors in the study sample. It could also be the case that the use or acceptance of MT is underrepresented in this study; perhaps our chosen GitHub collection does not include some categories of open-source software or some types of scientific software for which MT is widely used or accepted.

The study participants were recruited by sending each of the potential participants a recruitment e-mail using the e-mail address that was either listed on the potential participant's GitHub user page or obtainable via the GitHub API. This e-mail contained information about the study and researcher, participant eligibility criteria, and a link to the survey. The participants were required to be at least 18 years of age and to speak English fluently to be eligible for inclusion in the study. It should be noted that by limiting the pool of study participants to only those with these particular characteristics, the study excluded GitHub contributors who were under 18 years of age and who did not speak fluent English, which may have introduced biases into the study findings, as it could have been the case that minors and non-English speakers would have had different experiences and perceptions of MT than adults and English speakers. When the potential participants followed the survey link in the recruitment e-mail, they saw a letter of informed consent that listed the eligibility criteria for the study and required the potential participant to agree to the study information in the letter before proceeding to the actual survey.

Based on a published study in which GitHub contributors were surveyed, the expected response rate for this survey was approximately 24% (Kalliamvakou et al., 2016). Given this expected response rate, the recruitment email was sent to 700 potential participants to ensure an adequate sample size. In order to ensure that study participants were recruited from all software projects in the selected GitHub collection, stratified sampling was used to select the 700 potential participants from whom the study sample would be drawn. As we discussed in Section 1.1.1, core developers guide the development, testing, and maintenance of an OSS project, and they are more active contributors than are the non-core developers (Yamashita et al., 2015). Thus, we selected the potential participants from the 100 most active contributors to each repository in the GitHub collection. This selection method was used in order to recruit the contributors who were the most likely to be the core developers of each OSS project.

From the initial group of 700 potential participants, a sample size of 41 participants was obtained for this study (Hoard, 2021). Power analysis for a two-tailed bivariate normal correlation test was performed using the G*Power 3.1.9.4 software program (Faul et al., 2007). In this power analysis, the statistical power was 0.8 and the significance level was 0.05. It was determined that the minimum correlation coefficient that the study could detect was 0.42, which is sufficient for this study since all of the correlation coefficients found are greater than 0.42. Because we employed a correlational design with a single sample, one group was included in the analysis.

It should be noted that a deliberate study design decision was made not to filter study participants without MT experience out of the sample. The reason for this decision is that a major objective of the study was to characterize the perceptions, use, and acceptance of MT among OSS developers in general, not just those developers who are already experienced in using MT. Meeting this study objective necessitated the inclusion of all OSS developers in the study sample regardless of their MT experience. Excluding developers who are not experienced with MT from the study sample would have resulted in the loss of valuable insight into the rate at which MT is actually being used among OSS developers, as well as information about the perceptions of MT by those who do not yet have experience using MT.

2.3. Instrumentation

The survey instrument used in this study is based on the instrument published and used by the developers of the UTAUT framework, with minor rewording to clarify that the questions pertain to the MT method (Venkatesh et al., 2003). The initial version of the instrument was designed for use in field studies to collect data from employees at six organizations in various industries in which participants were introduced to new technologies in their work environment (Venkatesh et al., 2003). The purpose of these field studies was to validate UTAUT and compare its performance to that of other technology acceptance models.

The UTAUT developers used partial least squares to verify the reliability and validity of the measures used in their survey instrument. They performed 48 validity tests across two studies and three time periods to assess the instrument's discriminant validity and convergent validity (Venkatesh et al., 2003). They found that all internal consistency reliability values were higher than 0.70 (Venkatesh et al., 2003). They determined that the loading pattern was acceptable, as most of the loadings were at least 0.70. Furthermore, they observed high intra-construct survey item correlations and low inter-construct item correlations.

Two of the constructs that were measured by the survey instrument used in this study were the UTAUT constructs of performance expectancy (PE) and effort expectancy (EE). The PE construct was selected for this study because Venkatesh et al. (2003) found that it was the strongest predictor of the BI. The EE construct was selected because it measures the ease of use and ease of learning the technology; as there are challenges associated with the use and learning of MT, we decided that the EE was a construct of interest for this study. The UTAUT construct of behavioral intention (BI), which is a measure of technology acceptance, and the frequency of use (FoU) variable, which is a measure of technology use, were also measured. Lastly, the instrument measured each participant's level of experience with the MT method. This experience variable is not included in the correlational analyses presented in this article but is included in the descriptive analyses to provide additional information about the study sample. The instrument measures all constructs and variables using numerical Likert scales. Section 2.4 outlines each construct and variable that is measured by the instrument, including the survey items used to measure each one and the possible values for each item.

2.4. Operational definitions of variables

The theoretical constructs and variables outlined in this section were measured in this study.

2.4.1. Performance expectancy (PE)

The PE is an independent variable. It is a UTAUT construct that is defined as the extent to which the user believes that the new technology will help improve their job performance or result in other favored job-related outcomes (Venkatesh et al., 2003).

The PE construct was measured by summing the Likert scores of the four survey items adapted from the instrument published in Venkatesh et al. (2003). These items are as follows (Hoard, 2021):

- I would find the method useful in my job.
- Using the method enables me to accomplish tasks more quickly.
- Using the method increases my productivity.
- If I use the method, I will increase my chances of getting a raise.

Like the instrument used in Venkatesh et al. (2003), each item was measured using an ordinal scale, namely a seven-point Likert scale with possible values being integers in the range 1-7. The total possible score for the PE construct was an integer in the range 4-28 . The following scores are possible for each item:

- 1 – Strongly disagree
- 2 – Disagree
- 3 – Somewhat disagree
- 4 – Neither agree nor disagree
- 5 – Somewhat agree
- 6 – Agree
- 7 – Strongly agree

2.4.2. Effort expectancy (EE)

The EE is an independent variable. It is a UTAUT construct that represents the user's perceived ease of use of new technology and the ease of learning and understanding it (Venkatesh et al., 2003).

The EE construct was measured by summing the Likert scores of the four survey items adapted from the instrument published in Venkatesh et al. (2003). These items are as follows (Hoard, 2021):

- The method is clear and understandable.
- It would be easy for me to become skillful at using the method.
- I would find the method easy to use.
- Learning to use the method is easy for me.

Like the instrument used in Venkatesh et al. (2003), each item was measured using an ordinal scale, namely a seven-point Likert scale with possible values being integers in the range 1-7. The total possible score for the EE construct was an integer in the range 4-28 . The following scores are possible for each item:

- 1 – Strongly disagree
- 2 – Disagree
- 3 – Somewhat disagree
- 4 – Neither agree nor disagree
- 5 – Somewhat agree
- 6 – Agree
- 7 – Strongly agree

2.4.3. Behavioral intention (BI)

The BI is a dependent variable. It is a UTAUT construct defined as the user's intention to use the new technology within a certain number of months (Venkatesh et al., 2003). The BI is a measure of user acceptance of new technology.

The BI construct was measured by summing the Likert scores of three survey items adapted from the instrument published in Venkatesh et al. (2003). These items are as follows (Hoard, 2021):

- I intend to use the method in the next 12 months.
- I predict I would use the method in the next 12 months.
- I plan to use the method in the next 12 months.

Like the instrument used in Venkatesh et al. (2003), each item was measured using an ordinal scale, namely a seven-point Likert scale with possible values being integers in the range 1-7. The total possible score for the BI construct was an integer in the range 3-21 . The following scores are possible for each item:

- 1 – Strongly disagree
- 2 – Disagree
- 3 – Somewhat disagree
- 4 – Neither agree nor disagree
- 5 – Somewhat agree
- 6 – Agree
- 7 – Strongly agree

2.4.4. Frequency of use (FoU)

The FoU is a dependent variable that represents how often the user uses the new technology. The original UTAUT developers measured technology usage via system logs rather than including a relevant item in their research questionnaire (Venkatesh et al., 2003). Since the use of system logs was irrelevant to this study, the survey instrument included a question that asked the participant about their usage frequency, as in the survey used in Venkatesh et al. (2012).

The FoU was measured using a single survey item with a seven-point Likert scale. The item is (Hoard, 2021):

- Please choose your usage frequency for the metamorphic testing method.

Like the instrument used by Venkatesh et al. (2012), the item was measured using an ordinal scale, namely a seven-point Likert scale with possible values being integers in the range 1-7. The following scores are possible for this item:

- 1 – Never
- 2 – Rarely, in less than 10% of the chances when I could have
- 3 – Occasionally, in about 30% of the chances when I could have
- 4 – Sometimes, in about 50% of the chances when I could have
- 5 – Frequently, in about 70% of the chances when I could have
- 6 – Usually, in about 90% of the chances when I could have
- 7 – Every time

2.4.5. Experience

The participant's level of experience with MT was collected to gain additional insight about the study sample. This variable is not included in the correlational analyses presented in this article but is included in the descriptive analyses to provide additional information about the study sample.

Experience was measured using a single survey item with a five-point Likert scale. The item is (Hoard, 2021):

- Please choose your level of experience with the metamorphic testing method.

Like the instrument used by Kim et al. (2005), this item was measured using an ordinal scale, namely a five-point Likert scale with possible values being integers in the range 1-5. The following scores are possible for this item:

- 1 – Less than 6 months
- 2 – 6 months to less than 12 months
- 3 – 1 to 3 years
- 4 – 4 to 6 years
- 5 – 7 years or more

2.5. Study procedures

First, we created a list of potential study participants by collecting e-mail addresses from the 100 most active contributors to each repository in the GitHub 'Software in science' collection (Hoard, 2021). We decided to limit the set of potential participants to the most active contributors to each project to maximize the number of core contributors included in the sample. The literature suggests that core contributors tend to be more active contributors and more involved in guiding the development and maintenance of OSS projects than non-core contributors (Yamashita et al., 2015). Therefore, core contributors are more likely to make decisions regarding software testing. The actual number of contributors recruited from each project depended on the number of contributors whose e-mail addresses were either accessible on their GitHub user Web page or via the GitHub API.

The list of repositories in the GitHub 'Software in science' collection, the number of potential participants recruited from among the contributors to each repository, and a Yes/No value that specifies whether or not the repository contains scientific software are presented in Table 1. The other inclusion criteria, namely that the participants were at least 18 years of age and spoke English fluently, were listed in the recruitment e-mail that was sent to each potential participant. Also, the first page of the research instrument contained a letter of informed consent that listed the eligibility criteria for the study and required the potential participant to agree to the information in the letter before proceeding to the actual survey. In total, we selected 700 contributors for recruitment. We stored the list of e-mail addresses for the final set of potential participants in a spreadsheet file.

Next, we recruited the study participants by sending each of the potential participants a recruitment e-mail. This e-mail contained information about the study and researcher, participant eligibility criteria, participant activities and time commitments, instructions for participation, and a link to the survey. The e-mail contained a brief definition of MT to clarify what we mean by the term 'metamorphic testing', as it is possible that some participants use the MT method without knowing its name. We did not provide the participants with extensive background information about MT as the goal of the study was to measure participants' perceptions of MT without overly influencing these perceptions. The respondent was invited to click on the survey link if they were interested in participating and eligible to participate in the study. If a respondent clicked on the survey link, they were taken to a Web page that contains the Qualtrics survey. The first page of this survey provided the letter of informed consent and a multiple-choice question at the end of the letter that asked the respondent whether they agreed to participate in the study. The respondent had to select 'Yes, I agree' and click the Next button to access the survey. The survey included items to measure all relevant variables.

The data collection period was divided into two phases, each of which was two weeks long. We selected this time period based on research that found over 90% of survey responses occur within 2 weeks (Phillips et al., 2016). We sent recruitment e-mails to a set of 350 potential participants at the beginning of each phase. A reminder e-mail was automatically sent via Qualtrics to nonrespondents after

Table 1. Number of potential participants recruited from each repository (Hoard, 2021).

Repository	Potential participants	Scientific software
simbody/simbody	17	Yes
cms-sw/cmssw	77	Yes
ComputationalRadiationPhysics/picongpu	13	Yes
psas/av3-fc	3	Yes
astropy/astropy	88	Yes
dfm/emcee	21	Yes
cyverse/atmosphere	11	Yes
dib-lab/khmer	32	Yes
sympy/sympy	83	No
spack/spack	91	No
ipython/ipython	84	No
ropensci/rplos	4	No
sagemath/sage	62	No
gap-system/gap	19	No
Singular/Singular	22	No
fredrik-johansson/arb	11	No
broadinstitute/picard	46	No
markusschanta/awesome-jupyter	16	No

one week. Since the desired sample size was not reached by the end of the first two-week collection phase, a second set of 350 potential participants was recruited.

2.6. Data analysis

The data was collected using an online Qualtrics survey and imported into IBM SPSS Statistics Version 26 for analysis (Hoard, 2021). Spearman correlation analyses were performed to measure the relationships among variables. We began the analysis by creating scatterplots for the following variable pairs and visually inspecting them for monotonicity: BI versus PE, BI versus EE, FoU versus PE, and FoU versus EE (Laerd Statistics, 2021). All of the relationships between variable pairs were monotonic. Then, the Spearman rank-order correlation coefficient, the degrees of freedom, and the p-value were obtained (Corder & Foreman, 2014). This data was used to determine whether statistically significant relationships exist between the variable pairs.

3. Results

This section begins with a presentation of the descriptive data and the results of the statistical analyses. Then, the study results are evaluated in a brief discussion.

3.1. Findings

Forty-one of the 688 potential study participants who received the e-mail (12 of the recruitment e-mails bounced) completed the survey, resulting in a response rate of 6.0% (Hoard, 2021). The possibility of nonresponse bias was mitigated by performing a wave analysis to estimate the nonresponse bias, which is discussed in Section 4.6.1.

3.1.1. Descriptive statistics

The descriptive statistics for the Experience variable, which operationalizes the participants' level of experience with MT, are presented in Table 2. Most participants (58.54%) indicated they have less than six months of experience with MT, with only 14.63% of participants indicating seven or more years of experience with MT. It should be noted that this experience data is not included in our correlational analyses, but was collected solely to provide additional insight into our study sample. The usage of MT by the study participants is captured by the FoU variable, which is discussed later in this section.

Descriptive statistics for the PE, EE, BI, and FoU are presented in Table 3. The scores for the PE, EE, and BI constructs were obtained by summing the Likert scores of the survey items that correspond to that construct. Given that the range of possible scores for the PE and EE is 4-28 , the mean and median values of the PE and EE are near the middle of the range of possible scores. The range of possible scores for the BI is 3-21 , meaning that the center of the BI data is near the middle of the range of possible scores. Since the range of possible scores for the FoU is 1-7, the mean and median of the FoU data are at the lower end of the range.

3.1.2. Survey data

The percentages of respondents who selected each answer for each survey item are provided in Appendix A in the supplementary material and are presented graphically in Figures 1–4. In this section, we present some noteworthy findings from this data, which are used to inform the recommendations provided in Section 5.

Four survey items correspond to the PE construct. The percentages of responses for each PE survey item are shown in Figure 1. For the survey item, 'I would find the method useful in my job', most responses are positive or neutral; they are relatively evenly spread across the four scores of 'Neither agree nor disagree' (21.95%), 'Somewhat agree' (19.51%), 'Agree' (24.39%), and 'Strongly agree' (19.51%). The item 'Neither agree nor disagree' could indicate that the participant does not know whether they would find the MT method useful in their job. This score ('Neither agree nor disagree') has the most responses for the two survey items, 'Using the method enables me to accomplish tasks more quickly' (51.22%) and 'Using the method increases my productivity' (46.34%). For the last item, 'If I use the method, I will increase my chances of getting a raise', most responses belong to the two scores 'Strongly disagree' (36.59%) and 'Neither agree nor disagree' (36.59%).

Four survey items correspond to the EE construct. The percentages of responses for each EE survey item are shown in Figure 2. For the survey item 'The method is clear and understandable', most responses are positive or neutral; they are mostly spread across the three scores of 'Neither agree nor disagree' (24.39%), 'Somewhat agree' (19.51%), and 'Agree' (24.39%) (Hoard, 2021). For the survey item 'It would be easy for me to become skillful at using the method', most responses belong to 'Neither agree nor disagree' (31.71%) and 'Agree' (31.71%). For the item 'I would find the method easy to use', the highest percentage of respondents answered 'Neither agree nor disagree' (31.71%), followed by 'Somewhat agree' (24.39%) and 'Agree' (24.39%). For the last item, 'Learning to use the method is easy for me', most responses belong to the two scores of 'Agree' (34.15%) and 'Neither agree nor disagree' (29.27%). We observe that a significant number of participants indicated they 'Neither agree nor disagree' with the survey item, suggesting a lack of knowledge about the EE construct as it pertains to MT.

Three survey items correspond to the BI construct. The percentages of responses for each BI survey item are shown in Figure 3. For the survey item 'I intend to use the method in the next 12 months', there are a significant number of negative responses mixed with positive ones; most responses have the three scores of 'Disagree' (21.95%), 'Agree' (21.95%), and 'Strongly disagree' (19.51%) (Hoard, 2021). Similarly, for the item 'I predict I would use the method in the next 12 months' most responses belong to the three scores 'Agree' (26.83%), 'Disagree' (24.39%), and 'Strongly disagree' (19.51%). For the item 'I

Table 2. Experience of study participants (Hoard, 2021).

Classification	Frequency ($N = 41$)	%
Less than 6 months	24	58.54
1–3 years	6	14.63
4–6 years	4	9.76
7 years or more	6	14.63
No response	1	2.44

Table 3. Descriptive statistics for the variables of interest [37].

Variable	Range of possible scores	Mean	Median	Standard deviation	Standard error
PE	[4–28]	16.54	16.00	5.13	0.80
EE	[4–28]	18.83	20.00	5.16	0.81
BI	[3–21]	11.32	12.00	6.47	1.01
FoU	[1–7]	2.44	2.00	1.75	0.27

plan to use the method in the next 12 months', most responses are evenly divided among the three scores of 'Disagree' (24.39%), 'Strongly disagree' (21.95%), and 'Agree' (21.95%).

The FoU construct corresponds to a single survey item. The percentages of responses for the FoU survey item are shown in Figure 4. The score with the most responses is 'Never' (48.78%) (Hoard, 2021). Most of the remaining responses are distributed across the four scores of 'Sometimes' (14.63%), 'Rarely' (12.20%), 'Occasionally' (9.76%), and 'Usually' (9.76%). No participant responded with 'Every Time'.

3.1.3. Relationships among constructs

The acceptance of MT was operationalized in this study by the BI. We performed Spearman's rank-order correlation test to assess the relationship between the BI and PE constructs. There is a statistically significant, moderate positive correlation between the BI and the PE, r_s (39) $= 0.636$, $p < 0.001$ (Hoard, 2021). A scatterplot of the BI versus the PE is shown in Figure 5.

We carried out another Spearman's rank-order correlation test to assess the relationship between the BI and the EE constructs. Similarly to the relationship between the BI and the PE constructs, there is a statistically significant, moderate positive correlation between the BI and the EE, r_s (39) $= 0.635$, $p < 0.001$ (Hoard, 2021). A scatterplot of the BI versus the EE is shown in Figure 6.

We performed a Spearman's rank-order correlation to assess the relationship between the FoU and the PE constructs. As with the relationship between the BI and the PE constructs, there is a statistically significant, moderate positive correlation between the FoU and the PE, r_s (39) $= 0.642$, $p < 0.001$ (Hoard, 2021). A scatterplot of the FoU versus the PE is shown in Figure 7.

We performed another Spearman's rank-order correlation to assess the relationship between the FoU and the EE constructs. There is a statistically significant, strong positive correlation between the FoU and the EE, r_s (39) $= 0.701$, $p < 0.001$ (Hoard, 2021). A scatterplot of the FoU versus the EE is shown in Figure 8.

3.2. Evaluation of the statistical analysis results

The first research question pertains to the relationship between the acceptance of MT among open-source software developers and the UTAUT constructs of the PE and EE. In this study, the acceptance of MT is represented by the UTAUT construct of the BI. We found a moderate positive correlation between the BI and the PE. Also, we found a moderate positive correlation between the BI and the EE. This finding is consistent with the UTAUT (Venkatesh et al., 2003).

The second research question asks about the relationship between the FoU of MT among open-source software developers and the UTAUT constructs of PE and EE. We found a moderate positive correlation between the FoU and the PE and a strong positive correlation between the FoU and the EE. These findings are consistent with the UTAUT (Venkatesh et al., 2003).

4. Discussion

In this section, we discuss the major study findings and explain how they contribute to the guiding theoretical framework and existing literature.

4.1. Research question 1

The study findings suggest that there is a statistically significant moderate positive relationship between the acceptance of MT among OSS developers and the PE and EE constructs. The study findings suggest that increasing the PE and EE, as they pertain to the MT method, among OSS developers may increase the developers' acceptance of the MT method. In other words, increasing the extent to which developers believe that MT will improve their job performance and be easy to learn and use could significantly increase MT's acceptance. This outcome could result in increased usage of the MT method, which could lead to fewer software defects. Since only correlations were examined in this study, we should also consider alternative interpretations of our findings. Developers who have accepted the MT method and

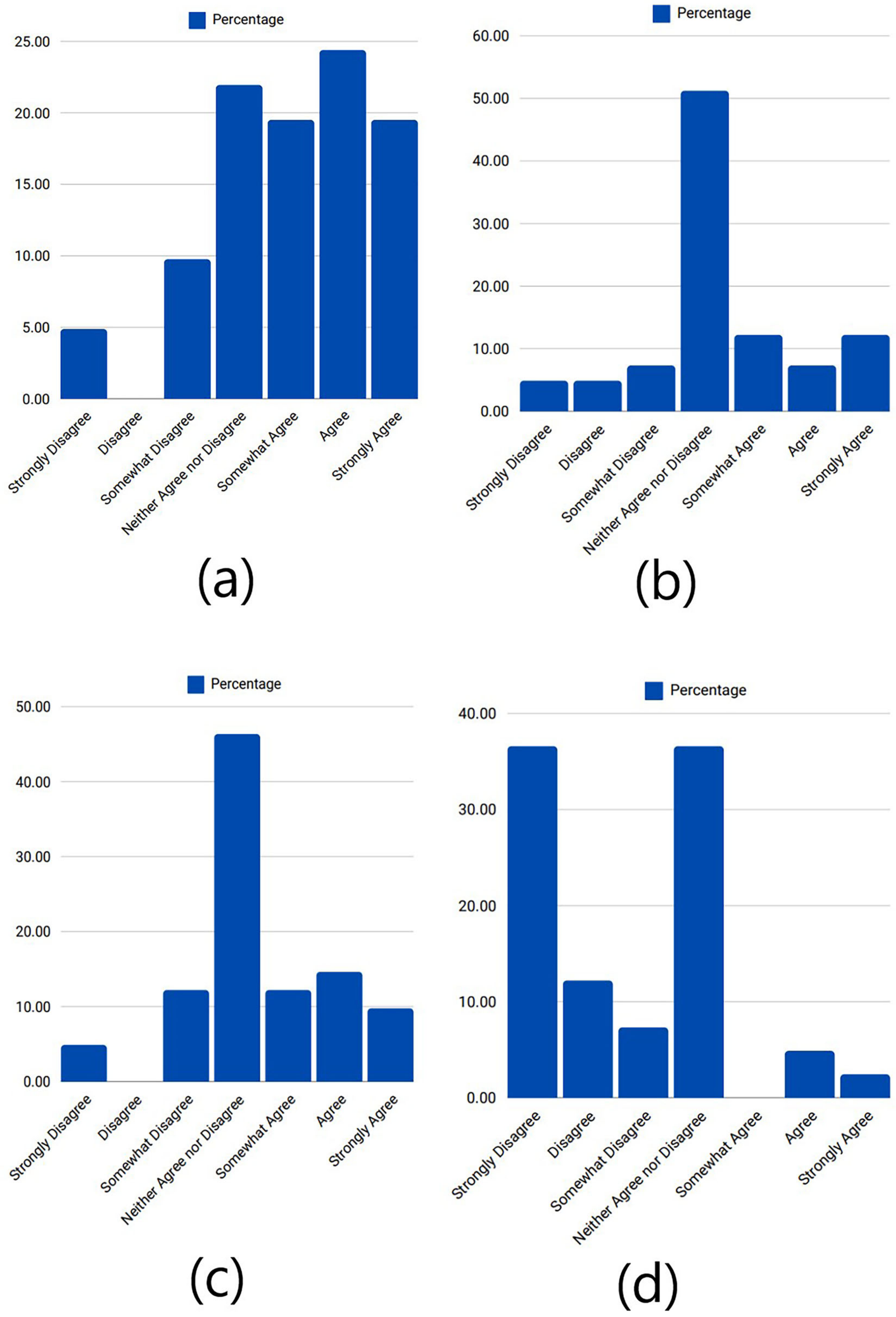

Figure 1. Percentages of Respondents for the Performance Expectancy (PE) Survey Items: (a) 'I would find the method useful in my job'. (b) 'Using the method enables me to accomplish tasks more quickly'. (c) 'Using the method increases my productivity'. (d) 'If I use the method, I will increase my chances of getting a raise'.

intend to use the method in the future may be more likely to perceive MT as useful and easy to use because they have already become familiar with the method.

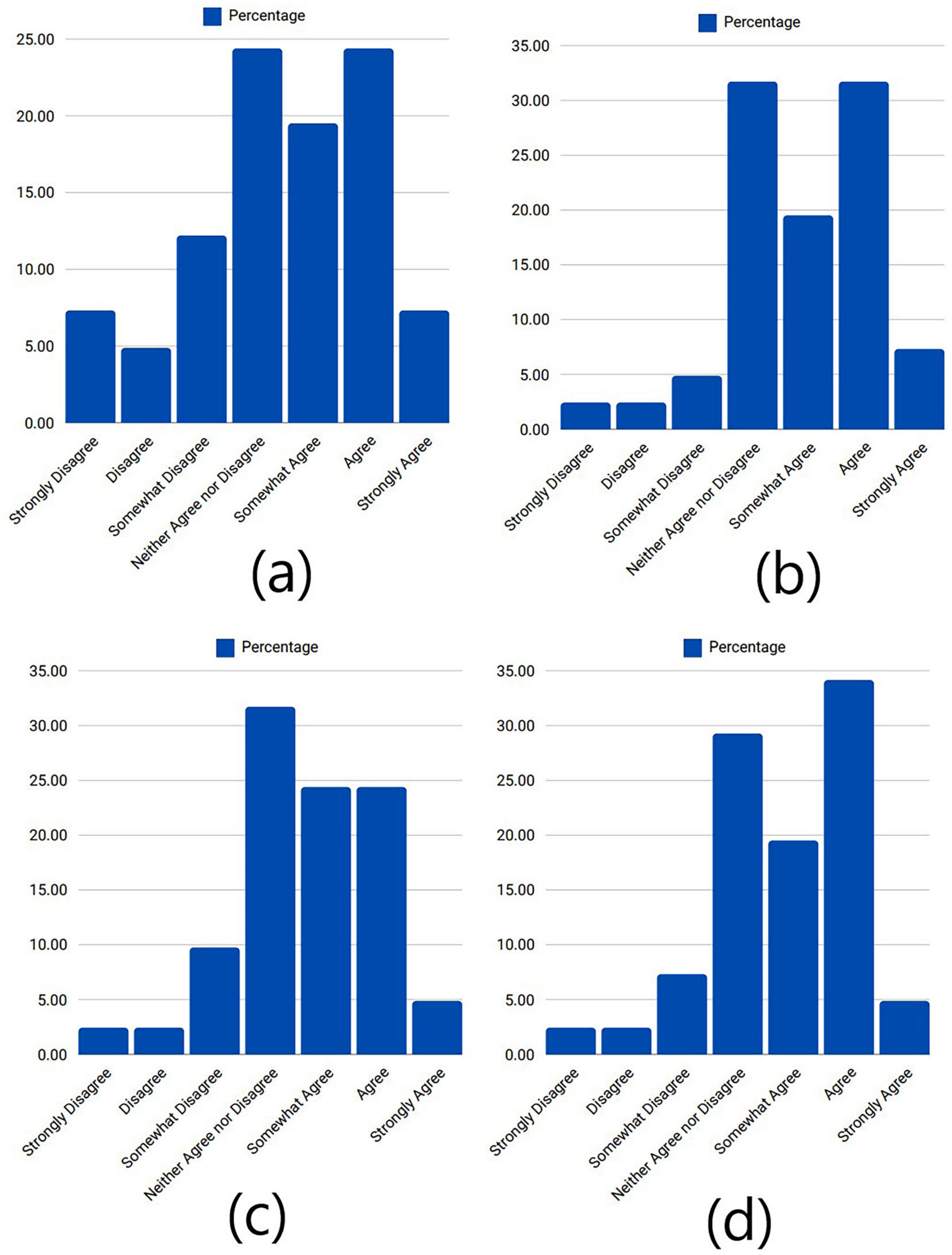

Figure 2. Percentages of Respondents for the Effort Expectancy (EE) Survey Items: (a) 'The method is clear and understandable'. (b) 'It would be easy for me to become skillful at using the method'. (c) 'I would find the method easy to use'. (d) 'Learning to use the method is easy for me'.

4.2. Research question 2

The study findings suggest that there is a statistically significant positive relationship between the FoU of the MT method among OSS developers and the PE and EE constructs. The study findings suggest that increasing the PE and EE, as they pertain to the MT method, among OSS developers may increase their usage of the MT method. Again, since only correlations were examined in this study, we should also consider alternative interpretations of our findings. Developers who use the MT method more frequently may be more likely to perceive MT as useful and easy to use because they already use the

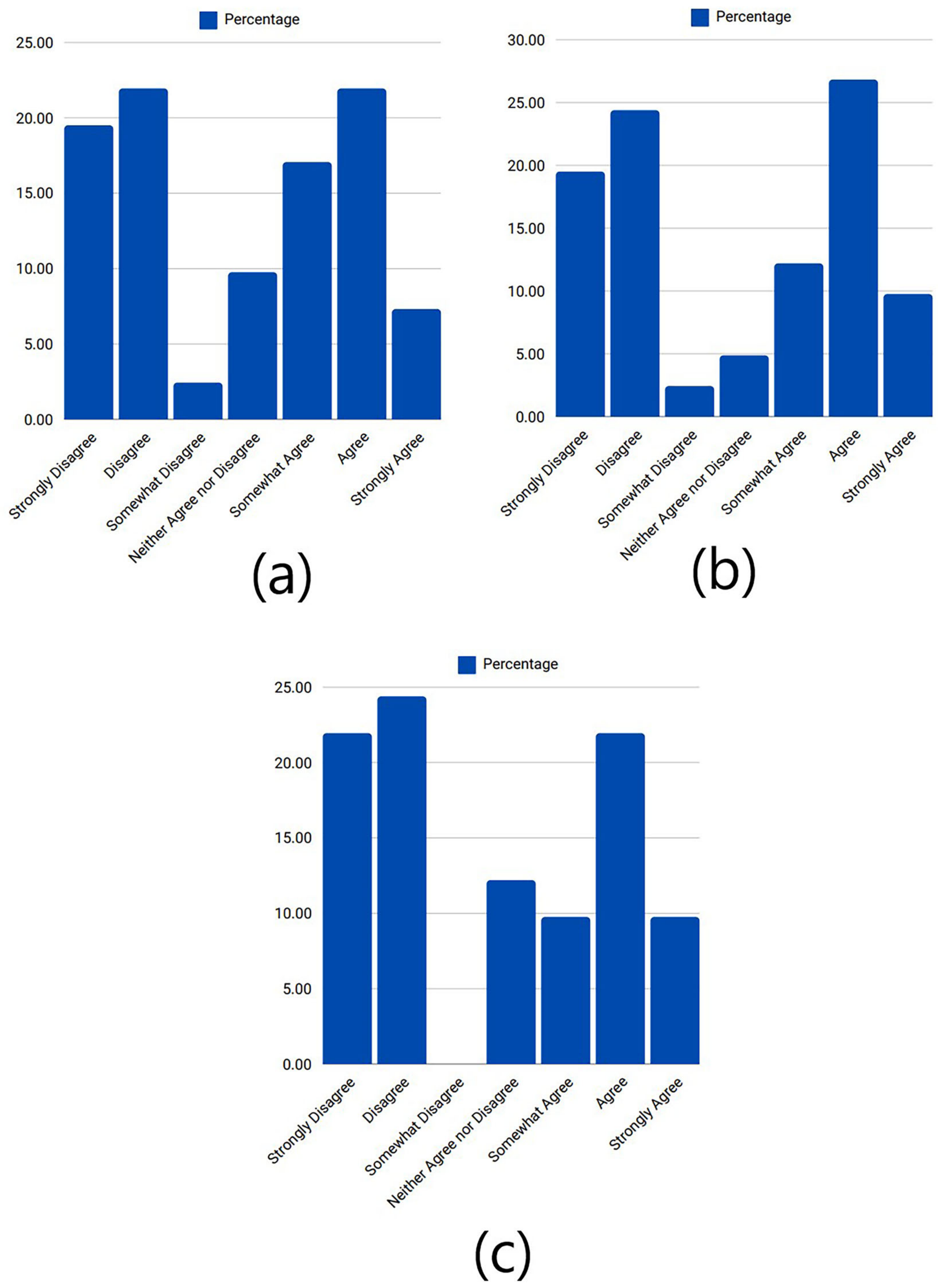

Figure 3. Percentages of Respondents for the Behavioral Intention (BI) Survey Items: (a) 'I intend to use the method in the next 12 months'. (b) 'I predict I would use the method in the next 12 months'. (c) 'I plan to use the method in the next 12 months'.

method regularly. In this scenario, since they have experience using MT, they already have some knowledge of how to use MT and have likely already benefited from using MT.

4.3. Comparison of findings with previous literature

The findings are consistent with the theoretical framework of UTAUT, which stipulates that the PE and EE are determinants of the BI (Venkatesh et al., 2003). Although a literature review did not uncover any

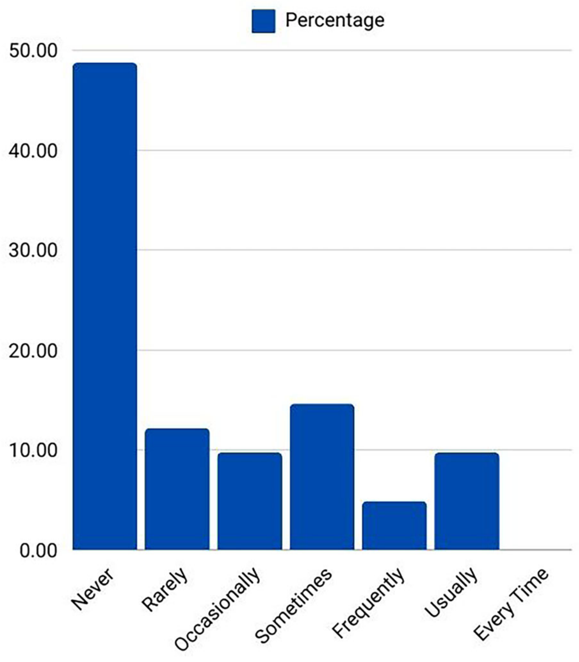

Figure 4. Percentages of respondents for the frequency of use (FoU) survey item.

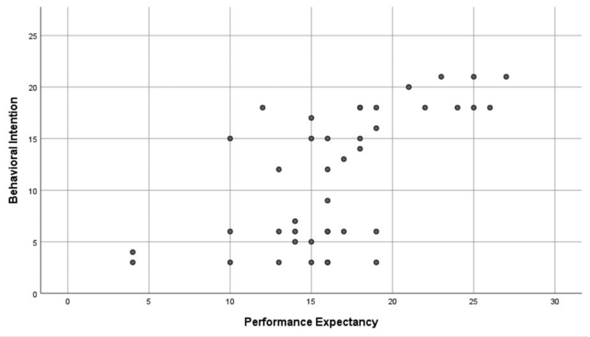

Figure 5. Scatterplot of Behavioral Intention versus Performance Expectancy (Hoard, 2021).

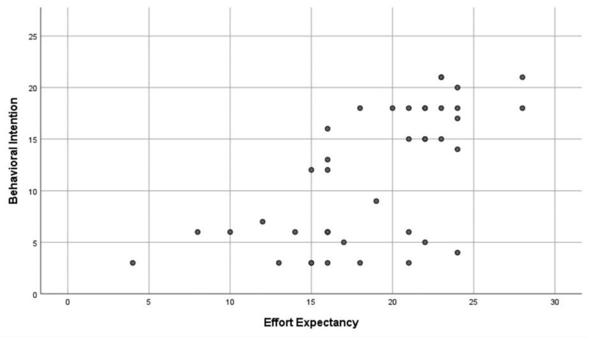

Figure 6. Scatterplot of Behavioral Intention versus Effort Expectancy (Hoard, 2021).

research in which UTAUT was applied to the study of software testing methods, the theory has been applied to other software development practices and processes (Anderson, 2019; Guardado, 2012). In an analysis of the adoption of continuous delivery among project managers, Anderson (2019) found a positive correlation between the PE and BI but did not find a significant relationship between the EE and BI.

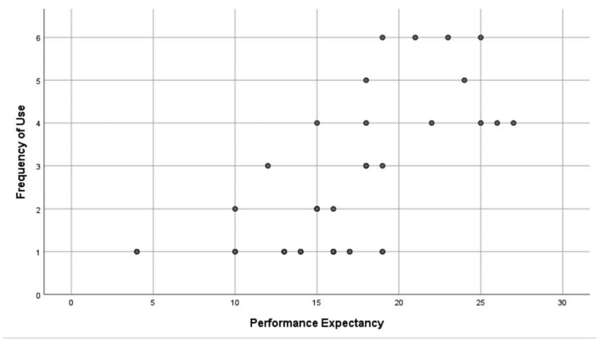

Figure 7. Scatterplot of frequency of use versus performance expectancy (Hoard, 2021).

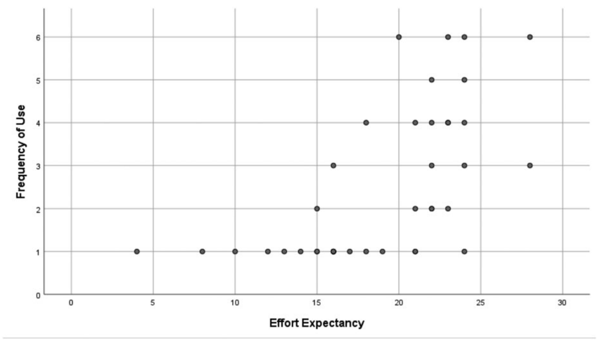

Figure 8. Scatterplot of frequency of use versus effort expectancy (Hoard, 2021).

On the other hand, a study of software development process acceptance found that the EE, but not the PE, was a significant determinant of the BI (Guardado, 2012). The inconsistency of these findings may be due to the differences in the studies' subject matter and in the composition of the study samples. Anderson (2019) examined the acceptance of a software development approach among a sample of software development project managers at various organizations. Most of them had at least three years of project management experience. In this context, the EE or ease of use of the approach may not have played a significant role in a typical project manager's decision to adopt the practice. Guardado (2012) studied software development process adoption among a sample of analysts, developers, managers, and technical experts who all work at the same company. In this context, the PE may not have been significantly related to the BI due to organizational culture or the study participants' roles..

4.4. Descriptive statistics

The descriptive statistics for the BI suggest that about 46% of participants do not intend to use MT within the next 12 months, and about 46% intend to use MT within the next 12 months. The descriptive statistics for the FoU indicate that about 49% of participants never use MT, 36% use MT but infrequently, and 15% use MT often. In summary, nearly half of the participants never use MT and do not intend to use it in the next year.

We observe that a significant number of participants (between 24.39% and 31.71%) indicated they 'Neither agree nor disagree' with each of the EE survey items, suggesting a lack of knowledge about the ease of learning and using the MT method. Similarly, the percentage of 'Neither agree nor disagree' responses (between 21.95% and 51.22%) to the PE survey items suggests that many participants do not know whether they will find MT useful or whether MT will allow them to be more productive or accomplish their tasks more quickly.

4.5. Theoretical framework

We found that the BI and the FoU of the MT method are moderately to strongly positively correlated with the UTAUT constructs of PE and EE. These findings support the applicability of a portion of the UTAUT theoretical framework, namely the PE and EE, to the study of software testing method adoption.

4.6. Limitations and threats to validity

One factor that could have influenced the study findings is the self-reported nature of the data. It is possible that some participants may not have answered all survey items honestly and accurately. Another factor to consider is that only contributors who were at least 18 years old and spoke English fluently were recruited for this study. Hence, caution should be taken when generalizing the study findings to OSS developers who do not have these characteristics. In addition, the small sample size (41 participants) limits the representativeness and generalizability of the study findings.

Another limitation of the study is that only contributors to the 18 OSS projects in the GitHub 'Software in science' collection were recruited. The projects in this collection are scientific software projects and other types of software that could be useful to scientists. MT is not necessarily useful to all types of OSS software. Therefore, if this study were expanded to include contributors to other OSS projects, it is quite possible that the percentage of participants who would find MT useful in their work would decrease, which would decrease the average PE score. We would expect that the average BI and FoU scores would also decrease because participants who do not find MT useful would also be less likely to use MT or intend to use it in the future. It is also possible that our chosen GitHub collection does not include some types of OSS projects for which MT is widely used or accepted. If our study sample included contributors to these particular projects, we would expect the average BI and FoU scores to increase because we would have more participants who use MT frequently and who intend to use MT.

As discussed in Section 2.2, we made the purposeful decision not to exclude participants with no MT experience or who reported that they never used MT from our study sample. If we had excluded such participants from our sample and only included data from participants who had used MT in our analysis, we expect there would have been a much higher average FoU score since we would only have had participants who reported that they used MT. Based on the moderate to strong correlations found in this study between the PE and the FoU and between the EE and the FoU, we would also expect that the average PE and EE scores would have been higher if we had excluded non-users of MT from our study.

The acceptance of MT was operationalized in this study by the UTAUT construct of BI, which only represents the participants' intention to use the new technology within a specific time period. In this study, the time period was 12 months. Hence, the study findings do not consider participants who only intend to use the MT method more than 12 months in the future. Finally, not all of the potential participants responded to the survey, which introduces the possibility of nonresponse bias. The steps taken to mitigate this concern are described in Section 4.6.1.

4.6.1. Nonresponse bias

In order to estimate the nonresponse bias, we performed a wave analysis. In wave analysis, which is a widely used method for assessing nonresponse bias, the study participants are divided into waves based on the time it took them to respond to the survey (Phillips et al., 2016). The final wave's responses are compared to the first wave's responses. The late respondents are used as proxies for nonrespondents, and the early respondents are treated as the true respondents. (The potential participants who received a study invitation but never took the survey are the actual nonrespondents.) The nonresponse bias is calculated by first obtaining the mean value of the variable of interest for each of the two waves of responses. Then, the proportion of actual nonrespondents is multiplied by the difference between these means.

In our study, the participants were divided into two waves: the first wave consisted of those who completed the survey before the reminder e-mail was sent. The second wave consisted of those who completed the survey after the reminder e-mail was sent. The results of our wave analysis are presented in Appendix B in the supplementary material. The findings suggest that nonrespondents would tend to score slightly less on the FoU, BI, PE, and EE than did the respondents (Hoard, 2021). This finding may be due to nonrespondents being less interested or less familiar with MT than respondents are, and hence being less likely to accept or use MT.

5. Conclusions

In this quantitative study, we examined relationships between the UTAUT constructs of PE and EE and the MT method's use and acceptance among OSS developers. The study results indicate significant moderate to strong positive relationships between the PE and EE constructs and MT use and acceptance. The study findings support the relevance of the UTAUT constructs and relationships to the adoption of software testing methods. Also, the study results suggest that increasing the extent to which developers believe that MT will improve their job performance and improving its ease of use may increase their acceptance and use of MT.

5.1. Recommendations for practice

The results of the study have important implications for software development practice. The study findings suggest that many developers have little experience with the MT method, with the majority of participants indicating they have less than six months of experience with MT, and almost half of the participants responding that they never use MT. This finding supports the need for education about MT on an introductory level, which includes describing the basic concept of MT, explaining what metamorphic relations (MRs) are, and outlining the advantages of MT for addressing the oracle problem and detecting software defects.

Examining the data for the survey items provides additional insight. Over half of the participants (51.22%) chose the option 'Neither agree nor disagree' for the item, 'Using the method enables me to accomplish tasks more quickly'. This finding could indicate that developers lack knowledge about the efficiency of MT. In order to address this knowledge gap, education about frameworks for efficient MT use, such as the Amsterdam implementation framework (Murphy et al., 2009), could be beneficial. In addition, education and experience regarding the efficient selection or generation of appropriate MRs are vital, since MR selection is not a trivial process.

In addition, the survey data suggests that a significant number of participants lack knowledge about the ease of learning and using the MT method. Therefore, increasing the subjects' hands-on experience with MT could be beneficial, such as training exercises in which the subject identifies MRs and performs MT on a set of sample programs. The subject could begin learning MT at the unit testing level by applying MT to a set of short and simple programs for which the MRs can be easily identified, like programs that calculate trigonometric functions. When the subject has gained some experience using MT, they could learn to use MT at the system/function level to test more complex, real-world programs. It should be noted that although MT is mostly used for testing at the system/function level, training developers on using MT at the unit testing level would be suitable for introductory training before progressing to more advanced training, which would focus on using MT at the system/function level.

The results of the correlation analysis suggest that practitioners should aim to increase the PE and EE among software developers in order to potentially improve the acceptance and usage of the MT method. Explaining how the MT method can improve job performance, specifically the detection of software defects, could increase the PE among developers. The MT method has been able to find defects in well-tested software programs that were not found previously using other testing methods. For example, over 100 defects were found in popular C compilers and widely used protein function prediction tools using MT (Lidbury et al., 2015; Pourreza Shahri et al., 2019). Furthermore, MT uncovered a new defect in the widely used Siemens test suite, which was the subject of testing research for 20 years (Rao et al., 2013). Research that demonstrates the effectiveness of MT for detecting software defects and handling the oracle problem could be used to support the claim that MT can enhance job performance (Cañizares et al., 2019; Ding et al., 2016; Kanewala & Bieman, 2014; Kanewala & Chen, 2019; Zhou & Sun, 2019).

Developers should be familiar with research about the appropriate selection of MRs. This action could increase the EE because choosing MRs is a critical step when learning to use MT effectively. Many MR selection approaches have been proposed by researchers (Ding et al., 2016; Kanewala, 2014; Lin et al., 2018; Zhou et al., 2016). In addition to specialized methodologies, researchers have developed frameworks to help identify useful MRs, such as the METRIC framework (Chen et al., 2016).

5.2. Recommendations for OSS developers

Previous research suggests that increasing the acceptance and use of MT among OSS developers could be useful for improving the quality of OSS projects. In a large scale study of over 20,000 OSS projects, Kochhar et al. (2013) found that 85% of OSS projects had fewer than 100 test cases. Kochhar et al. (2013) also noted that a high number of test cases does not guarantee that a software product will be free of defects. Since the use of the MT method has been able to find defects that were not found previously using other testing methods (Lidbury et al., 2015; Pourreza Shahri et al., 2019), a developer's decision to use MT to generate some of an OSS project's few test cases could reduce the number of defects in the OSS project.

Education and training on the MT method should be provided to OSS developers. Educators should incorporate the MT method into software testing books, curricula for software engineering and computer science degree programs, and online resources for OSS developers. Education on MT could increase the PE and EE for the MT method among OSS developers. Furthermore, organizations could increase MT acceptance and usage by training employees on the MT method. Since some OSS developers contribute to OSS projects as part of their employment, organizational training on MT could increase the PE and EE among OSS developers. Based on the study findings, increasing the EE and PE would be likely to result in the growth of MT acceptance and actual usage of MT among OSS developers.

5.3. Recommendations for future research

Researchers might build on this study by examining the factors associated with MT usage and acceptance using qualitative methodology. In such a research study, the participants would answer open-ended questions about their perception of the MT method and their reasons for using it or not using it. The questions could be guided by theoretical frameworks such as UTAUT but would allow participants to provide additional information about their responses. Using this methodology, researchers could learn about factors associated with MT adoption that are not included in the theoretical constructs of interest. Researchers could also examine whether specific contexts, such as the type or complexity of the software project being tested, are related to participants' decisions to use the MT method. Such research could also focus on participants' use of MT tools and whether the use of such tools is correlated with the extent to which participants view MT as being useful, effective, and easy to learn and use, as well as how frequently participants use MT. It may be the case that participants who use automated MT tools use MT more frequently than those who do not and may have a more positive opinion of MT. It could also be true that current MT tools are frustrating for participants to use, which could negatively affect participants' perception of MT and how often they use MT. Another suggestion is to design a research study that focuses on participants' understanding of MRs and how MRs are selected, as well as any barriers to this understanding, such as the lack of a common method of MR specification. Such a study could provide valuable insight into one of the major challenges associated with MT use and how it relates to participants' adoption of MT. Future research could also involve assessing the effectiveness of methods used to increase MT acceptance and use.

For this study, we only recruited participants who have contributed to OSS projects, limiting the study findings' generalizability to organizational contexts. Researchers could recruit software developers and testers from organizations to overcome this limitation. Also, the generalizability of the study could be improved by recruiting participants from a wider variety of OSS projects rather than just the GitHub 'Software in science' collection of projects, and by using a sample size that is significantly larger than the 41 participants used in this study.

Another logical step in this line of research is to examine the relationship between the other UTAUT constructs, namely social influence (SI) and facilitating conditions (FC), and MT adoption. The SI represents the effect of other people's opinions on the user's decision to use technology (Venkatesh et al., 2003). The FC is defined as the user's perception that the technical and organizational infrastructure supports the use of the technology. This research would provide additional evidence regarding the applicability of UTAUT to software testing methods. It would provide information about the relationship of these constructs to MT use and acceptance. Existing research supports the idea that software testers face challenges caused by human and organizational factors (Goncalves et al., 2017; Seth et al., 2014).

Future research could also involve studying the relationship between MT adoption and constructs from other theoretical frameworks. For example, the HMSAM constructs of curiosity, immersion, joy, and control could be included in future studies (Lowry & Gaskin, 2013). Although HMSAM was developed to study the adoption of hedonic-motivation systems (HMS) rather than utilitarian-motivation systems (UMS), researchers have recognized that hedonic motivation may play a role in UMS adoption (Venkatesh & Davis, 2000). Hence, research that focuses on the relationship between software testing method adoption and HMSAM constructs may be enlightening.

Finally, the study could guide further research and development with regard to MT tools and methods. The findings suggest that the development of tools, resources, techniques, and frameworks that make MT easier to use, easier to learn, and more efficient could spur developers to adopt MT. These resources should mitigate the significant challenges associated with using MT. One suggestion is the development of a consistent method for specifying MRs that is widely accepted by both software researchers and software practitioners. This could mitigate the challenges associated with the lack of a standard method for MR description (Segura et al., 2017). Because a method of specifying MRs should be commonly accepted to be useful, a collaborative effort between a large number of practitioners and researchers from various institutions and organizations may be the best approach to this effort. Another suggestion is the creation of additional MT tools to address the limitations of the few MT tools currently publicly available. Automated MT tools that can test software written in various programming languages would be beneficial for increasing the efficiency of MT testing. Tools that could automatically generate MRs for various problem domains and prioritize MRs by their usefulness would be especially helpful. Developers of MT tools might obtain guidance from approaches that already exist in the literature, such as the incremental approach to MR identification proposed in (Lin et al., 2018). The creation of tools that automate such approaches is one avenue of possible future research.

Disclosure of interest

No potential conflict of interest was reported by the authors.

Ethics statement

Ethical approval was obtained from the Institutional Review Board of Northcentral University before conducting this study.

Consent to participate

Written informed consent was obtained from all participants before they participated in the study.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

About the author

Brittany R. Hoard currently works as a Faculty Supervisor in the School of Technology at Western Governors University. Dr. Hoard obtained her PhD degree from Northcentral University, her MS degree from the University of New Mexico, and her BS degree from Penn State Erie. Her primary research interests are modeling/simulation and software testing.

ORCID

Brittany R. Hoard http://orcid.org/0000-0002-1897-1388

Data availability statement

The data that support the findings of this study are available from the corresponding author, BRH, upon reasonable request.

References

Al-Mamary, Y. H., Shamsuddin, A., & Aziati, N. (2015). Investigating the key factors influencing on management information systems adoption among telecommunication companies in Yemen: The conceptual framework development. *International Journal of Energy, Information and Communications*, *6*(1), 59–68. https://doi.org/10.14257/ijeic.2015.6.1.06

Anderson, A. (2019). Examination of adoption theory on the DevOps practice of continuous delivery (Doctoral dissertation). Walden University, *1*, 1–183.

Avelino, G., Passos, L., Hora, A., & Valente, M. T. (2016). *A novel approach for estimating Truck Factors* [Paper presentation]. 2016 IEEE 24th International Conference on Program Comprehension (ICPC), Austin, TX, USA (pp. 1–10). IEEE. https://doi.org/10.1109/ICPC.2016.7503718

Bahamdain, S. S. (2015). Open source software (OSS) quality assurance: A survey paper. *Procedia Computer Science*, *56*, 459–464. https://doi.org/10.1016/j.procs.2015.07.236

Cañizares, P. C., Núñez, A., & de Lara, J. (2019). An expert system for checking the correctness of memory systems using simulation and metamorphic testing. *Expert Systems with Applications*, *132*, 44–62. https://doi.org/10.1016/j.eswa.2019.04.070

Chaleshtari, N. B., Pastore, F., Goknil, A., & Briand, L. C. (2023). Metamorphic testing for Web system security. *IEEE Transactions on Software Engineering*, *49*(6), 1–43. https://doi.org/10.1109/TSE.2023.3256322

Chen, T. Y., Cheung, S. C., & Yiu, S. M. (1998). Metamorphic testing: A new approach for generating next test cases. Technical Report HKUST-CS98-01. Department of Computer Science, Hong Kong University of Science and Technology.

Chen, T. Y., Kuo, F.-C., Liu, H., Poon, P.-L., Towey, D., Tse, T. H., & Zhou, Z. Q. (2018). Metamorphic testing: A review of challenges and opportunities. *ACM Computing Surveys*, *51*(1), 1–27. https://doi.org/10.1145/3143561

Chen, T. Y., Poon, P.-L., & Xie, X. (2016). METRIC: METamorphic Relation Identification based on the Category-choice framework. *Journal of Systems and Software*, *116*, 177–190. https://doi.org/10.1016/j.jss.2015.07.037

Chen, T., Huang, D., Tse, T., & Zhou, Z. (2004). Case studies on the selection of useful relations in metamorphic testing. *Proceedings of the 4th Ibero-American Symposium on Software Engineering and Knowledge Engineering (JIISIC '04)* 569–583.

Chen, T., Tse, T., & Zhou, Z. (2003). Fault-based testing without the need of oracles. *Information and Software Technology*, *45*(1), 1–9. https://doi.org/10.1016/S0950-5849(02)00129-5

Choy, L. T. (2014). The strengths and weaknesses of research methodology: Comparison and complimentary between qualitative and quantitative approaches. *IOSR Journal of Humanities and Social Science*, *19*(4), 99–104. https://doi.org/10.9790/0837-194399104

Corder, G. W., & Foreman, D. I. (2014). *Nonparametric statistics: a step-by-step approach*. (2th ed.), John Wiley & Sons.

Creswell, J. W. (2014). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th ed.). SAGE Publications.

Ding, J., Zhang, D., & Hu, X.-H. (2016). An application of metamorphic testing for testing scientific software. in *Proceedings of the 1st International Workshop on Metamorphic Testing, MET '16*, Association for Computing Machinery, pp. 37−–43. https://doi.org/10.1145/2896971.2896981

E., T., Barr, M., Harman, P., McMinn, M., Shahbaz., & S., Yoo. (2015). The oracle problem in software testing: A survey. *IEEE Transactions on Software Engineering*, *41*(5), 507–525. https://doi.org/10.1109/TSE.2014.2372785

Faul, F., Erdfelder, E., Lang, A.-G., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods*, *39*(2), 175–191. https://doi.org/10.3758/BF03193146

GitHub, Inc. (2020a). Collection: Software in science. https://github.com/collections/software-in-science

GitHub, Inc. (2020b). URL Build software better, together. https://www.github.com

Goncalves, W. F., de Almeida, C. B., de Araujo, L. L., Ferraz, M. S., Xandu, R. B., & de Farias, I. (2017). *The influence of human factors on the software testing process: The impact of these factors on the software testing process* [Paper presentation].2017 12th Iberian Conference on Information Systems and Technologies (CISTI), Lisbon (pp. 1–6). IEEE. https://doi.org/10.23919/CISTI.2017.7975873

Guardado, D. (2012). Process acceptance and adoption by IT software project practitioners (Doctoral dissertation). Capella University, *1*, 1–177.

Gunawan, H. (2018). *Identifying factors affecting smart city adoption using the unified theory of acceptance and use of technology (UTAUT) Method* [Paper presentation]. 2018 International Conference on Orange Technologies (ICOT), Nusa Dua, BALI, Indonesia (pp. 1–4). IEEE. https://doi.org/10.1109/ICOT.2018.8705803

Hoard, B. (2021). Metamorphic testing among open-source software developers: A quantitative correlational study (Doctoral dissertation). Northcentral University, *1*, 1–139.

Hui, Z.-W., & Huang, S. (2013). *Achievements and challenges of metamorphic testing* [Paper presentation]. 2013 Fourth World Congress on Software Engineering (pp. 73–77). https://doi.org/10.1109/WCSE.2013.16

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2016). An in-depth study of the promises and perils of mining GitHub. *Empirical Software Engineering*, *21*(5), 2035–2071. https://doi.org/10.1007/s10664-015-9393-5

Kanewala, (2014). U. *Techniques for Automatic Detection of Metamorphic Relations* [Paper presentation].2014 IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops, OH, USA (pp. 237–238). IEEE. https://doi.org/10.1109/ICSTW.2014.62

Kanewala, U., & Bieman, J. M. (2014). Testing scientific software: A systematic literature review. *Information and Software Technology*, *56*(10), 1219–1232. https://doi.org/10.1016/j.infsof.2014.05.006

Kanewala, U., & Chen, T. Y. (2019). Metamorphic testing: A simple yet effective approach for testing scientific software. *Computing in Science & Engineering*, *21*(1), 66–72. https://doi.org/10.1109/MCSE.2018.2875368

Kim, S. S., Malhotra, N. K., & Narasimhan, S. (2005). Research note—Two competing perspectives on automatic use: A theoretical and empirical comparison. *Information Systems Research*, *16*(4), 418–432. https://doi.org/10.1287/isre.1050.0070

Kochhar, P. S., Bissyandé, T. F., Lo, D., & Jiang, L. (2013). *An empirical study of adoption of software testing in open source projects* [Paper presentation]. 2013 13th International Conference on Quality Software (pp. 103–112). https://doi.org/10.1109/QSIC.2013.57

Laerd Statistics. (2021). Spearman's Rank-Order Correlation - A guide to when to use it, what it does and what the assumptions are.

Lidbury, C., Lascu, A., Chong, N., & Donaldson, A. F. (2015). Many-core compiler fuzzing. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI '15* (pp. 65–76). Association for Computing Machinery. https://doi.org/10.1145/2737924.2737986

Lin, X., Simon, M., & Niu, N. (2018). Hierarchical metamorphic relations for testing scientific software. In *Proceedings of the International Workshop on Software Engineering for Science* (pp. 1–8). ACM. https://doi.org/10.1145/3194747.3194750

Liu, Y., Liu, Y., Chen, T. Y., & Zhou, Z. Q. (2020). *A testing tool for machine learning applications* [Paper presentation]. IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20, in, https://doi.org/10.1145/3387940.3392694

Lowry, P., & Gaskin, J. (2013). Taking "fun and games" seriously: Proposing the hedonic-motivation system adoption model (HMSAM). *Journal of the Association for Information Systems*, *14*(11), 617–671. https://doi.org/10.17705/1jais.00347

M., A., Almaiah, M. M., Alamri., & W., Al-Rahmi. (2019). Applying the UTAUT model to explain the students' acceptance of mobile learning system in higher education. *IEEE Access*, *7*, 174673–174686. https://doi.org/10.1109/ACCESS.2019.2957206

Miller, G. (2006). A scientist's nightmare: software problem leads to five retractions. *Science (New York, N.Y.)*, *314*(5807), 1856–1857. https://doi.org/10.1126/science.314.5807.1856

Murphy, C., Shen, K., & Kaiser, G. (2009). Automatic system testing of programs without test oracles. in *Proceedings of the Eighteenth International Symposium on Software Testing and Analysis – ISSTA '09* (p. 189). ACM Press. https://doi.org/10.1145/1572272.1572295

Phillips, A. W., Reddy, S., & Durning, S. J. (2016). Improving response rates and evaluating nonresponse bias in surveys: AMEE Guide No. *Medical Teacher*, *38*(3), 217–228. https://doi.org/10.3109/0142159X.2015.1105945

Pourreza Shahri, M., Srinivasan, M., Reynolds, G., Bimczok, D., Kahanda, I., & Kanewala, U. (2019). Metamorphic Testing for Quality Assurance of Protein Function Prediction Tools. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)* (pp. 140–148). IEEE. https://doi.org/10.1109/AITest.2019.00017

Rao, P., Zheng, Z., Chen, T. Y., Wang, N., & Cai, K. (2013). Impacts of test suite's class imbalance on spectrum-based fault localization techniques. In *2013 13th International Conference on Quality Software* (pp. 260–267). ISSN: 2332-662X. https://doi.org/10.1109/QSIC.2013.18

Segura, S., Duran, A., Troya, J., & Cortes, A. R. (2017). A template-based approach to describing metamorphic relations. In *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)* (pp. 3–9). IEEE. https://doi.org/10.1109/MET.2017.3

Segura, S., Fraser, G., Sanchez, A. B., & Ruiz-Cortes, A. (2016). A survey on metamorphic testing. *IEEE Transactions on Software Engineering*, *42*(9), 805–824. https://doi.org/10.1109/TSE.2016.2532875

Seth, F. P., Taipale, O., & Smolander, K. (2014). *Organizational and customer related challenges of software testing: An empirical study in 11 software companies* [Paper presentation].2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS), Marrakech, Morocco (pp. 1–12). IEEE. https://doi.org/10.1109/RCIS.2014.6861031

Ur Rehman, F., Umbreen, S., & Rehman, M. (2024). *MetaCDP: Metamorphic testing for quality assurance of containerized data pipelines* [Paper presentation]. 2024 IEEE Cloud Summit, IEEE, Washington, DC, USA. in https://doi.org/10.1109/Cloud-Summit61220.2024.00029

Venkatesh, Thong, Xu, Xu, Consumer acceptance and use of information technology: extending the unified theory of acceptance and use of technology, *MIS Quarterly 36* (1) (2012). 157. https://doi.org/10.2307/41410412

Venkatesh, V., & Davis, F. D. (2000). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science*, *46*(2), 186–204. https://doi.org/10.1287/mnsc.46.2.186.11926

Venkatesh, V., & Thong, J. (2016). Unified theory of acceptance and use of technology: A synthesis and the road ahead. *Journal of the Association for Information Systems.* 17 (5) 328–376. https://doi.org/10.17705/1jais.00428

Venkatesh, V., Morris, M. G., Davis, G.B., & Davis, F. D. User acceptance of information technology: Toward a unified view, *MIS Quarterly 27* (3) (2003). 425. https://doi.org/10.2307/30036540

Walldén, S., Mäkinen, E., & Raisamo, R. (2016). A review on objective measurement of usage in technology acceptance studies. *Universal Access in the Information Society*, *15*(4), 713–726. https://doi.org/10.1007/s10209-015-0443-y

Wu, P., Xiao-Chun, S., Jiang-Jun, T., & Hui-Min, L. (2005). Metamorphic testing and special case testing: A case study. *Journal of Software*, *16*(7), 1210. https://doi.org/10.1360/jos161210

Xiang, Z., & Wu, W. (2018). *The Willingness to use the campus express delivery service based on UTAUT* [Paper presentation]. 2018 5th International Conference on Industrial Economics System and Industrial Security Engineering (IEIS), Toronto, ON (pp. 1–5). IEEE. https://doi.org/10.1109/IEIS.2018.8597792

Yamashita, K., McIntosh, S., Kamei, Y., Hassan, A. E., & Ubayashi, N. (2015). Revisiting the applicability of the Pareto principle to core development teams in open source software projects. In *Proceedings of the 14th International Workshop on Principles of Software Evolution* (pp. 46–55). ACM. https://doi.org/10.1145/2804360.2804366

Zhang, X.-Y., Liu, Y., Arcaini, P., Jiang, M., & Zheng, Z. (2024). MET-MAPF: A metamorphic testing approach for multi-agent path finding algorithms. *ACM Transactions on Software Engineering and Methodology*, *33*(8), 1–37. https://doi.org/10.1145/3669663

Zhou, Z. Q., & Sun, L. (2019). Metamorphic testing of driverless cars. *Communications of the ACM*, *62*(3), 61–67. https://doi.org/10.1145/3241979

Zhou, Z. Q., Xiang, S., & Chen, T. Y. (2016). Metamorphic testing for software quality assessment: A study of search engines. *IEEE Transactions on Software Engineering*, *42*(3), 264–284. https://doi.org/10.1109/TSE.2015.2478001

Zhu, H. (2015). *JFuzz: A tool for automated Java unit testing based on data mutation and metamorphic testing methods* [Paper presentation]. 2015 Second International Conference on Trustworthy Systems and Their Applications, Hualien, Taiwan. IEEE. https://doi.org/10.1109/TSA.2015.13