Metamorphic Testing Among Open-Source Software Developers: A Quantitative Correlational Study

Dissertation Manuscript

Submitted to Northcentral University

School of Technology

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

by

BRITTANY RAFAELA HOARD

La Jolla, California

December 2020

Approval Page

Metamorphic Testing Among Open-Source Software Developers: A Quantitative Correlational Study

By

BRITTANY RAFAELA HOARD

Approved by the Doctoral Committee:

DocuSigned by: Milton Kabia BA3218B4929B445...	PhD, DHA	03/24/2021 | 07:55:27 MST
Dissertation Chair: Milton Kabia	Degree Held	Date
DocuSigned by: Christopher Weppler 08612569C0724F6...	DCS	03/23/2021 | 19:56:57 MST
Committee Member: Christopher Weppler	Degree Held	Date
DocuSigned by: Efosa Osayamwen 204F34E04EA5424...	PhD	03/23/2021 | 15:12:52 MST
Committee Member: Efosa Osayamwen	Degree Held	Date

Abstract

The study addresses the problem of the inadequacy of conventional software testing methods to detect all software defects. This problem affects software users and researchers due to poor software performance, reduced precision or accuracy of software output, and retractions of research publications. Detecting software defects may also be challenging due to the oracle problem. Existing research supports the metamorphic testing method's effectiveness for handling the oracle problem and finding software defects that conventional testing methods cannot detect. The research questions ask about the relationships among the use and acceptance of the metamorphic testing method among open-source developers and the constructs of performance expectancy and effort expectancy. The study's purpose was to examine these relationships. Another objective of the study was to understand how the variables of age, gender, and experience moderate these relationships. The guiding theoretical framework of the study was the unified theory of acceptance and use of technology. In this study, a quantitative methodology with a correlational design was employed. The participants were contributors to open-source software projects contained in the GitHub "Software in science" collection. The data was collected via an online survey instrument. The data were analyzed using Spearman's rank-order correlation tests and moderated multiple regression analysis. Moderate to strong positive relationships were found between both the performance expectancy and effort expectancy and the acceptance and use of metamorphic testing. This finding suggests that increasing the extent to which developers believe that metamorphic testing will improve their job performance and improving its ease of use will increase the adoption of metamorphic testing. The creation of interventions to educate developers on the use of metamorphic testing is recommended. The study results support the applicability of the unified theory of acceptance

and use of technology to software testing methods. Future research could involve studying the relationship between metamorphic testing adoption and other factors, such as social influence and facilitating conditions.

Acknowledgements

First and foremost, I would like to thank my Dissertation Chair, Dr. Milton Kabia, for expertly guiding me through the dissertation process. You went above and beyond to review my work, answer my e-mails, and schedule meetings promptly. It was a great pleasure to work with you. Also, I would like to thank my other dissertation committee members, Dr. Christopher Weppler and Dr. Efosa Osayamwen, for taking the time to review my work and provide helpful comments. Next, I would like to thank my Faculty Mentors, namely Dr. Brian Holbert, Dr. Bhanu Kapoor, Dr. Frank Appunn, Dr. John Enamait, and Dr. Danielle Babb. Your expertise and feedback were invaluable throughout my doctoral journey. Thank you to Dr. J. Robert Sapp for the useful discussion we had regarding my dissertation topic at the Dissertation Boot Camp I attended.

Finally, I would like to acknowledge all the faculty and staff who have provided vital services to NCU students, including but not limited to the Library, the Center for Teaching and Learning, IT Support Services, the Institutional Review Board, and the Dissertation Boot Camps. I would like to extend a special thank-you to my Academic and Finance Advisor, Mr. Michael Edwards, for easing the process of course registration and handling financial matters. Completing this degree program would not have been possible without all of you.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Software testing is an essential component of the software development life cycle. There are various challenges associated with the process of testing software for defects. 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 (Chen et al., 2018). The oracle problem applies to many types of scientific software. Another significant challenge for software testing is the ability to detect all defects in a software product (Lidbury et al., 2015; Rao et al., 2013; Shahri et al., 2019). The development of software testing methods that can effectively address these challenges is an important area of research.

One promising approach to addressing the oracle problem and enhancing defect detection effectiveness is metamorphic testing (MT). 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 input changes in a specific way. The concept of MT was introduced over 20 years ago in a technical report (Chen et al., 1998). The MT method has been applied to a variety of domains since then. 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 many significant advantages, including conceptual simplicity, straightforward implementation, and cost-effectiveness. There are also several challenges associated with the use of MT, including a lack of MT tools, difficulties with systematically identifying and selecting useful MRs, and problems with generating effective test cases. Kanewala and Chen (2018) explained, using a testing theory framework, why MT is preferable for testers who lack software development experience and how programs without test oracles can be tested using MT. They noted that the concept of MT is

easy to learn and that tests that employ MT tend to be easy to implement once MRs are selected. The advantages and drawbacks associated with MT use could affect a software developer or tester's decision regarding whether to use MT for testing their software products.

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

One theoretical framework that could be used to understand the acceptance and use of MT better is the unified theory of acceptance and use of technology (UTAUT). This theory was initially developed to explain why individuals decide whether to accept and use software system innovations. In their seminal work, Venkatesh et al. (2003) developed UTAUT as a unifying model that integrates elements of eight previous models of information technology (IT) acceptance. The UTAUT specifies that a set of constructs directly determines user acceptance of new technology. Two UTAUT constructs that are highly relevant to MT are performance expectancy (PE) and effort expectancy (EE). Performance expectancy is the extent to which the user believes that the new technology will improve their job performance. Effort expectancy is the perceived ease of use of the new technology. The UTAUT specifies four moderating variables, namely age, gender, experience, and voluntariness of use.

Since its development, the UTAUT has been employed in a wide variety of contexts. The theory has been applied extensively in studies that examined mobile payment services (Al-Saedi et al., 2019). The UTAUT has also been employed to understand the readiness of local governments to adopt smart city technology (Gunawan, 2018), students' acceptance of mobile learning systems (Almaiah et al., 2019), and the willingness of potential customers to use delivery services (Xiang & Wu, 2018). The theory was recently employed to study the acceptance of open-source software (OSS) across organizations (Alrawashdeh et al., 2019).

Venkatesh et al. performed a thorough review of the literature that employed UTAUT and assessed the theory's quality using an existing framework for information system (IS) theory evaluation (2016). They concluded that UTAUT is a high-quality theory that performs exceptionally well in the areas of importance, falsifiability, novelty, and defining its parts. Venkatesh et al. (2016) stated that 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. They recommended that further research be performed on different technologies, user groups, and organizational contexts to improve the generalizability of UTAUT. Although UTAUT has been extensively validated in the literature, it has not been generalized to the use of specific software testing methods. The successful application of UTAUT to the topic of the acceptance and use of MT would improve the theory's generalizability.

Statement of the Problem

The problem that was addressed by this study is the inadequacy of conventional software testing methods to detect all software defects, which could be mitigated by the use of the MT method (Lidbury et al., 2015; Rao et al., 2013). The finding supports the existence of the

problem that the MT method has been able to find defects in widely used and 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; Shahri et al., 2019). Another example of the problem is the detection of new defects that had not been previously found in the widely used Siemens test suite despite this suite being the subject of testing research for 20 years (Rao et al., 2013). The problem is significant because defects in software can have severe consequences for software users and researchers. These consequences include poor software performance, reduced precision or accuracy of software output, and retractions of research publications for which defective software was used (Kanewala & Chen, 2018). 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). Detecting software defects can be challenging due to various factors such as the oracle problem, code complexity, and constraints on time and resources (Chen et al., 2018; Ding et al., 2016; Zhou & Sun, 2019). The MT method is especially useful for handling the oracle problem (Kanewala & Chen, 2018; Lin et al., 2018).

Purpose of the Study

The purpose of this quantitative correlational study was to address the problem of the inadequacy of software testing methods for detecting all defects by enabling the researcher to examine relationships between the use and acceptance of the MT method among open-source developers and the constructs of performance expectancy (PE) and effort expectancy (EE). These constructs may be related to MT use and acceptance. Another objective of the study was for the researcher to understand how the variables of age, gender, and experience moderate these relationships. The study purpose aligns with the research problem because the MT method has

been found to detect software defects that other testing methods failed at finding (Lidbury et al., 2015; Rao et al., 2013; Shahri et al., 2019). An improved understanding of the factors associated with the acceptance and use of MT is helpful for designing interventions to increase MT usage and hence reducing the number of software defects. Furthermore, this study's results are useful for assessing the applicability of the unified theory of acceptance and use of technology (UTAUT) to the acceptance and use of software testing methods.

Introduction to Theoretical Framework

The study's theoretical framework is the unified theory of acceptance and use of technology (UTAUT). The UTAUT was developed to explain why individuals decide whether to accept and use new technology (Venkatesh et al., 2003). 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. A fourth construct called facilitating conditions (FC) determines the new technology's actual usage but not the BI. The framework contains four variables that moderate these constructs' relationships, namely age, gender, experience, and voluntariness of use.

The main concepts relevant to the study are the PE, EE, BI, and actual usage. 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. The BI can be used as a measure of user acceptance of the technology. The technology of interest is metamorphic testing (MT) in the context of this study. In addition, the variables of age, gender, and experience are relevant to the study because they moderate

relationships among the PE, EE, and BI according to the theoretical framework (Venkatesh et al., 2003).

The theoretical framework helped guide the study's direction by outlining factors commonly accepted with technology usage and acceptance. The researcher employed this framework, combined with the literature review, to select a set of constructs that may be relevant to the acceptance and usage of MT. These constructs were used to develop the research questions and study purpose.

In this study, the researcher determined whether the UTAUT is useful for understanding the acceptance and use of a software testing method. Before this study was performed, the UTAUT had not been applied to the adoption of software testing methods. The researcher examined the relationships between actual usage, the BI, the PE, and the EE. The results suggest that significant relationships among these variables exist, so it can be concluded that UTAUT can be generalized to the adoption of software testing methods. The UTAUT constructs of PE and EE are based mainly on the technology acceptance model (TAM) variables of perceived usefulness (PU) and perceived ease of use (PEOU), which are the primary determinants of technology acceptance in the TAM (Davis, 1986). Thus, the study's findings also provided information about the relevance of the TAM to the acceptance of software testing methods. The hedonic-motivation system adoption model (HMSAM) contrasts with the UTAUT and the TAM because it was designed for application to hedonic-motivation systems rather than utilitarian-motivation systems (Lowry et al., 2013). However, the HMSAM also includes the PU and PEOU, so the study's findings provided information about the applicability of this portion of HMSAM to software testing methods.

Introduction to Research Methodology and Design

The researcher employed a quantitative methodology to carry out the study. A quantitative method was used because it is suitable for validating existing theories, testing hypotheses, and measuring known constructs (Choy, 2014; Creswell, 2014). Quantitative research methods align with the study's purpose, including testing the applicability of UTAUT to software testing methods and measuring known variables. Quantitative research is ideal for answering the research questions, which ask about the existence of relationships among known variables and theoretical constructs. Furthermore, the research questions were addressed using hypothesis testing, which falls under quantitative methodology.

A correlational research design was used to carry out this research. A correlational design was the most appropriate research design for this study because the purpose of the study is to examine relationships between variables (Creswell, 2014). These variables are the acceptance and use of MT and two UTAUT constructs, namely performance expectancy (PE) and effort expectancy (EE) (Venkatesh et al., 2003). Furthermore, the research questions ask about these relationships. The study design is appropriate for addressing the research problem by examining the correlation between the use and acceptance of the MT method, which effectively finds defects, and factors that could be related to the use and acceptance of the MT method.

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). A custom survey instrument was used to collect data from a sample of the target population. This population consisted of contributors to OSS projects on GitHub, a public repository for OSS projects. Venkatesh et al. (2003) published and validated an

instrument to collect data about UTAUT constructs. The instrument used in the study was like it; however, the survey items were customized to suit the study objectives. There were survey items designed to elicit information regarding the acceptance and use of MT and the moderating variables of age, gender, and experience. Due to the cost-effectiveness and ease of use of online surveys, the survey was hosted on the Internet. A link to the survey was e-mailed to the participants, along with information about the study.

A sample size of 41 participants was obtained for this study. Based on a power analysis for which the sample size is 41, the statistical power is 0.8, and the significance level is 0.05, the minimum correlation coefficient that can be detected is 0.42. Based on an existing study in which GitHub contributors were surveyed, the expected response rate was approximately 24% (Kalliamvakou et al., 2016). Given this response rate, the survey was sent to 700 contributors to ensure an adequate sample size.

Data analysis consisted of two phases: the preliminary analysis and the post-collection analysis. The initial analysis entailed using wave analysis to check for response bias (Phillips et al., 2016). The post-collection analysis took place after data collection was complete. Correlation statistics were calculated to determine the strength of the relationships among the variables and the direction of these associations.

Research Questions

RQ1

To what extent is there a relationship between the acceptance of metamorphic testing (MT) among open-source software developers and the following constructs: (a) performance expectancy and (b) effort expectancy?

RQ2

To what extent is there a relationship between the frequency of use of MT among open-source software developers and the following constructs: (a) performance expectancy and (b) effort expectancy?

Hypotheses

$H1_0$

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.

$H1_a$

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.

$H2_0$

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.

$H2_a$

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.

Significance of the Study

The study is important because the researcher used its findings to examine factors that could affect the use and acceptance of MT. The UTAUT, a widely used and well-validated

theory, served as the guiding framework. Achieving the study purpose and answering the research questions resulted in an improved understanding of the relationship between the use and acceptance of MT and UTAUT constructs. This enhanced knowledge could guide the design of interventions to increase MT usage among software developers, software testers, and other potential users. The study findings also enhanced current knowledge regarding the use of MT for open-source software projects. Since MT effectively detects software defects, addressing the research problem of the inadequacy of conventional testing methods for defect detection through the increased use of MT could result in higher quality software products. The researcher also contributed to the body of UTAUT literature by applying UTAUT to a software testing method. The findings of the study provided insight into the generalizability of UTAUT to software testing methods.

Definitions of Key Terms

Information system

An information system is a group of components that work together to gather, create, process, distribute, and store data to support decision making, analysis, communication, control, and visualization (Laudon & Laudon, 2012). Theories to explain user acceptance and use of information systems, such as the TAM and the UTAUT, were developed and applied to various other technologies (Al-Mamary et al., 2015; Davis, 1986; Davis et al., 1989; Venkatesh et al., 2003).

Metamorphic relation

A metamorphic relation specifies how software output should change when the input is changed in a specific way (Kanewala & Chen, 2018). Metamorphic relations are the foundation of metamorphic testing.

Metamorphic testing

Metamorphic testing is a software testing method that checks that metamorphic relation(s) among a set of tests is satisfied (Kanewala & Chen, 2018). Metamorphic testing is useful for addressing the oracle problem and detecting defects (Chen et al., 2018; Ding et al., 2016; Kanewala & Chen, 2018; Lin et al., 2018).

Oracle problem

The oracle problem is the difficulty inherent in testing software output for correctness due to the lack of a test oracle to compare it (Chen et al., 2018). The oracle problem is a common challenge associated with scientific testing software (Kanewala & Bieman, 2014).

Scientific software

Scientific software is developed for scientific purposes; it is primarily used to understand and make predictions about scientific processes in the real world (Kanewala & Bieman, 2014). It is used to perform research and help make critical decisions (Kanewala & Chen, 2018; Yang et al., 2018).

Software defect

A software defect is a fault in a software product that results in not meeting a software specification or end-user requirement (Burnstein, 2003). Software defects can have severe consequences for users (Kanewala & Chen, 2018).

Software development life cycle

A software development life cycle is a series of phases to develop and revise a software product (Everett & McLeod, 2007). This life cycle includes planning, design, implementation, testing, deployment, and maintenance.

Software testing

Software testing is the process of validating a software product against a set of requirements or expected behaviors (Everett & McLeod, 2007). Testing software is vital to ensure its performance, precision, and accuracy (Kanewala & Chen, 2018; Przedzinski et al., 2020).

Test oracle

A test oracle is a mechanism used to verify software test output (Howden, 1978). Test oracles do not exist for all software programs, making testing difficult (Chen et al., 2018).

Summary

In this quantitative correlational study, the researcher addressed the inadequacy of conventional software testing methods for defect detection by examining the relationships between the use and acceptance of MT and constructs from UTAUT, a widely used and well-validated theory of technology acceptance and use. A cross-sectional survey method was used to measure the variables of interest. A custom survey instrument was developed to collect data from the target population, which consisted of contributors to OSS projects on GitHub. The correlation statistics were calculated during data analysis. The study findings could help researchers and practitioners develop interventions for potential MT users and provide insight into the generalizability of UTAUT to software testing methods.

Chapter 2: Literature Review

Introduction

The literature review is organized to address the study's purpose: to look at the use and acceptance of MT among open-source software developers to improve the quality assurance of software with UTAUT as the guiding theoretical framework. The literature review covers various topics relating to the study problem and purpose. These topics include MT, scientific software, software testing, open-source software development, UTAUT, and other theoretical frameworks.

In the first section of the literature review, the researcher's methods to locate relevant literature are described, including keywords, search terms, Boolean logic, and databases searched. In the second section of the literature review, the researcher discusses the guiding theoretical framework, UTAUT, a complementary framework, namely the technology acceptance model (TAM), and the hedonic-motivation system adoption model (HMSAM), a contrasting framework. In the third section of the review, several themes related to the study's purpose are discussed. These themes include scientific software quality assurance, open-source software development, open-source software quality assurance, software testing challenges, MT frameworks, methodologies, effectiveness, and challenges.

Methods of Searching

The use of scholarly databases was necessary for performing a thorough search of pertinent literature. The Roadrunner Search Discovery Service provided by the Northcentral University Library was used to search a few databases, including *EBSCOhost*, *ProQuest*, and Google Scholar. These databases offer up-to-date scholarly publications to the research community, including journal articles, conference proceedings, theses, and dissertations. The

database searches were limited to scholarly/peer-reviewed works. Initial searches on each topic were limited to the publication years 2015-2020 to focus on the most up-to-date literature. After the initial search, the researcher performed another search without any date delimitations to identify works that were less recent but still important. The search results were sorted by relevance. Keywords used in the literature search included software testing, metamorphic testing, user acceptance of technology, UTAUT, unified theory of acceptance and use of technology, technology acceptance model, hedonic-motivation system adoption model, open-source software, OSS, software quality, and scientific software.

The researcher employed Boolean search strategies to locate articles that were highly relevant to software quality, MT, UTAUT, TAM, HMSAM, software testing, open-source software, and scientific software. Using Boolean search phrases such as metamorphic testing AND frameworks, metamorphic testing AND open-source software, open-source software AND (development OR developer), metamorphic testing AND scientific software, and (software quality OR software testing OR software defects), the researcher was able to narrow down a large number of search results to the most appropriate resources and to broaden the search of relevant literature when needed to find additional resources. Other Boolean search phrases used in the literature search included but were not limited to the following: HMSAM OR (hedonic-motivation system adoption model), TAM OR (technology acceptance model), UTAUT OR (unified theory of acceptance and use of technology), (quality OR testing OR defects) AND scientific software, (software quality OR software testing OR software defects) AND (UTAUT OR (unified theory of acceptance and use of technology), (open-source software OR OSS) AND (UTAUT OR (unified theory of acceptance and use of technology).

Without any inclusion criteria applied, a database search yielded over 1500 works on the topic of metamorphic testing, over 2900 works on user acceptance of technology, over 200,000 works on scientific software quality, and over 311,000 works on open-source software quality. When the inclusion criteria of peer-reviewed publications with date delimitations between 2015 and 2020 were applied, the researcher found 899 results for metamorphic testing, 1450 results for user acceptance of technology, 17,300 results for scientific software quality, and 61,000 works for open-source software quality. After each search using the inclusion criteria, which included keywords or Boolean search phrases, the researcher reviewed the abstracts of the first 50 works, sorted by relevance, for their applicability to this study.

The exclusion criteria used in the literature search included works on software testing and software quality that are not directly related to metamorphic testing, open-source software, scientific software, testing challenges, or software defects. Other exclusion criteria included studies related to user acceptance of technology but are not relevant to UTAUT, TAM, HMSAM, software quality, software testing, or open-source software. Even though they contained the search keywords, any resources that were not relevant to the topics covered in this literature review were also excluded.

Most of the literature review resources were published within the last five years (2015-2020). Scholarly journal articles and conference proceedings provided information on state-of-the-art research. Seminal works introduced and described essential concepts and frameworks such as MT, TAM, HMSAM, and UTAUT. Survey articles on these topics provided an overview of the current literature and insights derived from synthesizing it. The researcher identified 70 additional resources by manually searching the bibliographies of survey articles and seminal works. A total of 152 resources were identified for inclusion in this literature review. The

researcher did not locate any resources relating to the acceptance and adoption of metamorphic testing. One of the objectives of this study was to address this gap in the literature.

Theoretical Perspective of the Study

Unified Theory of Acceptance and Use of Technology

The unified theory of acceptance and use of technology (UTAUT) is a theory that was developed to explain why individuals decide whether to accept and use innovations in software systems. In their seminal work, Venkatesh et al. (2003) developed UTAUT as a unifying user acceptance model that integrates elements of eight previously developed models. These models are as follows: theory of reasoned action, technology acceptance model, theory of planned behavior, a model that combines the technology acceptance model and the theory of planned behavior, innovation diffusion theory, motivational model, model of PC utilization, and social cognitive theory (Ajzen, 1991; Bandura, 1986; Davis, 1989; Davis et al., 1992; Fishbein, 1967; Rogers, 1995; Taylor & Todd, 1995; Thompson et al., 1991; Venkatesh & Speier, 1999).

As a starting point for the development of UTAUT, Venkatesh et al. (2003) reviewed IT user acceptance literature and assessed the similarities and differences between the eight older user acceptance models. The authors performed a within-subjects longitudinal comparison and validation of these models to evaluate their explanatory power. The researchers formulated UTAUT based on the similarities across these eight models. The authors then empirically validated UTAUT by comparing its performance to each of the original models' performance. For this performance study, the authors gathered data from four organizations for six months relating to a new technology introduced to employees. After data collection was complete, the researchers analyzed how well each of the models explained the variance in the employees'

intentions to use the new technology. The authors found that UTAUT outperformed the other eight models.

Furthermore, the researchers cross-validated the UTAUT by applying the models to new data from two different organizations. The authors concluded that UTAUT is useful for understanding the factors underlying technology acceptance so that effective interventions targeted at users who are less likely to adopt new technologies can be designed. They recommended further research relating to different types of technology, user groups, and organizational contexts to improve the generalizability of UTAUT.

The UTAUT stipulates that four constructs directly determine user acceptance or usage of new technology. These constructs are as follows: performance expectancy (PE), effort expectancy (EE), social influence (SI), and facilitating conditions (FC). The PE construct encompasses the extent to which the user believes that the new technology will help improve their job performance or result in other favored outcomes, such as increased salary or promotion (Compeau et al., 1999; Davis, 1989; Davis et al., 1992; Moore & Benbasat, 1991; Thompson et al., 1991). After validating UTAUT, Venkatesh et al. (2003) found that the PE was the strongest predictor of a user's intention to use a new technology across all user acceptance models. The EE construct includes the user's perceived ease of use, ease of learning to use, and understanding of the new technology (Davis, 1989; Moore & Benbasat, 1991; Thompson et al., 1991). The SI construct includes the extent to which the user believes that other people think they should use the new technology and how they feel their social status will be affected by using the technology. This construct also encompasses the organization's perceived supportiveness for using the new technology (Ajzen, 1991; Moore & Benbasat, 1991; Thompson et al., 1991). The FC construct encompasses self-efficacy, constraints on user behavior, compatibility of the new technology

with the user's tasks and work style, and the extent to which the user believes that adequate technical and organizational infrastructure exists to support the latest technology (Ajzen, 1991; Moore & Benbasat, 1991; Thompson et al., 1991). Unlike the other three constructs, the FC construct is not a significant predictor of intention to use, but it is a significant predictor of actual use (Venkatesh et al., 2003).

In addition to the four primary constructs, UTAUT stipulates four key moderating variables, namely age, gender, experience with the new technology, and voluntariness of using the technology. When validating UTAUT, Venkatesh et al. (2003) found that the PE construct was significant regardless of experience or voluntariness of use. The significance of the EE construct declined as experience increased. Furthermore, the researchers observed that the importance of the PE and EE constructs was moderated by age and gender; the PE construct was more important for younger and male study participants, and the EE construct was more important for older and female participants. The authors found that the SI construct was more important for older and female participants when the new technology was mandatory and when the user lacked experience with the technology.

Since its development, UTAUT has been used and validated in a wide variety of applications. The UTAUT is one of the most popular models among researchers for predicting technology use and acceptance (Al-Mamary et al., 2015). Recent reviews of empirical studies that apply UTAUT found that this theory's quality is high, robust, and valid (Venkatesh et al., 2016; Walldén et al., 2016). Venkatesh et al. (2016) found that it performs exceptionally well in the areas of novelty, importance, falsifiability, and definition of its parts. The theory has been employed extensively in studies that examine the acceptance and adoption of social media and mobile payment services (Al-Qaysi et al., 2020; Al-Saedi et al., 2019). Besides, the UTAUT was

used to understand 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). The theory has been used to propose evaluating a tool that helps patients with social cognition deficits (García et al., 2019). Although the UTAUT has not been applied to software testing methods, it has been used for studying processes and practices for managing and developing software (Anderson, 2019; Guardado, 2012). Recently, the UTAUT was employed to analyze the acceptance of open-source software (OSS) across organizations (Alrawashdeh et al., 2019).

Studies that use UTAUT as a theoretical framework employ quantitative methodology with a correlational or regression research design (Almaiah et al., 2019; Alrawashdeh et al., 2019; García et al., 2019; Gunawan, 2018; Xiang & Wu, 2018). A review of mobile payment service adoption studies that applied UTAUT found that survey questionnaires were used to collect data in 80% of the studies (Al-Saedi et al., 2019). In a recent survey of information systems (IS) studies that employed UTAUT, Venkatesh et al. (2016) found that the effects of the standard UTAUT moderating variables are rarely included in these studies' research design. In many studies that applied UTAUT, the researchers modified or extended the theory by using constructs or moderating variables other than the ones specified by UTAUT to understand specific relationships better. In their meta-analysis, Dwivedi et al. (2017) found that only 25% of UTAUT studies did not include additional constructs and suggested that attitude should be included as a mediating construct in UTAUT. Perceived risk and perceived trust were commonly used as constructs in studies on mobile payment adoption, and IT specialty was used as a moderator in a study on the use and acceptance of OSS (Alrawashdeh et al., 2019; Al-Saedi et al., 2019). Venkatesh et al. (2012) designed UTAUT2, an extended version of the theory to study

technology acceptance and usage in consumer contexts. The UTAUT2 includes the additional constructs of price value, habit, and hedonic motivation.

Although UTAUT has been extensively validated and widely used to study technology acceptance and use, the theory has not been without criticism. In a systematic review of IS studies that utilized UTAUT, Venkatesh et al. (2016) used a systematic framework of theory evaluation to evaluate UTAUT. They concluded that, although the quality of the theory overall is high, its parsimony is low. This criticism was shared by Bagozzi (2007) in a critique of the TAM and its extensions. Venkatesh et al. (2008) criticized the UTAUT constructs of FC and behavioral intention (BI) because these constructs excluded external behavioral determinants that could change over time, such as limitations in user ability and environmental factors. Furthermore, these constructs had weak predictive power when information regarding behavior is incomplete. The authors proposed the new construct of behavioral expectations to address these limitations.

In this study, the researcher assessed whether UTAUT applies to the acceptance and use of software testing methods, specifically MT. The study's purpose was for the researcher to examine the relationships between two UTAUT constructs, namely PE and EE, and the acceptance and usage of MT. The research questions, along with their associated hypotheses, focus on these relationships. The extension of UTAUT to software testing methods addresses the research problem of the inadequacy of conventional software testing methods because it can improve the current understanding of the factors associated with adopting specific software testing methods. This increased knowledge could help develop interventions that increase the usage of more effective testing methods, which could lead to a decline in defects and an improvement in software quality. The UTAUT was the theoretical basis for the dissertation

research design. The UTAUT is the most widely used and validated technology acceptance model and has been shown to outperform other technology acceptance models.

Technology Acceptance Model

The technology acceptance model (TAM) predicts and explains user acceptance and usage of information systems and computing technologies. It is one of the eight user acceptance models on which UTAUT is based. The TAM was developed to provide a parsimonious acceptance model that was general enough to be successfully applied to a wide range of computing technologies and users (Davis, 1986; Davis et al., 1989). The TAM adapts to the more general theory of reasoned action (TRA) (Ajzen & Fishbein, 1980; Fishbein & Ajzen, 1975). In contrast to the TRA, the TAM is used only to predict and explain the usage of computing technologies.

In their seminal work, Davis (1986) proposed the TAM to enhance understanding of user acceptance of new systems, which would improve the success rate of system design and implementation. Another objective of the TAM was to provide the theoretical underpinning to user acceptance testing, in which prototypes of new systems would be demonstrated to potential users who would then be questioned about their intention to use the latest technologies. Davis (1986) selected the TRA as the TAM's theoretical framework and modified the TRA to suit the context of computing technology acceptance and usage. The TRA is a behavioral intention model from social psychology that has been used to explain human behavior in many different domains (Ajzen & Fishbein, 1980; Fishbein & Ajzen, 1975). Next, Davis (1986) reviewed literature in social factors and management information systems to locate empirical evidence for the proposed TAM and developed measures for the model variables. These variables were validated via a field survey of organizational end-users. Finally, an experiment that tested student

acceptance of two graphics systems was conducted to verify the TAM structure and check proposed modifications to the model.

The TAM stipulates that two variables, namely perceived usefulness (PU) and perceived ease of use (PEOU), are the primary determinants of technology acceptance behaviors. The PU is the extent to which the user believes that using the new technology will improve their job performance (Davis, 1986). The PEOU is the degree to which the user accepts that using the latest technology will be effortless. The PU and the PEOU are like the UTAUT constructs of the PE and the EE, respectively. However, because UTAUT was developed as a unifying model for TAM and seven other models, the UTAUT constructs are broader than the TAM variables. The PE construct includes not only the PU but also additional favored outcomes, such as increased salary or promotion (Compeau et al., 1999; Davis, 1989; Davis et al., 1992; Moore & Benbasat, 1991; Thompson et al., 1991). The EE construct encompasses the perceived ease of learning to use and understand the new technology and the PEOU. In the TAM, the PEOU is one of the determinants of the PU, whereas, in the UTAUT, the PE and EE constructs are not significantly related. The TAM specifies that the PEOU and the PU may be affected by external variables, such as system design characteristics, system features, or feedback-based learning (Davis et al., 1989). The four moderating variables in the UTAUT are absent from the TAM.

There are similarities and differences between the UTAUT and the TAM regarding the relationships between behavioral intention (BI) and the other model constructs. Like the UTAUT, the TAM stipulates that the BI determines actual usage behaviors (Davis et al., 1989; Venkatesh et al., 2003). However, in the UTAUT, the construct of facilitating conditions (FC) also determines usage behaviors. Furthermore, in the TAM, the BI is determined by two variables: the PU and the user's attitude toward using the new technology (A). The A construct is

itself determined by the PU and the PEOU. In the UTAUT, the BI is determined by three constructs: PE, EE, and social influence (SI). The A construct is not a component of UTAUT.

Some researchers have pointed out methodological weaknesses in studies that employ the TAM or a TAM derivative. One common flaw in studies that employ the TAM or UTAUT is that subjective measurements such as self-reporting are frequently used to measure technology usage, with objective measurements of usage seldomly used (Legris et al., 2003; Walldén et al., 2016). The difference between objective and subjective usage measures can be significant and affect the study findings relating to the relationships between model constructs (Straub et al., 1995; Wu & Du, 2012). The BI is also measured more often than actual usage (Turner et al., 2010). A meta-analysis performed by Wu and Du (2012) found that the BI was less correlated with usage when usage was measured objectively; they suggested that more TAM studies measure usage. Cross-sectional studies do not consider the possibility that the user's intention, perceptions, and use may change over time. Many TAM studies were laboratory studies, which means that these studies' findings are not necessarily generalizable to real-world scenarios (Yousafzai et al., 2007).

Although the TAM has been extensively validated, its simplicity has led to many researchers extending the original model to suit the objectives of their studies in educational, translation, and mobile technology contexts. Examples of the additional variables and constructs added to these extensions include perceived organizational culture, organizational policy, trust, self-efficacy, perceived compatibility, perceived satisfaction, perceived security, perceived task-technology fit, learning style, and motivation (Al-Azawei et al., 2017; Al-Maroof, 2020; Huang & Teo, 2019; Leong et al., 2018; Shankar & Kumari, 2019). The relationships between the TAM constructs have been replicated by other research teams (Adams et al., 1992; Hendrickson et al.,

1993; Segars & Grover, 1993; Subramanian, 1994; Szajna, 1994). Furthermore, the reliability, internal consistency, and validity of the survey instrument used by Davis (1989) were demonstrated by other researchers (Adams et al., 1992; Hendrickson et al., 1993; Szajna, 1994). However, the TAM has often been criticized for its low predictive and explanatory power in various contexts (Ajibade, 2018; Chandio et al., 2017; Chuttur, 2009; Hai & Alam Kazmi, 2015; Hojjati & Khodakarami, 2016; Legris et al., 2003; Li, 2020; Lim et al., 2016).

In addition to these extensions, several significant modifications to the original TAM have been developed to address its weaknesses, including TAM2, TAM3, and the UTAUT (Venkatesh & Bala, 2008; Venkatesh & Davis, 2000; Venkatesh et al., 2003). The TAM2 adds the constructs of social influence and cognitive instrumental processes as determinants of the PU. The social influence construct includes voluntariness, image, and subjective norm, which is the user's perception that important people believe they should use the new technology (Fishbein & Ajzen, 1975). The TAM2 posits that subjective norm is a determinant of the BI in addition to the PU. The construct of cognitive instrumental processes encompasses the PEOU, output quality, job relevance, and demonstrability of results. The TAM2 also posits experience and perceived voluntariness as moderators of the relationship between subjective norm and BI. The UTAUT, which has been discussed previously, was developed to unify TAM with seven other acceptance models and improve TAM's explanatory and predictive power.

The TAM3 combines the TAM2 with research findings suggesting that the constructs of perceived enjoyment, objective usability, computer self-efficacy, computer anxiety, computer playfulness, and perceptions of external control be added to the model as determinants of the PEOU (Venkatesh & Bala, 2008; Venkatesh & Davis, 2000). The TAM3 also posits that experience moderates more construct relationships than in the previous TAM models. The

TAM3 was created to have more potential for actionable guidance about creating effective interventions than the original TAM; consequently, the TAM3 is more comprehensive but less parsimonious than the original model. Although the TAM3 is successful at explaining the variance in the PU and the PEOU constructs, its explanatory power with regards to behavioral intention and usage is lower than that of UTAUT (Venkatesh & Davis, 2000; Venkatesh et al., 2003). Due to the lower explanatory and predictive power of TAM and its derivatives than UTAUT, the latter model was chosen to be the main theoretical framework for this study.

Hedonic-Motivation System Adoption Model

The hedonic-motivation system adoption model (HMSAM) was proposed by Lowry et al. (2013) to study the acceptance and use of a hedonic-motivation system (HMS), which is a technological system that is used mainly to satisfy intrinsic motivations. In other words, an HMS is adopted for enjoyment rather than utility. In its focus on HMS, the HMSAM contrasts with models like the TAM and UTAUT, which focus on the adoption of a utilitarian-motivation system (UMS) (Davis, 1986; Venkatesh et al., 2003). The HMSAM was developed to address the lack of IS research on HMS despite the substantial increase of HMS usage throughout the 2000s in the form of gaming systems, personal computing, mobile computing, and social media (Brown & Venkatesh, 2005; Hsu & Lu, 2007; Jegers, 2007). The necessity of developing models specifically for HMS acceptance and usage stems from the fundamental distinction between the reasons for using an HMS and those for using a UMS. The HMS users are driven chiefly by the promise of intrinsic rewards and the user experience, whereas UMS users are motivated primarily by external rewards and specific usage outcomes (Sweetser & Wyeth, 2005; Venkatesh et al., 2003). However, researchers have recognized that intrinsic motivation may also play a role

in UMS adoption; constructs representing intrinsic rewards have been added to UMS acceptance models such as TAM3 (Venkatesh & Davis, 2000).

The HMSAM has been employed to study the acceptance and use of HMS as well as gamified UMS. The model was used to assess the effects of gamification on a mobile health application, a tool for software requirements gathering, an educational environment for undergraduate students, a security training system, and a 3D multiuser virtual learning environment for training agricultural surveyors (Oluwajana et al., 2019; Saphira & Rusli, 2019; Setiawan & Suryadibrata, 2019; Silic & Lowry, 2020). The HMSAM was employed to evaluate consumer participation in virtual reality (VR) tourism; a moderating variable that specified whether the user was a visitor or non-visitor at the portrayed destination was added to the original model (Kim & Hall, 2019).

The HMSAM is based on the hedonic-system acceptance model, a variation of the TAM proposed by van der Heijden (2004). The hedonic-system acceptance model includes the constructs of joy, the PU, the PEOU, and the BI. One significant difference between HMSAM and this earlier model is the incorporation of the subconstructs of cognitive absorption (CA) into HMSAM. The older hedonic-system acceptance model represents intrinsic motivation only by joy and does not include other types of intrinsic motivation. The CA construct, which was proposed by Agarwal and Karahanna (2000), represents various factors associated with intrinsic motivation, namely joy, curiosity, control, focused immersion, and temporal dissociation. The joy construct represents the fun and pleasurable aspects of HMS usage. The curiosity subconstruct signifies the degree to which the user's experience using the HMS awakens their curiosity. Control is defined as the user's perception that they are in control of their interaction with the system. Focused immersion is the user's experience of complete engagement with the

HMS to the extent that they ignore other stimuli. Finally, temporal dissociation is the user's inability to notice the amount of time elapsed while using the HMS.

Lowry et al. (2013) validated the HMSAM by performing two studies relating to gaming: an experiment involving the use of various scripted scenarios and graphical storyboards and a laboratory experiment involving the use of real-world games. The participants were students at a private university in the United States. The scenarios included three different types of games and various levels of joy and PEOU to which participants were randomly assigned. After this study, the participants took a survey with questions about their hypothetical reactions to each scenario. Based on the results of the first study, the second study involved participants randomly assigned to play four games that were chosen to represent various levels of joy and PEOU.

The HMSAM consists of seven constructs: the PEOU, the PU, the BI, curiosity, immersion, joy, and control (Lowry et al., 2013). The model posits the following relationships among these constructs: joy and curiosity are determinants of immersion, curiosity is a determinant of the BI, the PEOU is a determinant of the PU as well as of curiosity and control, control is a determinant of both immersion and the BI, and immersion and the PU are determinants of the BI. The PEOU and the PU are like the UTAUT constructs of the PE and EE. However, the UTAUT constructs are broader than these TAM-derived constructs. The PE construct includes other favored outcomes and PU, such as increased salary or promotion (Compeau et al., 1999; Davis, 1989; Davis et al., 1992; Moore & Benbasat, 1991; Thompson et al., 1991). The EE construct encompasses the perceived ease of learning to use and understand the new technology and the PEOU. In the HMSAM, the PEOU is a determinant of the PU, whereas, in the UTAUT, the PE and EE constructs are not significantly related. The HMSAM construct of control is included in the UTAUT construct of facilitating conditions (FC). Some

researchers proposed the inclusion of the construct of attitude, which would include joy, in the UTAUT (Dwivedi et al., 2017). The UTAUT2, an extension of the UTAUT intended for consumer contexts, includes the construct of hedonic motivation (Venkatesh et al., 2012).

The HMSAM was not chosen as the main theoretical framework for this study because it is intended for studying HMS rather than UMS. The research problem addressed by this study was the inadequacy of conventional software testing methods for detecting defects. The study's purpose was to examine relationships relating to the acceptance and use of MT, a software testing method. Since the technology of interest in this study is primarily used for work-related purposes rather than enjoyment, the HMSAM is less pertinent to the study than technology acceptance models intended for use with UMS, such as UTAUT.

Review of the Literature

Scientific Software Quality Assurance

Scientific software is used to perform scientific research and make critical decisions and predictions (Kanewala & Bieman, 2014; Kanewala & Chen, 2018; Yang et al., 2018). Defects in scientific software can have severe consequences, including reduced computational performance, decreased accuracy or precision of software output, and publication retractions (Abackerli et al., 2010; Hatton, 1997; Hatton & Roberts, 1994; Kanewala & Chen, 2018; Miller, 2006). Thus, ensuring software quality is an essential component of scientific software development.

Scientific software quality assurance involves addressing challenges unique to scientific software, including code complexity, the oracle problem, approximations in numerical calculations, and cultural differences between software engineers and scientist-developers. Testing scientific software is critical for ensuring precision and accuracy (Przedzinski et al., 2020). Multidisciplinarity in scientific software can lead to overly complicated code, challenging

to validate and maintain (Ober & Ober, 2017). In a systematic literature review, Kanewala and Bieman (2014) found that the oracle problem was one of the leading causes of challenges associated with testing scientific software. The oracle problem is the difficulty or impossibility in verifying the correctness of test case results because a test oracle for the software program does not exist (Chen et al., 2018; Ding et al., 2016). Kanewala and Bieman (2014) found that metamorphic testing (MT) appears to be the most promising solution to testing without oracles. Recent literature has outlined approaches for developing metamorphic relations (MRs) and discussed the researchers' encouraging experiences with using MT for scientific software testing (Ding et al., 2016; Kanewala & Chen, 2018; Lin et al., 2018).

The use of computational approximations in scientific software poses additional quality assurance challenges. Computational approximations are often employed in software to increase a physical problem (Dubois, 2012; Hinsen, 2015; Toronto & McCarthy, 2014). These approximations include physical laws, discretization, floating-point algorithms, and optimization. Such approximations can cause inaccuracies in numerical results and lack of reproducibility due to the accumulation of rounding errors and the rearrangement of floating-point operations by compilers (Diethelm, 2012; Hinsen, 2015; Toronto & McCarthy, 2014). A case study performed by Hinsen (2015) found that these issues make testing scientific software especially challenging. The author suggested that programming languages allow developers to specify floating-point operations' order and assess the impact of optimizations by writing specifications at the floating-point level and testing programs against these specifications. Toronto and McCarthy (2014) explained the framework of floating-point theory to help developers understand how to implement and test floating-point operations. The authors introduced tools for debugging floating-point code. They recommended using a test-debug cycle with two different versions of

the same function, using higher precision numbers. However, the proposed approach was only tested on three small functions rather than on sizeable real-world software programs. Diethelm (2012) discussed the lack of reproducibility of many algorithms that run on high-performance computing (HPC) systems but noted that non-reproducibility is often not considered a problem. The researcher found that the output of a finite element method (FEM) simulation was slightly different when different processors were used to run it, even though the input values remained the same. The author recommended that developers discuss reproducibility issues in software documentation and provide reproducible versions of routines to end-users who need them.

Other issues associated with scientific software quality assurance involve the availability of resources for scientific software projects and differences in software development practices between software engineers and scientist-developers (Basili et al., 2008; Kanewala & Bieman, 2014; Katzman et al., 2018; Kelly et al., 2011; Leman et al., 2020; Méndez et al., 2014; Russell et al., 2019; Storer, 2017). Although not all traditional software engineering practices apply to scientific software projects, studies have shown that scientist-developers could benefit from collaboration with software engineers in various areas. These areas include design, refactoring, documentation, testing, maintenance, optimization, inspection, and project management (Farhoodi et al., 2013; Heaton & Carver, 2015; Kelly et al., 2011; Koteska et al., 2018; Russell et al., 2019; Sanders & Kelly, 2008). One reason for the gap between scientist-developers and software engineers is that scientist-developers tend to learn software development skills from other scientists rather than professional software developers (Basili et al., 2008; Kellogg et al., 2019). Scientist-developers tend to focus on the short-term goal of enabling the software package to meet the immediate requirements of the research and neglect the long-term goals of making the software package robust, maintainable, and generalizable (Morris & Segal, 2009; Riesch et

al., 2020). Three field studies found that scientist-developers who lead software development teams tend to lack an understanding that continued funding is needed to properly maintain the software (Segal, 2008a; Segal, 2008b; Segal, 2009). They also found that scientist-developers often do not allocate additional funding for ensuring that a software package is highly robust and generalizable to research problems other than the one it was initially intended to solve. Similarly, Katzman et al. (2018) noted there is often a lack of funds for the employment of a dedicated team of software developers due to funds decreasing after the initial version of the software is deployed. They also stated that many open-source scientific projects are released but not widely used over time.

Software development frameworks, specifically for use with scientific software projects, have been proposed to ameliorate these challenges (Katzman et al., 2018; Riesch et al., 2020). Riesch et al. (2020) created a template for small to medium-sized scientific software projects written in C++ and Python. This template implements a set of best software engineering practices, including version control, code formatting, static analysis, continuous integration, unit testing, code coverage reporting, and documentation. Katzman et al. (2018) proposed an evolutionary software development model intended to develop and maintain open-source scientific software packages. They created this model from a case study focusing on the eleven-year history of the development of AceTree, an open-source software package for the editing and analysis of nuclear lineage data (Bao et al., 2006; Boyle et al., 2006). A core team of developers, who work closely with those who regularly use the software, perform updates to the software project intermittently throughout its lifetime in response to feedback from its userbase, modifications to any software packages that the project depends on, and hardware updates (Katzman et al., 2018). The model provides a process by which a scientific software project can

remain usable and relevant to its userbase and the research community without requiring a full-time well-funded team of staffers to maintain it. The software project is not entirely dependent on the open-source community to perform essential updates and add new features.

Frameworks have also been proposed for the improvement of scientific software processes and the assessment of scientific software quality, such as the hybrid scientific software process improvement framework (SciSPIF) and a quantitative quality model (Koteska et al., 2018; Mesh et al., 2014; Mesh et al., 2016). Koteska et al. (2018) developed a model that assesses scientific applications' quality using a set of quantitative metrics and attributes. These metrics and attributes include reliability, maintainability, portability, complexity, and modularity. Mesh et al. (2014) proposed SciSPIF to allow scientific software developers to make more informed decisions regarding software development practices to meet software quality goals given real-world constraints such as cost, time, and effort. Software development is costly, especially during the maintenance phase, and each project faces unique constraints (Méndez & Tinetti, 2017; Mesh et al., 2016). One goal of SciSPIF is to understand how scientific software development teams make decisions and share this knowledge with other groups to improve their development practices. To demonstrate how the data used to build the SciSPIF should be gathered and analyzed, the authors provide an example of how qualitative data can be collected from written reports and interview transcripts, analyzed, encoded, and categorized into themes (Mesh et al., 2014). The effectiveness of SciSPIF was not systematically evaluated. When more data is gathered and interpreted, the authors plan to develop theories to describe the application of software development practices to scientific software projects (Mesh et al., 2014). Making the framework flexible and accessible to developers of various skill levels is another research goal (Mesh et al., 2016).

The unique challenges associated with testing scientific software necessitate the need for specialized testing approaches. Unique test development methodologies and software development processes have been used for the quality assurance of scientific software. Researchers have examined the use and adaptation of existing software testing methods for scientific software, proposed the use of variability modeling for developing system test applications for scientific frameworks, and proposed methodologies for granular test creation and test case selection (Dubey & Wan, 2018; Hovy & Kunkel, 2016; Kanewala & Bieman, 2014; Remmel, 2014). Remmel (2014) demonstrated that software product line engineering (SPLE), especially variability modeling, can improve the quality assurance of scientific software code frameworks, which are standard codes that provide solutions to mathematical problems. The author used environments for automated testing and feature-oriented development to demonstrate the approach. The author also performed a case study that examined developers' attitudes towards the development of variability models and desk-checking and found they were both feasible and accepted by the developers. It should be noted that only one framework was tested.

Hovy and Kunkel (2016) created a tool for automating the creation of unit tests for subroutines of high-performance scientific Fortran programs. Dubey and Wan (2018) proposed test development frameworks for scientific software testing involving granular test creation and test case selection using scaffolding. The authors used two complex multiphysics programs as case studies to assess these techniques. They found that testing time was significantly reduced, and that testing should be performed at the beginning of the software design process to maximize software quality and productivity. However, they did not evaluate their approach's effectiveness for the detection of software defects.

Test-driven development (TDD) is a software development methodology that reduces risk by incorporating thorough and continuous testing into the development lifecycle. Researchers have proposed a fine-grained implementation of TDD and examined the benefits and drawbacks of using TDD for high-performance scientific software (Nanthaamornphong & Carver, 2018; Rilee & Clune, 2014). In their case study of an HPC software project, Nanthaamornphong and Carver (2018) found that TDD helped the developers think about software design and focus on small functional units of code; the refactoring process helped reduce code complexity and enhance performance. The drawbacks of using TDD were the challenges associated with writing useful tests and refactoring code. A limitation of the study is that only one team participated in the study; this team was small and inexperienced with TDD. Rilee and Clune (2014) proposed the employment of a fine-grained implementation of TDD to avoid concerns regarding the oracle problem and error accumulation caused by using finite-precision arithmetic. They discussed HPC software development requirements and how their proposed approach fills the gaps between the parallel Fortran Unit testing framework (pFUnit), fine-grained testing of scientific software, and an easy-to-use TDD framework. They recommended refactoring coarse-grained code and creating a set of fine-grained unit tests. Along with pFUnit, these capabilities are ideally provided to the user via an integrated development environment (IDE). A limitation of this study was that the IDE is still in development and that the authors' proposed approach was not tested.

Two other software development methodologies that have been studied for application to scientific software are document-driven design (DDD) and change-driven development (CDD). Document-driven design is a software development process intended to enhance the rigor of software design and documentation. Smith et al. (2016) assessed the impact of redeveloping

scientific software projects using DDD by interviewing five software projects' code owners. The code owners harbored positive feelings towards using a systematic approach to software development. Still, they felt that the documentation required by DDD takes too much time and effort to create. The researchers noted that this problem might have been caused by the time needed for ethics approval for the study. As a result, there was a lack of communication between the researchers and the study participants regarding the DDD artifacts' purpose and development. Change-driven development, which was proposed by Méndez and Tinetti (2017), is agile, incremental, change-driven, and iterative. The purpose of CDD is to provide a systematic approach for updating and parallelizing sequential Fortran programs that were written decades ago but are still in use today. Another objective of CDD is to address the observation that scientific software packages tend to evolve slowly compared to other software types. The authors conducted two case studies in which CDD was employed to parallelize one Fortran 77 program and transform another Fortran 77 program into a Fortran 90 program. They used these case studies to demonstrate that CDD is a viable approach for the modernization of older scientific applications that are still in use today.

Open-Source Software Development

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, tend to be 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 (Mockus et al., 2002; Nakakoji et al., 2002; 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; Dinh-Trong & Bieman, 2005; Geldenhuys, 2010; Mockus et

al., 2002). The findings of several studies in which small sample sizes of one to nine projects were used indicated that OSS projects follow the Pareto principle, which means that approximately 20% of the developers make about 80% of the contributions (Goeminne & Mens, 2011; Koch & Schneider, 2002; Robles et al., 2004). However, a more recent study that used a sample size of thousands of projects employed multiple heuristics for determining which contributors were core developers (Yamashita et al., 2015). This study controlled for the variables of project age, team size, and project size. The study findings showed that a substantial percentage of OSS projects on GitHub does not follow the Pareto principle. For most of the projects that do not comply with the Pareto principle, the percentage of core developers is less than 20%. The same study found 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.

Because the number of core contributors to an OSS project is often small, the departure of contributors poses a significant problem for the long-term maintenance and evolution of the project (Coelho & Valente, 2017; Hertel et al., 2003). Contributor turnover can reduce productivity because new contributors must spend additional time learning how the software works and gaining knowledge relevant to the project (Lin et al., 2017). One primary reason for the departure of new contributors from OSS projects is the non-acceptance of their first contributions by the core developers (Steinmacher et al., 2014; Steinmacher et al., 2015; Steinmacher et al., 2018). Steinmacher et al. (2018) found that about one-third of new contributors who failed to get their GitHub pull requests accepted disagreed with the decision. A significant percentage of these contributors stated that the nonacceptance was demotivating or prevented them from contributing to the project.

Furthermore, the study participants cited duplicated pull-requests, vision mismatch between the participant and the rest of the project team, lack of interest in the contribution from project integrators, the participant's lack of experience or commitment, and the contribution not being an optimal solution as the main reasons why their contribution was not accepted. The contribution's lack of tests and its introduction of side effects were reasonably common cited reasons for nonacceptance. The same study found that project integrators cited a lack of need or relevance of the contribution and not following project guidelines as their reasons for not accepting new contributors' pull requests. Similarly, Gousios et al. (2015) found that code style, code quality, documentation, adherence to project norms, and alignment with project objectives were essential factors in integrators' decision to accept a contribution. The most common reason given by the new contributors for contributing to an OSS project was to fix a defect.

Open-Source Software Quality Assurance

Developer turnover can affect software projects (Foucault et al., 2015; Mockus, 2010). Foucault et al. (2015) examined the relationship between OSS quality as measured by the density of bug-fixing commits and developer turnover. They looked at the departures of contributors from five widely used OSS projects and developers' movement between different modules of the projects. The researchers found that the OSS projects had high turnover rates despite their popularity and that internal turnover had little relationship with software quality. However, they also observed that higher external turnover correlated strongly with lower quality software modules due to a strongly negative relationship between newcomer activity and software quality. The study was limited by its small sample size and its use of only one software quality measure.

Similarly, Lu et al. (2016) noticed that casual contributors introduced more code quality issues to OSS projects than core contributors. The problems were more severe than those caused

by core contributors. In a study of a single software project, Mockus (2010) found that external turnover negatively affects software quality because of the loss of experience and expertise resulting from developers' departure. However, the author observed that newcomers did not affect software quality, perhaps because newcomers were not assigned to essential tasks.

The reuse of third-party software libraries can affect OSS projects (Constantinou et al., 2015; Zaimi et al., 2015). Many studies of OSS projects found that the reuse of third-party libraries is common in such projects (Constantinou et al., 2015; Haefliger et al., 2008; Schwittek & Eicker, 2013; Zaimi et al., 2015). Constantinou et al. (2015) observed that the object-oriented design quality of white box reused classes was higher than that of system classes. However, the design quality metrics of cohesion and coupling were negatively affected by third-party library reuse. The researchers also found that white box reused elements were seldom removed from the project once added. The developers did not appear to prioritize design quality in their selection of reusable software elements. The researchers suggested basing reuse decisions on quality criteria and not overusing third-party software elements, as this could result in increased resources needed for testing and maintenance.

Similarly, Zaimi et al. (2015) observed that third-party library updates and removal are uncommon and that many reuse decisions are not revisited. They also noted a relationship between design quality and third-party library update decisions, although it was not statistically significant due to the small sample size. The authors recommended that developers revisit their reuse decisions often and update reused libraries to their newest versions since the updated versions are expected to be of higher quality, more thoroughly tested, and provide more functionality.

There are many differences between the testing and 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 lifecycle 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 do not assure that a software product will be free of defects.

General Software Testing Challenges

One challenge that software testers sometimes face is difficulty reproducing software defects, leading to excessive time and effort spent debugging. Chaparro et al. (2019) addressed low-quality steps for reproducing errors in user-written bug reports. The authors proposed Euler, a method that analyzes the text of a bug report, identifies the steps to reproduce the defect, evaluates the quality of these steps, and sends feedback regarding the quality to the reporters. However, the proposed method was not tested in a real-world development environment. This study complemented existing approaches for the generation of test cases from bug reports (Fazzini et al., 2018; Karagöz & Sözer, 2017).

Automatic testing methods can reduce the resources needed for software testing, but they also have limitations (Saddler & Cohen, 2017; Toffola et al., 2017; Zheng et al., 2013). Toffola et al. (2017) noted that automatic test generation techniques frequently fail to provide relevant

input values. The authors proposed and evaluated TestMiner, a method for mining a collection of tests for appropriate input values for unit test generators. The proposed method was not evaluated in a real-world software development environment. Related work described test generators that can reason symbolically about expected input values, although this approach has low scalability (Zheng et al., 2013). Saddler and Cohen (2017) addressed the ineffectiveness of existing automated testing methods for GUI usability testing. The authors proposed EventFlowSlicer, a highly scalable process that enables the GUI tester to create all the test cases relevant to an objective. Swearngin et al. (2013) proposed a similar technique, but it has several limitations, including low scalability.

Other challenges faced by software testers involve human and organizational factors (Gonçalves et al., 2017; Majchrzak, 2010; Seth et al., 2014; Zhang, 2009). Gonçalves et al. (2017) identified organizational, operational, and cognitive factors and how they can affect the software testing process. The study results suggested that incompetent management, poor infrastructure, excessive overtime, monotony, low employee self-esteem, stress, and lack of training can negatively affect the software testing process. The authors also found that conflict between testers and developers can adversely affect the software testing process, a finding shared by Zhang (2009). The role of software testers within an organization is to validate and verify developers' software. Due to the nature of their position, testers may become involved in an interpersonal conflict with developers, which may lead to a reduction in the quality of the software product and decreased job satisfaction. Zhang (2009) examined the impact of interpersonal conflict and task conflict on software quality and job satisfaction. The study's findings suggested that interpersonal conflict has a significant effect on both job satisfaction and software quality and that task conflict affects software quality. Seth et al. (2014) found that

software testers tend to be inadequately involved in software projects during the planning phase, underestimating the resources required to test the software thoroughly. They also found that project managers are often unwilling to spend significant resources on testing.

Furthermore, top-down organizational structures reduce the awareness of the importance of software testing and quality assurance. Majchrzak (2010) studied the status quo, best practices, and known problems relating to various software development organizations' software development processes. The author developed a framework for categorizing software testing recommendations, focusing on organizations' ability to implement these recommendations successfully. The author's recommendations include defining clear and distinct roles for developers and testers, creating a testing department, performing requirements engineering, and measuring testing performance.

Metamorphic Testing Methodologies

The concept of MT was introduced over 20 years ago in a technical report (Chen et al., 1998). It 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 input changes in a specific way. The selection of MRs is a critical step when using MT. Multiple MRs can be identified for a given problem, and it is useful for software testers to know which ones are the most appropriate to use in each testing scenario. Cao et al. (2013) addressed which MRs have the best chance of revealing software defects. The researchers looked at the effectiveness of MRs at detecting software defects and the difference between the execution profiles of the test cases. They found that the higher the difference between the initial execution profile and the follow-up profile, the better the MR is at defect detection.

Researchers have proposed various approaches for identifying MRs (Ding et al., 2016; Kanewala, 2015; Lin et al., 2018; Zhou et al., 2016). Some approaches entail selecting MRs independently of each other and checking their results separately (Ding et al., 2016; Zhou et al., 2016). In contrast, Lin et al. (2018) introduced a hierarchy of MRs created incrementally. In this approach, the results of one MR are employed to generate additional MRs to locate software defects. The authors used a case study to demonstrate their testing method by applying it to software system integration. The process was tested on only one program, which limits the generalizability of the study results. Kanewala (2015) proposed a machine learning-based approach to predict MRs at the function level automatically. This approach models the MR identification problem as a binary classification problem and uses walk-based graph kernels to enhance identification accuracy. The researcher validated their approach using 100 mathematical functions that have only numerical input and output values.

Metamorphic Testing Frameworks

In addition to specialized methodologies, researchers have developed frameworks to help perform MT and identify useful MRs for the MT method (Chen et al., 2016; Murphy et al., 2009). One such framework is the METamorphic Relation Identification based on the Category-choice framework (METRIC) (Chen et al., 2016). The purpose of the METRIC is to identify a set of useful MRs that adequately covers the code. 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. A category is defined as an input variable or condition that affects the software during execution. A choice is defined as a partition of a category that includes possible values for the category. A critical variable in the METRIC framework is the diversity metric, which is defined as the difference between two test cases according to the number of

distinct choices and categories to which the test cases are connected. One limitation of METRIC is its reliance on tester expertise for MR identification.

Another framework for performing MT is the Amsterdam implementation framework (Murphy et al., 2009). The purpose of the Amsterdam framework is to address the laboriousness of employing MT for large input datasets or input that is not human-readable and the difficulty of creating useful test input for MT. The testing can be performed without altering the source code. The basis of this framework is the Automated Metamorphic System Testing approach. With this approach, the software to be tested is a black box.

Furthermore, the application's metamorphic properties, which are specified by the tester, should hold after the software completes its run if there are no defects in the code. A metamorphic property is defined as an input transformation, program execution, or output comparison specification. An input transformation specifies how to modify the data set. A program execution specification includes execution commands and runtime options. An output comparison is an expected output in relation to the original output. The program designer or algorithm creator can specify these properties, but detailed knowledge of the source code is not required to define them. With the Amsterdam framework, multiple invocations of the software are executed, and their outputs are compared to assess whether the properties hold. One weakness of the Amsterdam framework is its requirement that the tester specifies the MRs.

Acceptance of Metamorphic Testing

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). In a survey of MT literature, 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. The authors also found that 49% of the surveyed papers focused on applying MT to over a dozen different problem domains, including Web applications, computer graphics, modeling and simulation, embedded systems, encryption software, and financial software. This finding indicates a strong acceptance of MT for solving different types of problems. 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. However, to the best of the researcher's knowledge, no published studies have evaluated MT's acceptance among a population of software developers or testers.

Metamorphic Testing Effectiveness

The findings of many studies indicate that MT is highly effective at detecting defects in software even when conventional testing methods have failed (Cañizares et al., 2019; Ding et al., 2016; Kanewala & Chen, 2018; Zhou & Sun, 2019). For example, over 100 defects were found in popular C compilers and widely used protein function prediction tools using MT (Lidbury et al., 2015; Shahri et al., 2019). In a survey of MT literature, Segura et al. (2016) reported that 295 real defects in 36 programs were detected with the help of MT. Also, MT discovered new faults that had not been previously identified in the widely used Siemens test suite despite this suite having been the subject of testing research for 20 years (Rao et al., 2013).

Zhou and Sun (2019) employed MT successfully to find defects in software used in autonomous vehicles. They addressed the possibility that driverless cars' software systems could incorrectly interpret the data provided by a sensor, which could cause fatal accidents. They discussed the application of MT to these software systems. The authors argued that it is challenging to test these software systems using conventional software testing approaches because of the oracle problem and the need to carry out many tests to test the system thoroughly.

Other challenges include creating system specifications that can be used to check the behavior of autonomous vehicles, the complexity of verifying fuzz test results, and resource and time constraints. The MT method could be a cost-effective alternative to testing such systems. The authors detected severe defects in the light detection and ranging (LiDAR) obstacle perception module of autonomous vehicles using MT. They also discovered errors in the Google Maps service that calculates the optimal driving route between two locations.

Mutation testing has been employed to test MT's effectiveness (Cañizares et al., 2019; Ding et al., 2016; Kanewala & Chen, 2018). Ding et al. (2016) proposed a framework for developing MRs and tested their approach using ADDA, an open-source implementation of discrete dipole approximation. The test coverage was calculated to be 100% for statements and close to 100% for other conditions. The researchers used limited mutation testing to verify the effectiveness of the MT tests. The study's limitations are that it was only applied to one software program, which limits its generalizability, and that only one module of the ADDA software was subjected to mutation testing. Kanewala and Chen (2018) explained why MT is preferable for testers without a background in software development, such as the ease of learning and implementing MT. The authors used MT to evaluate three programs using mutation testing. They created mutants and recorded the number of mutants that the MRs were able to find. They found that 90% of mutants were detected. Although these are promising results, they showed that 10% of the mutants were not detected. Cañizares et al. (2019) proposed an expert system that employs MT and simulation to verify a memory system's correctness. They conducted a one-shot pre-experimental research study. The study's purpose was for the research team to evaluate the effectiveness of their proposed expert system in terms of defect detection. A total of 25,000 test cases were generated from a set of memory models and executed for the system with mutants

added. The defect detection effectiveness of 10 MRs for each memory management algorithm was recorded in terms of the percentage of test cases in which the added faults were not detected when applying the given MR. These results were combined to determine the overall effectiveness of the expert system. The researchers found that their testing approach was effective at finding defects in the memory system and that certain combinations of MRs increased the method's effectiveness. However, they also found that other MR combinations performed worse than the individual MRs, and that accurate MR design was crucial to ensuring the effectiveness of the proposed method.

Metamorphic Testing Challenges

When applied appropriately, the MT method can be highly effective at detecting software defects. However, there are challenges associated with using MT effectively and efficiently. One problem is the lack of a standard method for describing MRs, which can make them challenging to understand and use (Segura et al., 2017). There is high variability in the way MRs are described, leading to miscommunications among researchers and practitioners. 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, Segura et al. (2016) found that important information about the MRs was often left out of publications.

Another major challenge associated with MT usage is knowing how to define, select, and generate useful MRs and test cases, critical for performing MT effectively. Developing helpful MRs requires knowledge of the problem domain, the properties of helpful MRs, and the processes needed to construct them. Research relating to MR identification is preliminary, and consistent and reliable guidance for identifying and creating helpful MRs is not present in the

literature (Chen et al., 2018; Segura et al., 2016). Using all relevant MRs in testing can be inefficient, so prioritizing MRs based on their effectiveness is an important task. The automated generation of likely MRs is one potential solution to this challenge but approaches for MR generation have been largely limited to numerical programs. Similarly, 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; Chen et al., 2018; Wu et al., 2005).

In addition to the challenges mentioned above, the MT method can be challenging to implement because of a lack of MT tools. In a survey of 119 MT research papers, Segura et al. (2016) found that only two of them proposed a tool as a primary contribution to the literature. The applicability of these tools is limited to specific problem domains. The authors stated that publicly available and well-maintained MT tools do not exist and that this limitation hinders the widespread adoption of MT.

Synthesis of the Research Findings

Testing scientific software is critical in ensuring its precision and accuracy. One challenge involved in scientific software quality assurance is differences in software development practices between scientists and professional software developers (Ober & Ober, 2017). Scientist-developers tend to focus on the short-term goal of developing a program to meet immediate research needs at the expense of making the software robust and maintainable (Morris & Segal, 2009; Riesch et al., 2020). Software development methodologies have been developed to help create more robust and higher-quality software products, such as test-driven development (TDD) (Nanthaamornphong & Carver, 2018; Rilee & Clune, 2014). Software development practices and methodologies could affect a software tester's decision to use MT in their software development projects and how often to use it throughout the lifecycle of the software.

Testing a scientific software program poses a unique set of challenges, including the oracle problem, code complexity, and numerical approximations (Chen et al., 2018; Ding et al., 2016). The literature suggests that MT appears to be the most promising solution to testing without oracles (Chen et al., 2018; Ding et al., 2016; Kanewala & Bieman, 2014). Studies have shown MT to be useful for defect detection in a variety of problem domains, even in software that has already been thoroughly tested using other methods (Cañizares et al., 2019; Ding et al., 2016; Kanewala & Chen, 2018; Lidbury et al., 2015; Rao et al., 2013; Segura et al., 2016; Shahri et al., 2019; Zhou & Sun, 2019). For this reason, testers may decide to use MT for non-scientific software and software that does not have the oracle problem.

Although MT's basic concept is simple to understand and often easy to implement once MRs are selected, developing and prioritizing MRs can be difficult. Furthermore, the current literature does not provide consistent guidance for constructing MRs (Chen et al., 2018; Segura et al., 2016). Methodologies and frameworks have been developed to help MT users. Still, they have limitations such as relying on tester expertise, requiring that MRs be chosen by the tester, not being well-validated on a variety of software programs, or being designed only for a specific problem domain (Chen et al., 2016; Murphy et al., 2009). Also, there are very few MT tools, which means that MT users will usually develop their own (Segura et al., 2016). These challenges could reduce the perceived ease of use of the MT method. Although researcher interest and acceptance in MT is growing, MT researchers have stated that MT has not been widely adopted by practitioners (Chen et al., 2018; Segura et al., 2016). A study in which the research team assesses MT's acceptance and usage among a population of software developers has not been conducted to the best of this researcher’s knowledge. The effectiveness of MT and the challenges associated with using it may play a role in a user's decision to adopt and use the

MT method for testing their software. This study's objective was to provide information that helps researchers understand the relationships between these factors better.

In this study, the researcher employed the UTAUT as the theoretical framework. The UTAUT is one of the most popular models among researchers for predicting technology use and acceptance (Al-Mamary et al., 2015). The UTAUT has not been applied to software testing methods to the best of this researcher's knowledge, but it has been used for studying processes and practices for software development (Anderson, 2019; Guardado, 2012). An objective of formulating UTAUT was to help researchers and industry leaders understand the factors underlying technology acceptance better so that effective interventions for improving technology usage can be designed. This objective aligns with that of this study.

Often, researchers who use UTAUT as a theoretical framework employ quantitative methodology with a correlational or regression research design (Almaiah et al., 2019; Alrawashdeh et al., 2019; García et al., 2019; Gunawan, 2018; Xiang & Wu, 2018). A review of mobile payment service adoption studies that applied UTAUT found that survey questionnaires were used to collect data in 80% of the studies (Al-Saedi et al., 2019). For this reason, the researcher employed a quantitative methodology with a correlational research design, and data were collected using a survey instrument.

In this study, the researcher examined the relationships between the UTAUT constructs of performance expectancy (PE) and effort expectancy (EE) and MT's acceptance and usage. The PE construct represents the degree to which the technology user believes that the new technology will help improve their job performance or result in other favored outcomes (Compeau et al., 1999; Davis, 1989; Davis et al., 1992; Moore & Benbasat, 1991; Thompson et al., 1991). Venkatesh et al. (2003) found that PE was the strongest predictor of a user's intention to use a

new technology, which is why the study included this construct. The EE construct encompasses the user's perceived ease of use and the ease of learning and understanding the new technology (Davis, 1989; Moore & Benbasat, 1991; Thompson et al., 1991). Concerning PE, many publications suggest that MT is highly effective at detecting software defects, which could improve job-related outcomes for software developers and testers (Cañizares et al., 2019; Ding et al., 2016; Kanewala & Chen, 2018; Lidbury et al., 2015; Rao et al., 2013; Segura et al., 2016; Shahri et al., 2019; Zhou & Sun, 2019). However, the literature also notes the existence of challenges associated with using MT successfully. Because the researcher is interested in the role these challenges play in MT's acceptance and use, the EE construct was included in this study.

Critique of the Previous Research Methods

The number of studies that report the findings of real programs being tested using MT has been growing, indicating an increasing acceptance of MT for software testing (Chen et al., 2018; Segura et al., 2016). However, comprehensive surveys of MT studies suggest a lack of understanding of MT's effectiveness for detecting defects in large, complex programs and across a wide variety of application domains (Chen et al., 2018; Segura et al., 2016). While studies have been performed to assess MT's effectiveness, the majority of these studies have only tested a set of small to medium-sized programs or software from a single application domain (Chen et al., 2018; Liu et al., 2014). To effectively assess the effectiveness of MT, a body of literature that contains the findings of empirical studies relating to their fault detection effectiveness is required (Chen et al., 2018; Segura et al., 2016). Comprehensive MT studies that include large, diverse sets of programs and datasets are lacking in the literature. Regarding this study, this limitation may cause some software developers not to adopt MT because they are not convinced of its effectiveness or applicability to their software projects.

Researchers have proposed a variety of approaches for automating MT and hence making MT easier to use for software developers and testers. However, the scope of many studies that examined the effectiveness of MT methodologies and frameworks was limited to a small sample of programs or datasets or a single problem domain. For example, Ding et al. (2016) proposed a framework for developing MRs, but they validated it using only one scientific software program. Lin et al. (2018) introduced a hierarchy of MRs created incrementally and applied their method to program integration, but they tested it using only one dataset. Similarly, Kanewala and Chen (2018) used MT to evaluate only three programs; Kanewala (2015) proposed an approach for the automatic prediction of MRs at the function level and evaluated it on a set of mathematical functions that had only numerical input and output. Cañizares et al. (2019) proposed an expert system that employs MT to verify a memory system's correctness. One strength of this study is that the authors used a large sample of 25,000 test cases. However, the effectiveness of the system was examined for only four memory management algorithms. This weakness limits the generalizability of the study results to other programs and other types of software. This limitation implies that software developers may decide not to adopt MT due to difficulty locating an existing approach appropriate for use with their software projects. Also, the participants may find MT overly challenging to learn and use effectively.

The theoretical framework that was used in this study, UTAUT, has many strengths and some weaknesses. One significant advantage of UTAUT is that it has been shown to outperform eight older models, including the widely used technology acceptance model (TAM), in explaining the variance in employee intentions to use new technology (Davis, 1986; Davis et al., 1989; Venkatesh et al., 2003). Venkatesh et al. (2003) developed UTAUT using a rigorous methodology that involved a within-subjects longitudinal comparison, empirical validation, and

cross-validation. Also, reviews of UTAUT studies found that the theory's quality, robustness, and validity is high (Venkatesh et al., 2016; Walldén et al., 2016). On the other hand, the UTAUT has been criticized for weaknesses associated with its constructs of facilitating conditions (FC) and behavioral intention (BI). These constructs exclude external behavioral determinants that can change over time, such as user ability and environmental factors (Venkatesh et al., 2008). In their meta-analysis, Dwivedi et al. (2017) suggested that attitude should be included as a mediating construct in UTAUT. However, these weaknesses are not highly relevant to the objectives of the study. The study was cross-sectional, and its main aim was to examine the relationships between the constructs of PE and EE and the acceptance and usage of MT. The researcher selected UTAUT as the theoretical basis for the study because it has been widely used and validated and has been shown to outperform other technology acceptance models.

Summary

Quality assurance (QA) is a critical component of the software development lifecycle. In contrast to closed-source projects, open-source software projects tend to use informal and unstructured QA and testing practices. Testing and QA are essential because the presence of undetected defects in software can have severe consequences. For scientific software used for research purposes, errors can lead to inaccurate research results and retractions of publications based on those results. Testing scientific software presents a unique set of challenges, such as the oracle problem and cultural differences between scientist-developers and professional software developers. Software development methodologies and testing methods have been developed to address these challenges. The MT method is a testing method that has often been employed to handle the oracle problem and detect defects that conventional testing methods cannot find.

However, studies on MT effectiveness have been limited to small or medium programs or a specific domain.

Furthermore, using MT effectively can be difficult due to a lack of tools and unclear guidance for selecting appropriate MRs, which is critical in the MT method. The perceived effectiveness of MT and the challenges associated with using this method may play a role in a software developer's decision to accept and use it. The QA practices employed by the developer's team may also affect their decision to use MT.

The UTAUT is a widely used theoretical framework for studying the acceptance and usage of technology. This theory has been employed in research studies to examine the acceptance of various technologies, including software development and management processes and practices. Still, it has not been applied to software testing methods. Although the literature indicates that researcher interest in MT has been increasing over time, no published studies have examined MT's acceptance and usage among practitioners to the best of the researcher's knowledge. This study addressed these gaps in the literature by applying UTAUT to explore the relationships between UTAUT constructs and MT's use and acceptance.

Chapter 3: Research Method

Quality assurance is a critical component of the software development life cycle. Testing software is essential to ensure the precision and accuracy of its output. One problem commonly encountered when testing scientific software is the oracle problem: the lack of a test oracle for output comparison (Chen et al., 2018). Detecting all defects in a software product is another major challenge. Metamorphic testing (MT) addresses the oracle problem and can improve defect detection. However, there are challenges associated with MT's use, such as difficulties in selecting good metamorphic relations (MRs) and developing effective test cases. These issues could affect a software developer's decision regarding the use of MT to test their programs.

The problem that was addressed by this study was the inadequacy of software testing methods to detect all software defects, which could be mitigated by the use of the MT method (Lidbury et al., 2015; Rao et al., 2013). The MT method has been able to find defects in widely used and 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; Shahri et al., 2019). Another example of the problem is the detection of new defects that had not been previously found in the widely used Siemens test suite despite this suite being the subject of testing research for 20 years (Rao et al., 2013). The problem is significant because defects in software can have severe consequences for users. These consequences include poor software performance, reduced precision or accuracy of software output, and retractions of research publications for which defective software was used (Kanewala & Chen, 2018). 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). Detecting software defects can be challenging due to various

factors such as the oracle problem, code complexity, and constraints on time and resources (Chen et al., 2018; Ding et al., 2016; Zhou & Sun, 2019).

The purpose of this quantitative correlational study was to address the problem of the inadequacy of software testing methods for detecting all defects by enabling the researcher to examine relationships between MT's use and acceptance among open-source developers and the constructs of performance expectancy (PE) and effort expectancy (EE). Another objective of the study was for the researcher to understand how the variables of age, gender, and experience moderate these relationships. The study purpose aligns with the research problem because the MT method has been found to detect software defects that other testing methods failed at finding (Lidbury et al., 2015; Rao et al., 2013; Shahri et al., 2019). An improved understanding of the factors associated with the acceptance and use of MT could help develop interventions to increase MT usage and hence reduce the number of software defects. Furthermore, this study's results are useful for assessing the applicability of the unified theory of acceptance and use of technology (UTAUT) to the acceptance and use of software testing methods.

For this study, the researcher employed a quantitative methodology with a correlational research design. In this chapter, the researcher discusses the research methodology's appropriateness and design, describes the sample and population of interest, outlines the sampling procedure, describes the research instrument used to collect data, and provides the variables' operational definitions. Then, the assumptions, limitations, and delimitations of the study are listed. Finally, ethical considerations for the study are discussed.

Research Methodology and Design

In this study, the author applied a quantitative methodology, which is suitable for validating existing theories, testing hypotheses, and measuring known constructs (Choy, 2014;

Creswell, 2014). Quantitative research methods align with the study's purpose, including applying a current theoretical framework to software testing methods and measuring known variables. Quantitative research is ideal for answering the study research questions, which pertain to relationships among known variables and theoretical constructs. Furthermore, the research questions were addressed using hypothesis testing, which falls under quantitative methodology.

A correlational research design was used to carry out the study. A correlational design was the most appropriate research design for this study because the purpose of the study was to examine relationships among variables (Creswell, 2014). These variables are the acceptance and usage of MT and two UTAUT constructs, namely performance expectancy (PE) and effort expectancy (EE) (Venkatesh et al., 2003). Furthermore, the research questions for the study relate to these relationships. The study design was appropriate for addressing the inadequacy of conventional software testing methods for defect detection by examining the correlation between MT's use and acceptance and factors that could significantly impact the MT method's use and acceptance.

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 in the form of UTAUT constructs (Creswell, 2014; Venkatesh et al., 2003). Due to its cost-effectiveness and high accessibility to the target population, an online survey instrument was used to collect the sample data.

Other research designs and methodologies were less suitable for this study. A descriptive design was not the most appropriate research design for this study since the study objective was not limited to gathering information about the population (Shields & Rangarajan, 2013).

Experimental and quasi-experimental designs were not appropriate for this study because the study did not involve administering a treatment or intervention and observing its effects (Creswell, 2014). A comparative design was not suitable since comparisons among multiple groups were not performed (Salkind, 2010). Longitudinal designs were not ideal for this study because the researcher gathered data at only one point in time (Salkind, 2010). Finally, the researcher preferred quantitative research methodology to qualitative methodology because the study's objective was to measure a set of constructs and analyze the relationships among them, which is more suited to quantitative methodology (Creswell, 2014).

Population and Sample

The study's target population consisted of GitHub users who have contributed to any of the public GitHub repositories contained in the “Software in science” collection on the GitHub Web site (GitHub, Inc., 2020a). This collection consists of repositories that host scientific software programs for research purposes and software libraries for mathematics and algorithms, a command shell, a package manager, a client for accessing scientific journal articles, and data processing tools. Also, the respondents were required to be at least 18 years of age and to speak English fluently for inclusion in the study. The size of the target population was approximately 4,500 contributors. The overall population consists of all contributors to open-source software projects hosted on GitHub. The overall population size can be estimated via the number of GitHub users, which is approximately 50 million, although not all GitHub user accounts are active (GitHub, Inc., 2020b). Based on the study problem and purpose, it was appropriate to specify the target population as the contributors to a collection of open-source projects containing scientific software and software in other problem domains.

Stratified sampling was used to obtain the study sample. In this sampling method, the target population is divided into strata, and study participants are recruited from each stratum (Oxford University Press, 2020). The sample consisted of respondents from a set of 700 potential participants who have contributed to the repositories in the GitHub "Software in science" collection. The potential respondents had to have an e-mail address either listed on their GitHub user page or obtainable via the GitHub API. The potential respondents were selected from the 100 most active contributors to each repository in the collection. This sampling method ensured that all software projects in the collection were represented in the set of potential participants. The sample was appropriate because the study problem applies to a variety of software types. Furthermore, the study purpose and research questions pertain to open-source software developers in general rather than only those who contribute to one specific project or type of project.

A sample size of 41 participants was obtained for this study. 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). It was determined that the minimum correlation coefficient that the study could detect was 0.42. In this power analysis, the sample size was 41, the statistical power was 0.8, and the significance level was 0.05. Since the researcher employed a correlational design with a single sample, one group was included in the analysis.

The study participants were recruited by sending each of the potential participants a recruitment e-mail using the e-mail address listed on their GitHub user page. This e-mail contained information about the study and researcher, participant eligibility criteria, and a link to the survey. Based on a published study in which GitHub contributors were surveyed, the expected response rate was approximately 24% (Kalliamvakou et al., 2016). Given this response

rate, the recruitment e-mail was sent to 700 potential participants to ensure an adequate sample size.

Instrumentation

The survey instrument published by Venkatesh et al. (2003), with minor rewording to clarify that the questions pertain to the MT method, was used to measure the relevant constructs and variables. A sample research instrument is provided in Appendix A. The constructs of performance expectancy (PE), effort expectancy (EE), and behavioral intention (BI), which is a measure of technology acceptance, were measured using this instrument. The frequency of use (FoU) as well as the UTAUT moderating variables of age, gender, and experience were also measured in the survey (Venkatesh et al., 2003; Venkatesh et al., 2012). The research instrument aligns with the study's quantitative methodology in that all constructs are measured using numerical Likert scales.

Furthermore, the variables of age and experience were measured numerically. Gender is at the nominal level of measurement; gender items were assigned numerical values for data analysis. The UTAUT moderating variable of voluntariness of use was not included because previous research suggests that this variable does not moderate the relationships between the PE, the EE, and the acceptance and usage of technology (Venkatesh et al., 2003).

The developers of UTAUT created and employed the survey instrument in their field studies to collect data from employees on the UTAUT constructs and moderating variables (Venkatesh et al., 2003). These field studies were conducted at six organizations in various industries in which participants were introduced to new technologies in their work environment. These field studies aimed to validate UTAUT and compare its performance to other technology acceptance models.

Venkatesh et al. (2003) 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. They found that all internal consistency reliability values were higher than 0.70. 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.

Operational Definitions of Variables

The theoretical constructs and variables outlined in this section were measured in this study. An independent variable explains the outcome of one or more dependent variables (Creswell, 2014). A dependent variable depends on one or more independent variables. A moderating variable affects the relationship between an independent variable and a dependent variable.

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 outcomes (Compeau et al., 1999; Davis, 1989; Davis et al., 1992; Moore & Benbasat, 1991; Thompson et al., 1991; Venkatesh et al., 2003). The PE was used as an independent variable in the Spearman correlation tests.

The PE construct was measured by summing the four survey items adapted from the instrument published by Venkatesh et al. (2003). These items are as follows: “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,” and “If I use the method, I will increase my

chances of getting a raise." Like the instrument used by 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]. On this scale, a one represents "Strongly disagree," four means "Neither agree nor disagree," and a seven represents "Strongly agree." The total possible score for the PE construct was an integer in the range [4-28].

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 (Davis, 1989; Moore & Benbasat, 1991; Thompson et al., 1991). The EE was used as an independent variable in Spearman correlation tests. The EE construct was measured by summing the four survey items adapted from the instrument published by Venkatesh et al. (2003). These items are as follows: "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," and "Learning to use the method is easy for me." Like the instrument used by 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]. On this scale, a one represents "Strongly disagree," four means "Neither agree nor disagree," and a seven represents "Strongly agree." The total possible score for the EE construct was an integer in the range [4-28].

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 specified number of months (Venkatesh et al., 2003). The BI is a measure of user acceptance of new technology. This construct was used as a dependent variable in the Spearman correlation tests. The BI construct was measured by summing the

measurements of three survey items adapted from the instrument published by Venkatesh et al. (2003). These items are as follows: “I intend to use the method in the next 12 months,” “I predict I would use the method in the next 12 months,” and “I plan to use the method in the next 12 months.” Like the instrument used by 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]. On this scale, a one represents “Strongly disagree,” four means “Neither agree nor disagree,” and a seven represents “Strongly agree.” The total possible score for the BI construct was an integer in the range [3-21].

Frequency of Use (FoU)

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

The FoU was measured using a single survey item with a seven-point Likert scale. The item is, “Please choose your usage frequency for the metamorphic testing method.” Like the instrument used by Venkatesh et al. (2012), each item was measured using an ordinal scale, namely a seven-point Likert scale with possible values being integers in the range [1-7]. On this scale, a one represents “Never,” a four represents “Sometimes, in about 50% of the chances when I could have,” and a seven represents “Every time” (Vagias, 2006).

Age

The user's age is a moderating variable that affects UTAUT constructs (Venkatesh et al., 2003). The user's age was used as a moderating variable in the Spearman correlation tests. Age is at the ratio level of measurement. This variable was operationalized as a continuous variable that measures the user's age in years (Morris & Venkatesh, 2000). A single survey item was used to collect age data from participants. This item was, "Please enter your age in years." Since participants in the study had to be at least 18 years of age, the possible score for the age variable was an integer in the range [18-120]. Respondents were given the option of "Prefer not to answer."

Gender

The user's gender is a moderating variable that affects UTAUT constructs (Venkatesh et al., 2003). The user's gender was used as a moderating variable in the Spearman correlation tests. Gender is at the nominal level of measurement. A single multiple-choice survey item was used to collect gender data from participants. This item is "Please select your gender." The possible choices were "Male," "Female," "Other," and "Prefer not to answer."

Experience

User experience with the new technology is a moderating variable that affects UTAUT constructs (Venkatesh et al., 2003). The user's experience was used as a moderating variable in the Spearman correlation tests. The relevant survey item is, "Please choose your level of experience with the metamorphic testing method." Like the instrument used by Kim et al. (2005), each item was measured using an ordinal scale, namely a five-point Likert scale with possible values being integers in the range [1-5]. On this scale, a one represents "less than six months," a three represents "one to three years," and a five represents "seven years or more."

Study Procedures

In this cross-sectional study, the researcher employed an online survey to collect all relevant data from the study participants. The survey instrument is a slightly modified version of the instrument designed by Venkatesh et al. (2003). The instrument designers verified the reliability and validity of the instrument for collecting data on the UTAUT constructs.

First, the researcher created a list of potential study participants by collecting the e-mail addresses from the 100 most active contributors to each repository in the GitHub "Software in science" collection who had publicly listed e-mail addresses. The 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 collection and the number of potential participants recruited from among the contributors to each project is presented in Table 1. This method is like the population selection method used by Kalliamvakou et al. (2016). 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 the researcher sent 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, the researcher selected 700 contributors for recruitment based on GitHub users' response rate obtained by Kalliamvakou et al. (2016). The researcher stored the list of e-mail addresses for the final set of 700 potential participants in a spreadsheet file.

Table 1

Number of Potential Participants Recruited from Each Repository

Name of Repository	Number of Potential Participants
simbody/simbody	17
cms-sw/cmssw	77
ComputationalRadiationPhysics/picongpu	13
psas/av3-fc	3
astropy/astropy	88
dfm/emcee	21
cyverse/atmosphere	11
dib-lab/khmer	32
sympy/sympy	83
spack/spack	91
ipython/ipython	84
ropensci/rplos	4
sagemath/sage	62
gap-system/gap	19
Singular/Singular	22
fredrik-johansson/arb	11
broadinstitute/picard	46
markusschanta/awesome-jupyter	16

Next, the researcher 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 (Dembsey, 2020). The respondent was invited to click on the 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 a

Qualtrics survey. The first page of this survey provided a 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 and UTAUT constructs (Venkatesh et al., 2003). The first, second, and third pages of the survey provided the items that measured the PE, the EE, and the BI, respectively (Anderson, 2019). The fourth page contained items that measured the FoU and experience. The items that measured the participant's age and gender were on the fifth page. A progress bar at the bottom of the page displayed the percentage of the survey completed. When the participant finished the survey, a new page thanked them for their time and participation.

The data collection period was divided into two phases, each of which was two weeks long. The researcher selected this time period based on research that found over 90% of survey responses occur within two weeks (Phillips et al., 2016). The researcher 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 one week. Since the desired sample size was not reached by the end of the first two-week collection phase, the second set of 350 potential participants was recruited from the GitHub contributors to the "Software in science" collection.

Data Analysis

The data was collected using an online Qualtrics survey (see Appendix A for an example survey) and imported into IBM SPSS Statistics Version 26 for analysis. The study hypotheses were tested by performing Spearman correlation analyses to measure the relationships among variables. The researcher began the analysis by creating scatterplots for the following variable

pairs and visually inspecting them for monotonicity: BI vs. PE, BI vs. EE, FoU vs. PE, and FoU vs. EE (Lund Research Ltd, 2018c). The PE and EE were the independent variables, and the BI and FoU were the dependent variables in the data analysis. All of the relationships between variable pairs were monotonic.

The researcher performed Spearman correlation analyses using IBM SPSS Statistics 26. The sample size, the Spearman rank-order correlation coefficient, the degrees of freedom, and the p-value were obtained (Corder & Foreman, 2014). The p-value indicates the statistical significance of the correlation. The hypotheses were tested by determining whether statistically significant relationships exist between the BI and PE, the BI and EE, the FoU and PE, and the FoU and EE. Also, the effects of the moderating variables, namely age, gender, and experience, on the relationships between the independent and dependent variables were tested using moderated multiple regression. In moderated regression, interaction terms between a moderating variable and an independent variable are added to the regression equation to test whether any of the interactions are statistically significant (Lund Research Ltd, 2018b).

Finally, the researcher performed a wave analysis to check for nonresponse bias. The study 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. In wave analysis, which is a widely used method for assessing nonresponse bias, the final wave's responses are compared to the first wave's responses (Phillips et al., 2016). The late respondents are used as proxies for nonrespondents, and the early respondents are treated as true respondents. The nonresponse bias is obtained by multiplying the proportion of nonrespondents by the difference in the two sets of responses.

Assumptions

The researcher assumed that the research instrument is valid and reliable because the reliability and validity of its measures have been verified by Venkatesh et al. (2003). The study participants were assumed to respond to the survey items honestly and accurately. The possibility of incorrect responses was mitigated by using a validated survey instrument and providing the participants with clear instructions regarding taking the survey. The participants were assumed to be familiar with MT's basic concept since a definition of MT was provided in the recruitment e-mail. Finally, all participants were believed to be eligible to participate in the study, as the eligibility criteria were clearly stated in the recruitment e-mail.

Limitations

One limitation of the study is that not all potential participants recruited for the study participated, resulting in nonresponse bias. This limitation was mitigated by performing wave analysis to estimate the nonresponse bias. Another limitation was that the study data was subjective because it only came from self-reported survey answers, which means the researcher could not verify the responses' accuracy. Furthermore, the study's correlational research design means that causal relationships between constructs and variables cannot be inferred from the study findings. Also, the study's quantitative methodology limited the collected data to the specified variables and constructs of interest. Future studies could incorporate qualitative methods to explore other possible factors about the study problem and purpose. On a related note, the construct of BI only measures the participants' intention to use new technology within a specific period. It does not consider participants who plan to use it in the more distant future.

Another study limitation was the sampling method, in which only a subset of contributors from a subset of all OSS projects was recruited for the study. This limitation implies that the

study findings may not be generalizable to all OSS developers. This limitation was mitigated by sampling from an OSS project collection that includes 18 diverse software projects. Future research could further reduce this limitation by sampling from more software projects across various problem domains.

Delimitations

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 et al., 2016; Walldén et al., 2016). Thus, the variables and constructs included in this study were drawn from the UTAUT.

The researcher limited the independent variables of interest to the UTAUT constructs of PE and EE. The PE was included because Venkatesh et al. (2003) found that it was the BI's strongest predictor. The BI construct operationalized the acceptance of MT in this study. The literature notes the existence of challenges associated with using MT effectively (Chen et al., 2018; Segura et al., 2016; Segura et al., 2017). The EE construct, which represents the ease of using new technology, was included because the researcher was interested in the role these challenges play in MT's acceptance. The dependent variables of interest were limited to the BI and FoU, as these two variables represent MT's acceptance and use, respectively. The variables selected for inclusion in this study align with the research questions.

The researcher limited the study sample to English-speaking participants at least 18 years of age who have contributed to an OSS project in the “Software in science” collection on the GitHub Web site. This collection includes software used for scientific research and other projects that are useful to scientists. The researcher selected this delimitation because the study purpose

and research questions are about examining MT's acceptance and usage, which is especially helpful in handling the oracle problem (Chen et al., 2018). Furthermore, the study problem is the inadequacy of conventional software testing methods for defect detection, and the oracle problem is a common obstacle to defect detection in scientific software (Kanewala & Bieman, 2014). The researcher also opted 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 (Mockus et al., 2002; Nakakoji et al., 2002; Yamashita et al., 2015). Therefore, core contributors are more likely to make decisions regarding software testing.

Ethical Assurances

The study received approval from Northcentral University's Institutional Review Board (IRB) before data collection (see Appendix B for the approval letter). The informed consent guideline was followed by providing a letter of informed consent on the first page of the online research instrument (see Appendix A). The researcher has obtained permission from the publisher to use the research instrument in this dissertation (see Appendix C for the license). The informed consent letter contained information regarding the study and researcher, participant eligibility criteria, participant activities and time commitment required, risks and possible benefits, privacy, data protection, how the researcher will use the results, and contact information. The letter also stated that answering demographic questions is optional and that participation in the study is voluntary. Furthermore, a recruitment e-mail was sent to potential participants that clearly stated that research participation is being solicited and that participation in the research study is voluntary. This e-mail also contained information about the study.

Possible risks associated with this study include minor physical discomforts related to taking an online survey. The participant could decrease the impact of these risks by ending participation at any time by closing their Web browser. The informed consent letter contained instructions on how to end involvement in the study. The researcher did not have undue influence over the study participants, as no personal or professional relationship existed between the researcher and any potential participants. The researcher did not offer any rewards for participation in the study. Participants had to be at least 18 years of age.

The privacy of participants and the protection of their data was ensured by using a Qualtrics survey to collect data. According to Qualtrics' security statement, Qualtrics' servers are protected by firewalls and are tested for vulnerabilities regularly (Qualtrics, 2020). Access to Qualtrics' systems is monitored and restricted to individuals with a need-to-know. Qualtrics anonymizes all survey responses and uses Transport Layer Security (TLS) encryption for data transmission. Independently audited data centers host Qualtrics services. Also, Qualtrics has ISO 27001 certification and is FedRamp authorized (Qualtrics, 2020).

To further protect participant privacy, the e-mail addresses of potential participants were obtained only from public Web pages and stored in a spreadsheet file on a password-protected laptop computer accessible only to the researcher. The researcher erased this file from the computer upon study closure. The survey instrument did not collect any personally identifiable information. Two survey items gathered potentially sensitive demographic information on age and gender. Still, the respondent could choose "Prefer not to answer" for these items and continue the survey. Research data are stored on a password-protected laptop computer accessible only to the researcher for three years from the date of study closure. Then, the researcher will erase it from the computer. Participants will not be identified in any publications

or presentations that present the study findings. Research data will be grouped in any publications or presentations.

Summary

In this study, the researcher used a quantitative methodology with a correlational research design and employed UTAUT as its guiding theoretical framework. The researcher addressed the inadequacy of conventional software testing methods for defect detection by examining the MT method's use and acceptance among a population of GitHub contributors. The data were collected using an online survey. The primary variables of interest were the PE, the EE, the BI, and the FoU. The researcher performed Spearman correlation tests to measure the relationships between variable pairs. The moderating variables of age, gender, and experience were also included in this study. The researcher performed moderator analyses to assess each moderator's effect on the relationships between the variable pairs.

Chapter 4: Findings

The purpose of this quantitative correlational study was for the researcher to examine the relationships between the UTAUT constructs of PE and EE and the use and acceptance of the metamorphic testing (MT) method among open-source software developers. Another objective of the study was to understand how the variables of age, gender, and experience moderate these relationships. An improved understanding of the factors associated with MT's acceptance and use could help researchers and practitioners design interventions to increase MT usage and reduce the number of defects in software. Furthermore, this study's findings provided information about the applicability of UTAUT to software testing methods.

This chapter begins with a discussion of the validity and reliability of the research instrument used in this study. This discussion is followed by a list of considerations for the validity of the data obtained. These considerations include nonresponse bias and the extent to which the data meet the statistical tests' assumptions. Next, the descriptive data for the study participants and the results of the statistical analyses are presented. Then, the study findings are evaluated in a brief discussion. The chapter ends with a summary of the main points raised in the chapter.

Validity and Reliability of the Data

The survey instrument used in this study was created and employed by the developers of UTAUT to collect data relating to the UTAUT constructs and moderating variables in their field studies (Venkatesh et al., 2003). Venkatesh et al. (2003) 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. They found that all internal consistency reliability values were

higher than 0.70. 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.

Another factor that could affect the interpretation of the study findings is the presence of nonresponse bias. The researcher performed a wave analysis to estimate the nonresponse bias for the variables of interest. In wave analysis, a widely used method for assessing nonresponse bias, the study participants are divided into groups, or "waves," according to when they completed the survey (Phillips et al., 2016). The first wave's respondents are considered true respondents, while the last wave's respondents are considered the proxy nonrespondents. 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 (Phillips et al., 2016). In this study, the respondents who completed the survey before the reminder e-mail was sent comprise the first wave. The last wave consists of respondents who completed the survey after the reminder e-mail was sent. The nonresponse bias for each variable and their possible values are presented in Table 2.

Table 2

Nonresponse Bias and Range of Possible Values for the Variables of Interest

Variable	Nonresponse Bias[a]	Range of Possible Values
Performance Expectancy	0.15	[4-28]
Effort Expectancy	1.8	[4-28]
Behavioral Intention	2.4	[3-21]
Frequency of Use	0.93	[1-7]

[a]The positive values of the nonresponse biases indicate that the mean value for the first wave of respondents was greater than the mean value for the second wave.

The statistical test used to answer the study research questions was Spearman's rank-order correlation test. Three assumptions must be made to perform this test (Lund Research Ltd, 2018d). They are listed as follows:

1. The variables must be measured on the ordinal, ratio, or interval scale.
2. The variables must represent paired observations.
3. There must be a monotonic relationship between the two variables.

The data meet the first assumption since the PE, EE, BI, and frequency of use (FoU) were all measured on the ordinal scale. The data meet the second assumption because the variables of interest represent paired observations for each participant. Visual inspection of scatterplots for each variable pair shows that the data meet the third assumption of monotonicity.

In addition to the relationships between the variables of interest, the researcher is interested in how the moderators, namely age, gender, and experience, affect these relationships. The researcher assessed these three variables' moderating effect by performing moderator analyses using moderated multiple regression (MMR). In MMR, an interaction term between a

moderating variable and an independent variable is added to the regression equation to test whether the interaction is statistically significant (Lund Research Ltd, 2018a). Six assumptions must be made before performing MMR. These assumptions are listed as follows:

1. The variables must be continuous.
2. The variables must be linearly related to each other.
3. Multicollinearity should not be present.
4. The data should not contain significant outliers.
5. The data should show homoscedasticity.
6. The residuals should be normally distributed.

Because the variables of interest are ordinal variables with at least five categories, the researcher treated them as continuous variables per studies that suggest doing so is permissible for statistical analysis (Taylor et al., 2006; Zumbo & Zimmerman, 1993). Therefore, the first assumption was met. The researcher verified the linearity of the variables by visual inspection of scatterplots. The independent variables were mean centered before analysis to reduce multicollinearity (Lund Research Ltd, 2018a). The data met the assumption of not showing multicollinearity, as evidenced by no tolerance values less than 0.1 (Lund Research Ltd, 2018a). There were no significant outliers for most analyses, as evidenced by no studentized deleted residuals greater than ±3 standard deviations (Cohen et al., 2003). The data met the assumption of homoscedasticity. The researcher verified this assumption by visual inspection of the studentized residuals plotted against the predicted values (Lund Research Ltd, 2018a). For most analyses, the residuals were normally distributed, as assessed by Shapiro-Wilk's test ($p > .05$) (Lund Research Ltd, 2018a). The researcher handled any violations of these assumptions by transforming the dependent variable (Lund Research Ltd, 2018a).

Results

In this quantitative correlational study, the researcher collected data for the variables of interest via survey. The researcher performed data analysis to assess the correlation between these four variable pairs: the PE and BI, the EE and BI, the PE and FoU, and the EE and FoU. The researcher performed moderator analysis to evaluate the effect of age, gender, and experience on the relationships between these variable pairs.

The researcher recruited study participants from the contributors to the open-source software projects in the GitHub "Software in science" collection. The participants had to be at least 18 years of age and speak English fluently to be eligible for this study. The study recruitment letter, containing a link to the survey instrument, was e-mailed to a list of 700 potential participants. Twelve e-mails bounced, leaving 688 potential participants who successfully received the recruitment e-mail. Forty-one of these 688 potential participants completed the survey, resulting in a response rate of 6.0%. The demographic information about the participants is presented in Table 3. The sample is highly unevenly distributed across the gender categories, with 90.2% of participants identifying as male and only 4.9% identifying as female. The participants are distributed across age groups ranging from 21-24 to 50-54, with the largest number of participants (26.8%) belonging to the 35-39 group. Most participants (58.54%) indicated they have less than six months of experience with MT.

Table 3

Gender, Age, and Experience of Study Participants

Variable	Classification	Frequency (N=41)	%
Gender	Male	37	90.2
	Female	2	4.9
	Other	-	0.0
	No response	2	4.9
Age	21-24	1	2.44
	25-29	3	7.32
	30-34	8	19.5
	35-39	11	26.8
	40-44	5	12.2
	45-49	3	7.32
	50-54	7	17.1
	No response	3	7.32
Experience	Less than 6 months	24	58.54
	1 to 3 years	6	14.63
	4 to 6 years	4	9.76
	7 years or more	6	14.63
	No response	1	2.44

Note. No participants identified their gender as "Other."

Descriptive statistics for the PE, EE, BI, and FoU are presented in Table 4. 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 center of the FoU data is at the lower end of the range.

Table 4

Descriptive Statistics for the Variables of Interest

Variable	Mean	Median	Standard Deviation	Standard Error of the Mean
PE	16.54	16.00	5.13	0.80
EE	18.83	20.00	5.16	0.81
BI	11.32	12.00	6.47	1.01
FoU	2.44	2.00	1.75	0.27

Table 5 presents the survey items that correspond to the PE construct and their possible scores. The table contains the percentage of the sample that indicated each score. For the survey item, “I would find the method useful in my job,” most responses 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 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%).

Table 5

Percentages of Responses for Performance Expectancy (PE) Survey Items

Item	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Agree	Strongly Agree
I would find the method useful in my job.	4.88	0.00	9.76	21.95	19.51	24.39	19.51
Using the method enables me to accomplish tasks more quickly.	4.88	4.88	7.32	51.22	12.20	7.32	12.20
Using the method increases my productivity.	4.88	0.00	12.20	46.34	12.20	14.63	9.76
If I use the method, I will increase my chances of getting a raise.	36.59	12.20	7.32	36.59	0.00	4.88	2.44

Note. Survey items were adapted from Venkatesh et al. (2003).

Table 6 presents the survey items that correspond to the EE construct and their possible scores. The table contains the percentage of the sample that indicated each score. For the survey item “The method is clear and understandable,” most responses belong to the three scores of “Neither agree nor disagree” (24.39%), “Somewhat agree” (19.51%), and “Agree” (24.39%). 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%).

Table 6

Percentages of Responses for Effort Expectancy (EE) Survey Items

Item	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Agree	Strongly Agree
The method is clear and understandable.	7.32	4.88	12.20	24.39	19.51	24.39	7.32
It would be easy for me to become skillful at using the method.	2.44	2.44	4.88	31.71	19.51	31.71	7.32
I would find the method easy to use.	2.44	2.44	9.76	31.71	24.39	24.39	4.88
Learning to use the method is easy for me.	2.44	2.44	7.32	29.27	19.51	34.15	4.88

Note. Survey items were adapted from Venkatesh et al. (2003).

Table 7 presents the survey items that correspond to the BI construct and their possible scores. The table contains the percentage of the sample that indicated each score. For the survey item "I intend to use the method in the next 12 months," most responses belong to the three scores of "Disagree" (21.95%), "Agree" (21.95%), and "Strongly disagree" (19.51%). 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 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%).

Table 7

Percentages of Responses for Behavioral Intention (BI) Survey Items

Item	Strongly Disagree	Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Agree	Strongly Agree
I intend to use the method in the next 12 months.	19.51	21.95	2.44	9.76	17.07	21.95	7.32
I predict I would use the method in the next 12 months.	19.51	24.39	2.44	4.88	12.20	26.83	9.76
I plan to use the method in the next 12 months.	21.95	24.39	0.00	12.20	9.76	21.95	9.76

Note. Survey items were adapted from Venkatesh et al. (2003).

Table 8 presents the survey item that corresponds to the FoU and its possible scores. The table contains the percentage of the sample that indicated each score. The score with the most responses is “Never” (48.78%). 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.”

Table 8

Percentage of Responses for Frequency of Use (FoU) Survey Item

Never	Rarely	Occasionally	Sometimes	Frequently	Usually	Every Time
48.78	12.20	9.76	14.63	4.88	9.76	0.00

Research Question 1/Hypothesis

RQ1

To what extent is there a relationship between the acceptance of metamorphic testing (MT) among open-source software developers and the following constructs: (a) performance expectancy and (b) effort expectancy?

$H1_a$

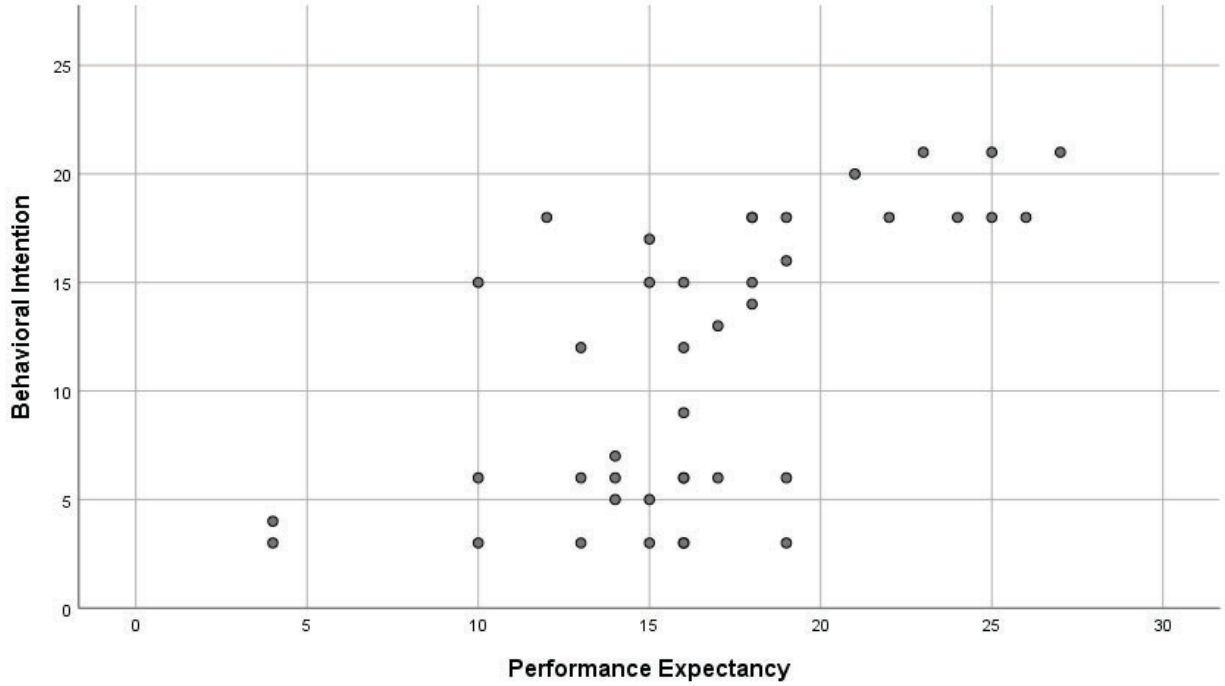
There is a statistically significant relationship between MT's acceptance among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.

The acceptance of MT was represented in this study by the BI. The researcher performed Spearman's rank-order correlation test to assess the BI and PE relationship among contributors to open-source software projects. There was a statistically significant, moderate positive correlation

between the BI and the PE, $r_s(39) = .636$, $p < .001$. Therefore, we can reject the null hypothesis and accept the alternative hypothesis. A scatterplot of the BI versus the PE is shown in Figure 1.

Figure 1

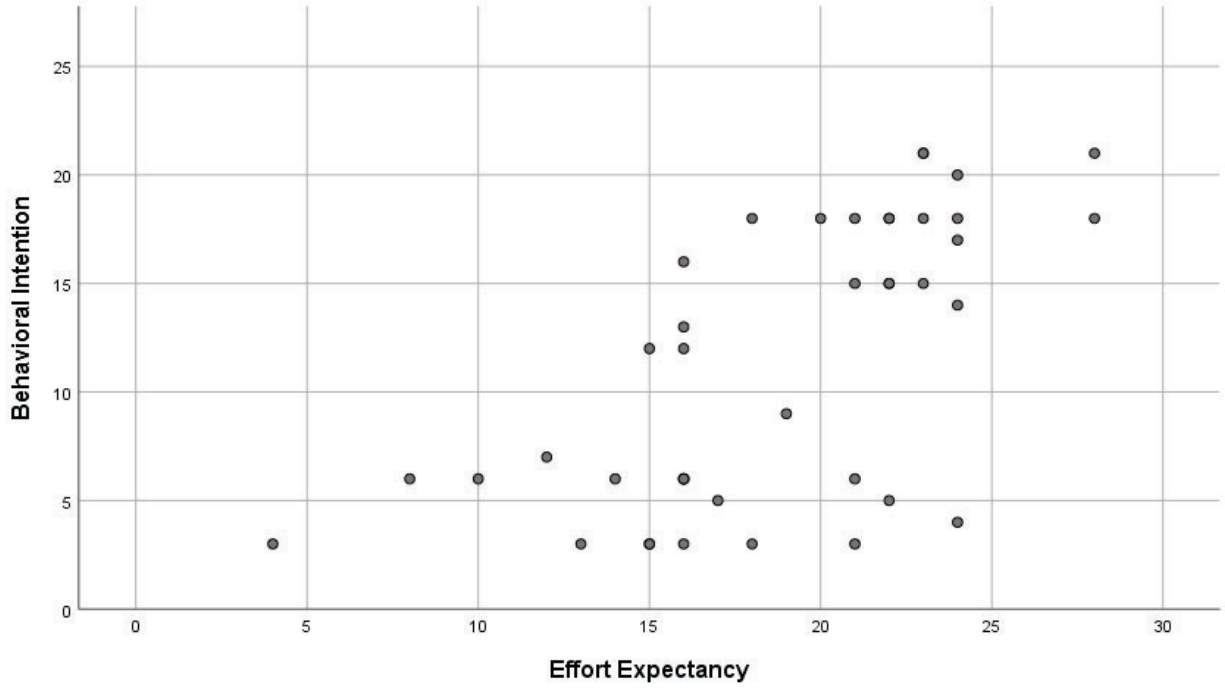
Scatterplot of Behavioral Intention vs. Performance Expectancy



The researcher carried out another Spearman's rank-order correlation test to assess the relationship between the BI and the EE among contributors to open-source software projects. Similarly to the relationship between the BI and the PE, there was a statistically significant, moderate positive correlation between the BI and the EE, $r_s(39) = .635$, $p < .001$. A scatterplot of the BI versus the EE is shown in Figure 2.

Figure 2

Scatterplot of Behavioral Intention vs. Effort Expectancy



The researcher performed MMRs to assess age's effect on the relationship between the BI and PE and the relationship between the BI and EE. Age did not moderate the relationship between the BI and the PE, as evidenced by an increase in total variation explained of 1.2%, which was not statistically significant ($F(1, 34) = .767$, $p = .387$). Age did not moderate the relationship between the BI and the EE, as evidenced by an increase in total variation explained of 2.7%, which was not statistically significant ($F(1, 34) = 1.721$, $p = .198$).

The researcher performed MMRs to assess gender's effect on the relationship between the BI and PE and the relationship between the BI and EE. Gender did not moderate the relationship between the BI and the PE, as evidenced by an increase in total variation explained of 0.4%, which was not statistically significant ($F(1, 35) = .260$, $p = .613$). Gender did not

moderate the relationship between the BI and the EE, as evidenced by an increase in total variation explained of 0.2%, which was not statistically significant ($F(1, 35) = .123$, $p = .728$).

The researcher performed MMRs to assess experience's effect on the relationship between the BI and PE and on the relationship between the BI and EE. Experience did not moderate the relationship between the BI and the PE, as evidenced by an increase in total variation explained of 0.0%, which was not statistically significant ($F(1, 36) = .002$, $p = .961$). Experience did not moderate the relationship between the BI and the EE, as evidenced by an increase in total variation explained of 0.1%, which was not statistically significant ($F(1, 36) = .052$, $p = .822$).

The data met the assumptions of the statistical tests for all analyses except for one. For the analysis that evaluated gender's effect on the relationship between the BI and EE, the assumption of a normal distribution of the residuals was violated. The researcher handled this violation by transforming the dependent variable using a "reflect and square root" transformation (Lund Research Ltd, 2018e).

Research Question 2/Hypothesis

RQ2

To what extent is there a relationship between the frequency of use of MT among open-source software developers and the following constructs: (a) performance expectancy and (b) effort expectancy?

$H2_a$

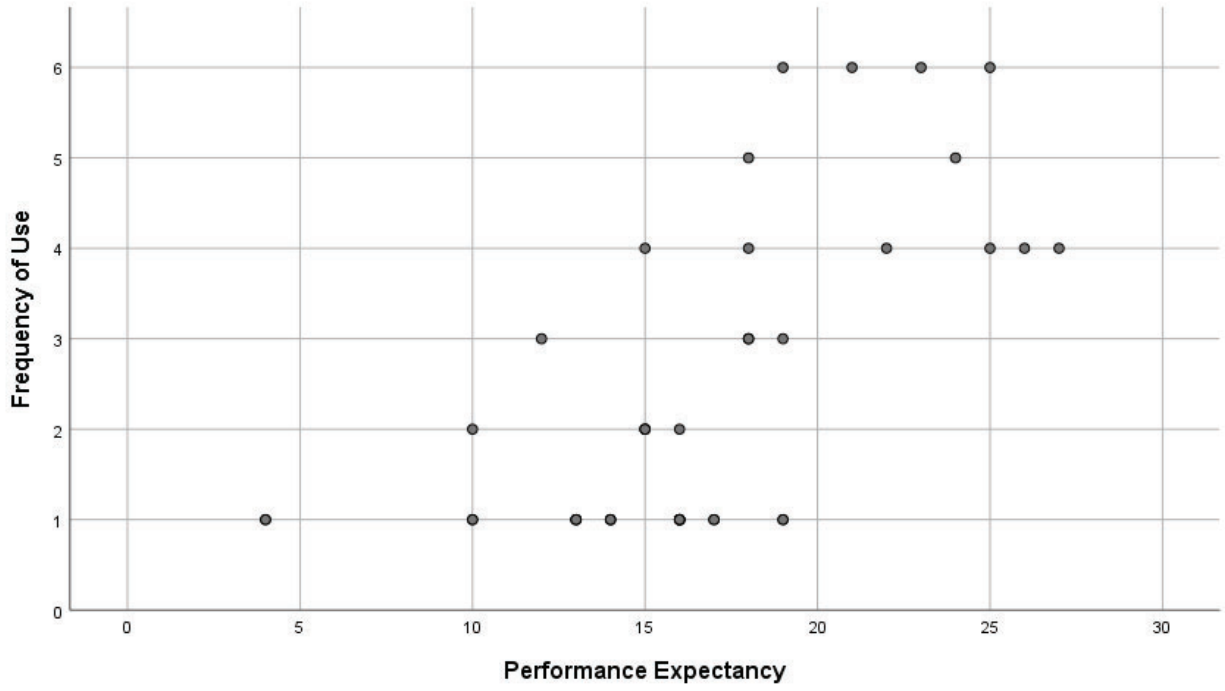
There is a statistically significant relationship between the frequency of use of MT among open-source software developers and either of the following constructs: (a) performance expectancy or (b) effort expectancy.

The researcher performed a Spearman's rank-order correlation to assess the relationship between the FoU and the PE among contributors to open-source software projects. As with the relationship between the BI and the PE, there was a statistically significant, moderate positive correlation between the FoU and the PE, $r_s(39) = .642$, $p < .001$. Therefore, we can reject the null hypothesis and accept the alternative hypothesis. A scatterplot of the FoU versus the PE is shown in

Figure **3**.

Figure 3

Scatterplot of Frequency of Use vs. Performance Expectancy

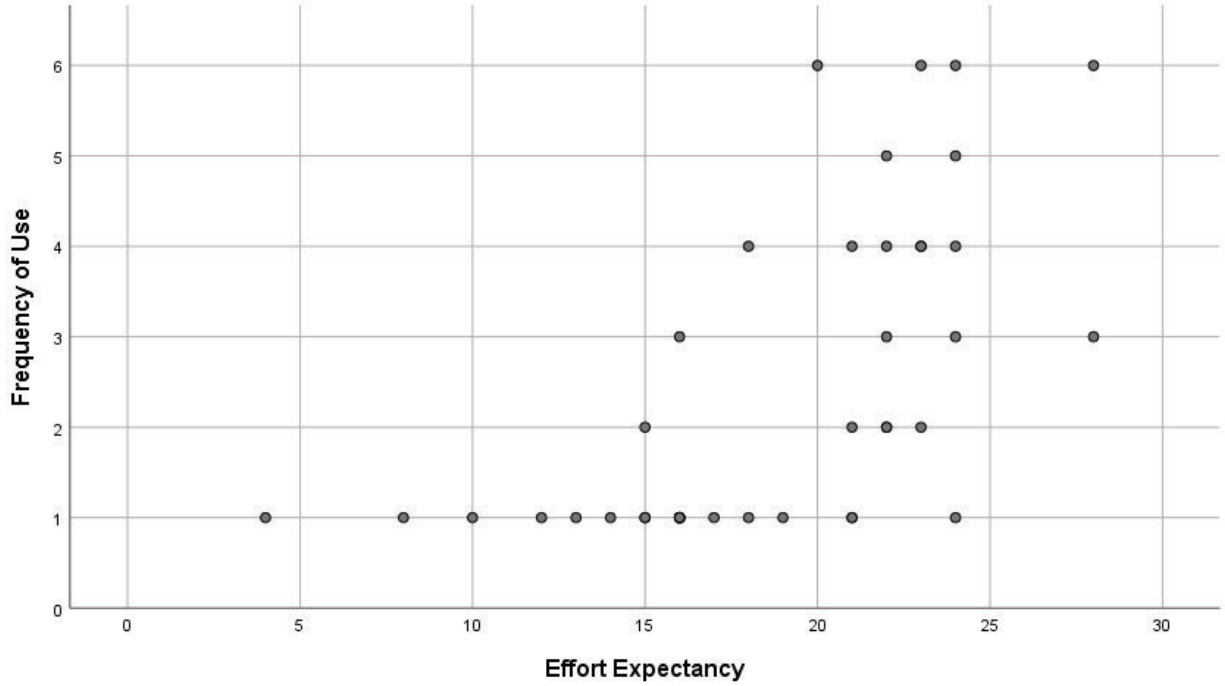


The researcher performed another Spearman's rank-order correlation to assess the relationship between the FoU and the EE among contributors to open-source software projects.

There was a statistically significant, strong positive correlation between the FoU and the EE, $r_s(39) = .701$, $p < .001$. A scatterplot of the FoU versus the EE is shown in Figure 4.

Figure 4

Scatterplot of Frequency of Use vs. Effort Expectancy



The researcher carried out MMRs to assess age's effect on the FoU and PE relationship and the FoU and EE relationship. Age did not moderate the relationship between the FoU and the PE, as evidenced by an increase in total variation explained of 3.3%, which was not statistically significant ($F(1, 34) = 2.222$, $p = .145$). Age did not moderate the relationship between the FoU and the EE, as evidenced by an increase in total variation explained of 6.5%, which was not statistically significant ($F(1, 34) = 3.930$, $p = .056$).

The researcher performed MMRs to assess gender's effect on the relationship between the FoU and PE and the relationship between the FoU and EE. Gender did not moderate the

relationship between the FoU and the PE, as evidenced by an increase in total variation explained of 0.0%, which was not statistically significant ($F(1, 35) = .020$, $p = .888$). Gender did not moderate the relationship between the FoU and the EE, as evidenced by an increase in total variation explained of 0.0%, which was not statistically significant ($F(1, 35) = .007$, $p = .936$).

The researcher carried out MMRs to assess experience's effect on the relationship between the FoU and PE and between the FoU and EE. Experience did not moderate the relationship between the FoU and the PE, as evidenced by an increase in total variation explained of 0.3%, which was not statistically significant ($F(1, 36) = .480$, $p = .493$). Experience did not moderate the relationship between the FoU and the EE, as evidenced by an increase in total variation explained of 1.0%, which was not statistically significant ($F(1, 36) = 1.107$, $p = .300$).

The data met the assumptions of the statistical tests for all analyses except for two. For the analysis that assessed experience's effect on the relationship between the FoU and the PE, one outlier was detected. The researcher handled this violation by transforming the dependent variable using a "square root" transformation (Lund Research Ltd, 2018e). For the analysis that assessed experience's effect on the relationship between the FoU and EE, the assumption of a normal distribution of the residuals was violated. The researcher also addressed this violation by transforming the dependent variable using a "square root" transformation.

Evaluation of the Findings

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. The researcher found a moderate positive correlation between the BI and the PE. Also, the researcher found a moderate positive correlation between the BI and the EE. This finding is consistent with

UTAUT, which stipulates that the PE and EE relate to the BI (Venkatesh et al., 2003). Other studies in which the researchers applied UTAUT to software development practices and processes have yielded inconsistent findings. In an analysis of the adoption of continuous delivery among software development project managers, Anderson (2019) found that the BI was moderately positively correlated with the PE but did not find a significant relationship between the EE and the BI. On the other hand, in a study of software development process adoption, Guardado (2012) found that the EE, but not the PE, was a significant determinant of the BI.

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. The researcher found a moderate positive correlation between the FoU and the PE and a strong positive correlation between the FoU and the EE. This result is consistent with the technology acceptance model (TAM), on which UTAUT is partially based. The TAM stipulates that perceived usefulness (PU) and perceived ease of use (PEOU), which correspond heavily to the PE and EE respectively, are the primary determinants of technology usage behaviors (Davis, 1986).

The results indicate that none of the previous relationships were moderated by age, gender, or experience. This finding is inconsistent with UTAUT. In UTAUT, age, gender, and experience moderate the relationship between the EE and the BI, and age and gender moderate the relationship between the PE and the BI (Venkatesh et al., 2003). However, Anderson (2019) found that experience did not moderate the relationship between the PE and BI, nor did it moderate the EE and BI relationship.

Summary

In this study, the researcher administered a previously validated survey instrument to a group of open-source software contributors to measure their FoU of the MT method and the

UTAUT constructs of the PE, EE, and BI. The researcher carried out a descriptive analysis of the variables of interest. Next, the researcher performed Spearman's rank-order correlation tests and found significant positive relationships between the BI and PE, the BI and EE, the FoU and PE, and the FoU and EE. This finding is consistent with the theoretical frameworks of the UTAUT and the TAM. The researcher did not detect moderating effects of age, gender, or experience on any of the relationships examined, which is inconsistent with the UTAUT. Other studies in which researchers employed UTAUT in software development contexts yielded mixed results.

Chapter 5: Implications, Recommendations, and Conclusions

The study addressed the inadequacy of software testing methods to detect all defects, which could be mitigated by using the metamorphic testing (MT) method (Lidbury et al., 2015; Rao et al., 2013). Software defects can result in poor software performance, reduced precision or accuracy of software output, and retractions of research publications for which defective software was used (Kanewala & Chen, 2018). Detecting software defects can be challenging due to the oracle problem, code complexity, and time and resource constraints (Chen et al., 2018; Ding et al., 2016; Zhou & Sun, 2019).

The purpose of this study was to examine relationships between the use and acceptance of the MT method among open-source developers and the constructs of performance expectancy (PE) and effort expectancy (EE), which may be related to MT use and acceptance. The PE and EE are stipulated by the unified theory of acceptance and use of technology (UTAUT), which served as the theoretical framework for this study. Another goal of this study was to understand how the variables of age, gender, and experience moderate these relationships. The study findings can help practitioners create interventions to increase MT usage and reduce the number of software defects. Furthermore, the study results can be used to evaluate the applicability of the UTAUT to software testing methods.

The researcher applied a quantitative methodology with a correlational research design to carry out this study. The study design was appropriate for addressing the study problem and purpose; in this study, the researcher examined the correlation between the use and acceptance of the MT method, which is effective at finding software defects, and factors that could be significantly related to the use and acceptance of the MT method. The UTAUT construct of behavioral intention (BI) represented MT acceptance in this study. The use of MT was

operationalized in this study by the variable of frequency of use (FoU). Due to its cost-effectiveness and high accessibility to the target population, an online survey instrument was used to collect data from the study sample. After performing data analysis, the researcher found moderate to strong positive correlations between these four variable pairs: the PE and BI, the EE and BI, the PE and FoU, and the EE and FoU. The study results did not indicate a moderating effect of age, gender, or experience on any of the relationships between these variable pairs.

One limitation of the study was nonresponse bias since not all potential participants recruited for the study completed the survey. Another limitation was the study data's subjectivity because the researcher could not verify the accuracy of the self-reported survey responses. Furthermore, the correlational research design means that the researcher could not infer causal relationships between constructs and variables from the study findings. The study methodology limited the results to the variables and constructs of interest. Another study limitation was the sampling method. The researcher only recruited contributors to the open-source software (OSS) projects in the GitHub “Software in science” collection who were at least 18 years old and spoke English fluently. Hence, the study findings may not be generalizable to all OSS developers.

In this chapter, the researcher discusses the study's implications for the study problem, purpose, and research questions. Also, the researcher explains how the study findings contribute to the guiding theoretical framework and existing literature. Next, recommendations for applying the study findings to theory and practice are outlined. Then, suggestions for future research that builds on the study findings are listed. Finally, the researcher draws conclusions regarding the research problem addressed and the significance of the study findings.

Implications

One factor that could have affected the interpretation of the study findings was the presence of nonresponse bias. The wave analysis results suggest that nonrespondents would tend to score slightly less on the FoU, BI, PE, and EE than did the study sample. This finding may be due to nonrespondents being less interested or less familiar with MT than respondents are, and hence being less likely to use or intend to use MT. Nonrespondents may also be less likely to believe that MT would help improve their job performance or be easy to use. Another factor that could have influenced the study findings' interpretation is the self-reported nature of the data. Some participants may not have answered all survey items honestly and accurately. Finally, only contributors to the OSS projects in the GitHub "Software in science" collection who were at least 18 years old and spoke English fluently were recruited for this study. Hence, the study findings may not be generalizable to OSS developers who do not have these characteristics.

Research Question 1/Hypothesis

The researcher found a statistically significant, moderate positive correlation between the BI and the PE and between the BI and the EE. Therefore, the study findings suggest that the alternative hypothesis can be accepted. In other words, there is a statistically significant relationship between the acceptance of MT among OSS developers and either the PE or EE.

These findings align with the study purpose, which is to examine relationships between the MT method's use and acceptance among OSS developers and the PE and EE constructs. The research problem is the inadequacy of conventional software testing methods to detect all software defects. Existing research suggests that this problem could be mitigated using the MT method when testing software (Lidbury et al., 2015; Rao et al., 2013). The study findings suggest that increasing the PE and EE, as they pertain to the MT method, among OSS developers will

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 use will increase MT's acceptance. This outcome could mitigate the study problem by resulting in fewer software defects.

The findings are consistent with the theoretical framework of UTAUT, which stipulates that the PE and EE are related to the BI (Venkatesh et al., 2003). Although a literature review did not uncover any 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. On the other hand, in a study of software development process acceptance, Guardado (2012) found that the EE, but not the PE, was a significant determinant of the BI. The inconsistency of these findings may be due to the different subject matter of the studies and differences in the study samples' composition. 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.

The acceptance of MT was operationalized in this study by the UTAUT construct of BI, which may have influenced these results' interpretation. The BI construct only represents the participants' intention to use the new technology within a specific time period. In this study, this 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 into the future.

Research Question 2/Hypothesis

The researcher found a statistically significant, moderate positive correlation between the FoU and the PE. Also, the researcher found a statistically significant, strong positive correlation between the FoU and the EE. Therefore, the study findings suggest that the alternative hypothesis can be accepted. In other words, there is a statistically significant relationship between the FoU of the MT method among OSS developers and either the PE or EE.

These findings align with the study purpose, which is to examine relationships between the MT method's use and acceptance among OSS developers and the PE and EE constructs. The research problem is the inadequacy of conventional software testing methods to detect all software defects. Existing research suggests that this problem could be mitigated by using the MT method when testing software (Lidbury et al., 2015; Rao et al., 2013). The study findings suggest that increasing the PE and EE, as they pertain to the MT method, will increase the MT method's acceptance and usage among OSS developers. In other words, increasing the extent to which developers believe that MT will improve their job performance and be easy to use will increase MT usage. Increasing usage of the MT method may mitigate the study problem by resulting in fewer software defects.

The study findings indicate that none of the relationships mentioned earlier are moderated by the variables of age, gender, or experience. This finding is inconsistent with UTAUT, in

which age, gender, and experience moderate the relationship between the EE and the BI (Venkatesh et al., 2003). Furthermore, UTAUT specifies that age and gender moderate the relationship between the PE and the BI. One factor that may affect these findings is the highly uneven gender distribution of the study sample, with only 4.9% identifying as female. Another possible reason for the divergent results is that UTAUT was developed using a study that was performed in a different context than this study. Venkatesh et al. (2003) created UTAUT based on a study on the acceptance and use of an IT system among non-IT employees who were initially unfamiliar with the system. It is possible that age, gender, and experience played a more critical role in technology adoption in this context. Furthermore, Venkatesh et al. (2003) collected data from a larger sample, which gave the study greater statistical power to detect moderating effects. On the other hand, the study findings were consistent with Anderson (2019), who found that experience did not moderate the relationship between the PE and BI or between the EE and BI.

Another contribution this study makes to the existing literature is measuring MT's acceptance and usage among a sample of OSS developers. The BI's descriptive statistics 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.

Theoretical Framework

The study findings have implications for the UTAUT, which is the theoretical framework that guided the design of this study. Also, the study findings provide insights into related

theoretical frameworks, namely the TAM and the HMSAM. In this section, the researcher discusses the contributions of the study to theory.

The researcher 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. The results also support adding positive relationships between actual technology usage and both the PE and EE constructs to the UTAUT framework. Although the existing UTAUT framework stipulates positive relationships between the BI construct and the PE and EE, the framework does not specify relationships between actual usage and the PE and EE. In addition, the study findings do not support the idea that the variables of age, gender, and experience moderate the relationships between the BI, PE, and EE. Hence, the study results support removing these moderating variables from the UTAUT framework in the context of software testing method adoption.

These findings also support extending the technology acceptance model (TAM), on which the UTAUT is partially based, to the study of software testing method usage. The TAM stipulates that perceived usefulness (PU) and perceived ease of use (PEOU), which correspond heavily to the PE and EE respectively, are the primary determinants of technology usage behaviors (Davis, 1986). Since the hedonic-motivation system adoption model (HMSAM) is a variation of the TAM containing the PU and PEOU constructs, the study findings support the applicability of this portion of the HMSAM to software testing methods (Lowry et al., 2013).

Recommendations for Practice

The results of the study have important implications for software development practice. The researcher developed recommendations for practice based on the study findings. In this

section, the researcher lists the recommendations that correspond to each research question addressed in the study.

Research Question 1/Hypothesis

1. Practitioners should develop interventions to increase the PE and EE among OSS developers in order to improve MT acceptance. The finding that MT acceptance is positively related to the PE and EE constructs supports the need for interventions that increase the PE and EE among OSS developers. The result that nearly half of the sample does not intend to use MT supports such interventions' importance. These interventions should explain how using the MT method can improve job performance. Such explanations could increase the PE among OSS developers. 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 & Chen, 2014; Kanewala & Chen, 2018; Zhou & Sun, 2019).
2. Education and training on the MT method should be provided to OSS developers. Educators should incorporate the MT method into software testing books and curricula for software engineering and computer science degree programs. Education on MT could increase the PE and EE among OSS developers. Furthermore, organizations could increase MT acceptance 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 EE among OSS developers. Based on the study findings, increasing the EE and PE would be likely to result in a growth of MT acceptance among OSS developers.

Research Question 2/Hypothesis

1. Practitioners should develop interventions to increase the EE among OSS developers in order to increase MT usage. The finding that MT usage is strongly positively related to the EE construct supports the need for interventions that increase the EE among OSS developers. The result that nearly half of the sample never uses MT supports such interventions' importance. These interventions should familiarize developers with research about the appropriate selection of metamorphic relations (MRs). Such interventions 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, 2015; 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).

Recommendations for Future Research

Future researchers might build on this study by examining the factors associated with MT use and acceptance using qualitative methodology. 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. From this methodology, the researcher 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. Future research could also involve developing interventions to

increase MT usage and testing their effectiveness using an experimental or quasi-experimental design.

The researcher only recruited participants who have contributed to OSS projects, limiting the study findings' generalizability to organizational contexts. Future 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. Recruiting more female participants would increase the study's ability to detect the effect of gender on the relationships between the constructs of interest.

The next 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 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 give 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 (Gonçalves et al., 2017; Majchrzak, 2010; Seth et al., 2014; Zhang, 2009).

Also, future research could 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 et al., 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.

Conclusions

The study addressed the problem of the inadequacy of conventional software testing methods to detect all software defects. This problem is significant because software defects can cause various issues, including poor software performance, low precision or accuracy of software output, and incorrect research results. Existing research suggests that the MT method finds many defects that conventional testing methods cannot detect. The MT method can also handle the oracle problem, a common challenge associated with testing scientific software. However, there is little research regarding the acceptance and use of MT among software developers and factors associated with its use and acceptance.

In this quantitative correlational study, the researcher examined relationships between the MT method's use and acceptance among OSS developers and the UTAUT constructs of PE and EE. The UTAUT served as the guiding theoretical framework for this study. Another objective of the study was to understand how the UTAUT moderating variables of age, gender, and experience affect these relationships.

The study results indicated significant moderate to strong positive relationships between MT use and acceptance with both the PE and EE constructs. On the other hand, the researcher did not detect a moderating effect of age, gender, or experience on any relationship between these variables. The study findings support the relevance of the UTAUT constructs and relationships, but not the UTAUT moderating variables, to adopting software testing methods among developers. Furthermore, the findings support the applicability of the TAM constructs of

PU and PEOU to 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 will increase their acceptance and use of MT.

References

Abackerli, A. J., Pereira, P. H., & Calonego, N. (2010). A case study on testing CMM uncertainty simulation software (VCMM). *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, *32*(1), 8-14. doi:10.1590/S1678-58782010000100002

Adams, D. A., Nelson, R. R., & Todd, P. A. (1992). Perceived usefulness, ease of use, and usage of information technology: A replication. *MIS Quarterly*, *16*(2), 227-247. doi:10.2307/249577

Agarwal, R., & Karahanna, E. (2000). Time flies when you're having fun: Cognitive absorption and beliefs about information technology usage. *MIS Quarterly*, *24*(4), 665-694. doi:10.2307/3250951

Ajibade, P. (2018). Technology acceptance model limitations and criticisms: Exploring the practical applications and use in technology-related studies, mixed-method, and qualitative researches. *Library Philosophy and Practice*. http://digitalcommons.unl.edu/libphilprac/1941

Ajzen, I. (1991). The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, *50*(2), 179-211. https://doi.org/10.1016/0749-5978(91)90020-T

Ajzen, I., & Fishbein, M. (1980). *Understanding attitudes and predicting social behavior*. Prentice-Hall.

Al-Azawei, A., Parslow, P., & Lundqvist, K. (2017). Investigating the effect of learning styles in a blended e-learning system: An extension of the technology acceptance model (TAM). *Australasian Journal of Educational Technology*, *33*(2). https://doi.org/10.14742/ajet.2741

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

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. http://dx.doi.org/10.14257/ijeic.2015.6.1.06

Al-Maroof, R. S., Salloum, S. A., AlHamadand, A. Q. M., & Shaalan, K. (2020). Understanding an extension technology acceptance model of Google Translation: A multi-cultural study in United Arab Emirates. *International Journal of Interactive Mobile Technologies*, *14*(3). https://doi.org/10.3991/ijim.v14i03.11110

Al-Qaysi, N., Mohamad-Nordin, N., & Al-Emran, M. (2020). A systematic review of social media acceptance from the perspective of educational and information systems theories and models. *Journal of Educational Computing Research*, *57*(8), 2085-2109. doi:10.1177/0735633118817879

Alrawashdeh, T. A., Elbes, M. W., Almomani, A., ElQirem, F., & Tamimi, A. (2019). User acceptance model of open source software: An integrated model of OSS characteristics and UTAUT. *Journal of Ambient Intelligence and Humanized Computing*. https://doi.org/10.1007/s12652-019-01524-7

Al-Saedi, K., Al-Emran, M., Abusham, E., & El Rahman, S. (2019). Mobile payment adoption: A systematic review of the UTAUT model. *2019 International Conference on Fourth Industrial Revolution (ICFIR)*. doi:10.1109/ICFIR.2019.8894794

Anderson, A. J. (2019). *Examination of adoption theory on the DevOps practice of continuous delivery* (Publication No. 22624635) [Doctoral dissertation, Walden University]. ProQuest.

Avelino, G., Passos, L., Hora, A., & Valente, M. T. (2016). A novel approach for estimating Truck Factors. *2016 IEEE 24th International Conference on Program Comprehension (ICPC)*. doi:10.1109/ICPC.2016.7503718

Bagozzi, R. P. (2007). The legacy of the Technology Acceptance Model and a proposal for a paradigm shift. *Journal of the Association for Information Systems*, *8*(4), 244-254. Retrieved from https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1406&context=jais

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

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Prentice-Hall.

Bao, Z., Murray, J. I., Boyle, T., Ooi, S. L., Sandel, M. J., & Waterston, R. H. (2006). Automated cell lineage tracing in *Caenorhabditis elegans*. *Proceedings of the National Academy of Sciences of the United States of America*, *103*(8), 2707-2712. https://doi.org/10.1073/pnas.0511111103

Basili, V. R., Carver, J. C., Cruzes, D., Hochstein, L. M., Hollingsworth, J. K., Shull, F., & Zelkowitz, M. V. (2008). Understanding the high-performance-computing community: A software engineer's perspective. *IEEE Software*, *25*(4). doi:10.1109/MS.2008.103

Boyle, T. J., Bao, Z., Murray, J. I., Araya, C. L., & Waterston, R. H. (2006). AceTree: A tool for visual analysis of *Caenorhabditis elegans embryogenesis*. *BMC Bioinformatics*, *7*. doi:10.1186/1471-2105-7-275

Brown, S. A., & Venkatesh, V. (2005). Model of adoption of technology in households: A baseline model test and extension incorporating household life cycle. *MIS Quarterly*, *29*(3), 399-426. doi:10.2307/25148690

Burnstein, I. (2003). *Practical software testing: A process-oriented approach*. Springer.

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. doi:10.1016/j.eswa.2019.04.070

Cao, Y., Zhou, Z. Q., & Chen, T. Y. (2013). Metamorphic relations and dissimilarities of test case executions. *2013 13th International Conference on Quality Software*, 153-162. doi:10.1109/QSIC.2013.43

Chandio, F. H., Irani, Z., Zeki, A. M., Shah, A., & Shah, S. C. (2017). Online banking information systems acceptance: An empirical examination of system characteristics and Web security. *Information Systems Management*, *34*(1), 50-64. https://doi.org/10.1080/10580530.2017.1254450

Chaparro, O., Bernal-Cárdenas, C., Lu, J., Moran, K., Marcus, A., Penta, M. D., Poshyvanyk, D., & Ng, V. (2019). Assessing the quality of the steps to reproduce in bug reports. *European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 86-96. doi:10.1145/3338906.3338947

Chen, T. Y., Cheung, S. C., & Yiu, S. M. (1998). *Metamorphic testing: A new approach for generating next test cases* (Report No. HKUST-CS98-01). Hong Kong, China: Hong Kong University of Science and Technology.

Chen, T. Y., Huang, D. H., Tse, T. H., & Zhou, Z. Q. (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. Retrieved from https://www.researchgate.net/publication/228781884_Case_Studies_on_the_Selection_of_Useful_Relations_in_Metamorphic_Testing

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). 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

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 (IOSR-JHSS)*, *19*(4), 99-104. Retrieved from http://www.academia.edu/download/37208325/N0194399104.pdf

Chuttur, M. (2009). Overview of the technology acceptance model: Origins, developments, and future directions. *Sprouts: Working Papers on Information Systems*, *9*(37). http://aisel.aisnet.org/sprouts_all/290

Coelho, J., & Valente, M. T. (2017). Why modern open source projects fail. *Foundations of Software Engineering*, 186-196. doi:10.1145/3106237.3106246

Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2003). *Applied multiple regression/correlation analysis for the behavioral sciences*. Lawrence Erlbaum Associates.

Compeau, D. R., Higgins, C. A., & Huff, S. (1999). Social cognitive theory and individual reactions to computing technology: A longitudinal study. *MIS Quarterly*, *23*(2), 145-158. doi:10.2307/249749

Constantinou, E., Ampatzoglou, A., & Stamelos, I. (2015). Quantifying reuse in OSS: A large-scale empirical study. *International Journal of Open Source Software & Processes*, *5*(3), 1-19. doi:10.4018/IJOSSP.2014070101

Corder, G. W., & Foreman, D. I. (2014). *Nonparametric statistics: A step-by-step approach*. Hoboken, NJ: John Wiley & Sons, Inc.

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

Davis, F. D. (1986). *A technology acceptance model for empirically testing new end-user information systems: Theory and results.* [Doctoral dissertation, Massachusetts Institute of Technology]. MIT Libraries. https://dspace.mit.edu/handle/1721.1/15192

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, *13*(3), 319-339. doi:10.2307/249008

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1989). User acceptance of computer technology: A comparison of two theoretical models. *Management Science*, *35*(8), 982-1003. doi:10.1287/mnsc.35.8.982

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1992). Extrinsic and intrinsic motivation to use computers in the workplace. *Journal of Applied Social Psychology*, *22*(14), 1111-1132. https://doi.org/10.1111/j.1559-1816.1992.tb00945.x

Dembsey, J. (2020, June 26). *What are the IRB requirements for my recruitment materials?* Northcentral University Institutional Review Board. https://ncu.libanswers.com/irb/faq/278108

Diethelm, K. (2012). The limits of reproducibility in numerical simulation. *Computing in Science & Engineering*, *14*(1), 64-72. doi:10.1109/MCSE.2011.21

Ding, J., Zhang, D., & Hu, X.-H. (2016). An application of metamorphic testing for testing scientific software. *2016 1st International Workshop on Metamorphic Testing*, 37-43. doi:10.1109/MET.2016.015

Dinh-Trong, T. T., & Bieman, J. M. (2005). The FreeBSD project: A replication case study of open source development. *IEEE Transactions on Software Engineering*, *31*(6), 481-494. doi:10.1109/TSE.2005.73

Dubey, A., & Wan, H. (2018). Methodology for building granular testing in multicomponent scientific software. *2018 IEEE/ACM 13th International Workshop on Software Engineering for Science.*

Dubois, P. F. (2012). Testing scientific programs. *Computing in Science & Engineering*, *14*(4), 69-73. doi:10.1109/MCSE.2012.84

Dwivedi, Y. K., Rana, N. P., Jeyaraj, A., Clement, M., & Williams, M. D. (2017). Re-examining the unified theory of acceptance and use of technology (UTAUT): Towards a revised theoretical model. *Information Systems Frontiers*, *21*, 719-734. doi:10.1007/s10796-017-9774-y

Everett, G. D., & McLeod, R., Jr. (2007). *Software testing: Testing across the entire software development life cycle*. John Wiley & Sons, Inc.

Farhoodi, R., Garousi, V., Pfahl, D., & Sillito, J. (2013). Development of scientific software: A systematic mapping, a bibliometrics study, and a paper repository. *International Journal of Software Engineering and Knowledge Engineering*, *23*(4). doi:10.1142/S0218194013500137

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.

Fazzini, M., Prammer, M., d'Amorim, M., & Orso, A. (2018). Automatically translating bug reports into test cases for mobile apps. *Software Testing and Analysis*, 141-152. doi:10.1145/3213846.3213869

Fishbein, M. (1967). *Readings in attitude theory and measurement*. John Wiley & Sons.

Fishbein, M., & Ajzen, I. (1975). *Belief, attitude, intention, and behavior: An introduction to theory and research*. Addison-Wesley.

Foucault, M., Palyart, M., Blanc, X., Murphy, G., & Falleri, J.-R. (2015). Impact of developer turnover on quality in open-source software. *Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 829-841. doi:10.1145/2786805.2786870

García, A. S., Fernández-Sotos, P., Fernández-Caballero, A., Navarro, E., Latorre, J. M., Rodriguez-Jimenez, R., & González, P. (2019). Acceptance and use of a multi-modal avatar-based tool for remediation of social cognition deficits. *Journal of Ambient Intelligence and Humanized Computing*. https://doi.org/10.1007/s12652-019-01418-8

Geldenhuys, J. (2010). Finding the core developers. *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, 447-450. doi:10.1109/SEAA.2010.66

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

GitHub, Inc. (2020b). *The world's leading software development platform - GitHub*. https://github.com

Goeminne, M., & Mens, T. (2011). Evidence for the Pareto principle in open source software activity. *CSMR 2011 Workshop on Software Quality and Maintainability (SQM)*. Retrieved from https://www.researchgate.net/profile/Tom_Mens/publication/228728263_Evidence_for_the_Pareto_principle_in_Open_Source_Software_Activity/links/0deec518e5b1e8ca77000000/Evidence-for-the-Pareto-principle-in-Open-Source-Software-Activity.pdf

Gonçalves, W. F., de Almeida, C. B., de Araújo, L. L., Ferraz, M. S., Xandú, 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. *2017 12th Iberian Conference on Information Systems and Technologies (CISTI).* doi:10.23919/CISTI.2017.7975873

Gousios, G., Zaidman, A., Storey, M.-A., & Deursen, A. v. (2015). Work practices and challenges in pull-based development: The integrator's perspective. *Software Engineering*, 358-368. doi:10.1109/ICSE.2015.55

Guardado, D. R. (2012). *Process acceptance and adoption by IT software project practitioners* (Publication No. 3512446) [Doctoral dissertation, Capella University]. UMI Dissertation Publishing.

Gunawan, H. (2018). Identifying factors affecting smart city adoption using the unified theory of acceptance and use of technology (UTAUT) method. *2018 International Conference on Orange Technologies (ICOT)*. doi:10.1109/ICOT.2018.8705803

Haefliger, S., von Krogh, G., & Spaeth, S. (2008). Code reuse in open source software. *Management Science*, *54*(1), 180-193. doi:10.1287/mnsc.1070.0748

Hai, L. C., & Alam Kazmi, S. H. (2015). Dynamic support of government in online shopping. *Asian Social Science*, *11*(22). doi:10.5539/ass.v11n22p1

Hatton, L. (1997). The T experiments: Errors in scientific software. *IEEE Computational Science and Engineering*, *4*(2), 27-38. doi:10.1109/99.609829

Hatton, L., & Roberts, A. (1994). How accurate is scientific software? *IEEE Transactions on Software Engineering*, *20*(10), 785-797. doi:10.1109/32.328993

Heaton, D., & Carver, J. C. (2015). Claims about the use of software engineering practices in science: A systematic literature review. *Information and Software Technology*, 207-219. doi:10.1016/j.infsof.2015.07.011

Hendrickson, A. R., Massey, P. D., & Cronan, T. P. (1993). On the test-retest reliability of perceived usefulness and perceived ease of use scales. *MIS Quarterly*, *17*(2), 227-230. doi:10.2307/249803

Hertel, G., Niedner, S., & Herrmann, S. (2003). Motivation of software developers in open source projects: An Internet-based survey of contributors to the Linux kernel. *Open Source Software Development*, *32*(7), 1159-1177. doi:10.1016/S0048-7333(03)00047-7

Hinsen, K. (2015). The approximation tower in computational science: Why testing scientific software is difficult. *Computing in Science & Engineering*, *17*(4), 72-77. doi:10.1109/MCSE.2015.75

Hojjati, S. N., & Khodakarami, M. (2016). Evaluation of factors affecting the adoption of smart buildings using the technology acceptance model. *International Journal of Advanced Networking and Applications*, *7*(6), 2936-2943. Retrieved from

https://search.proquest.com/openview/d2000e7122e5e73a34b74cd5f23fc0f1/1?pq-origsite=gscholar&cbl=886380

Hovy, C., & Kunkel, J. (2016). Towards automatic and flexible unit test generation for legacy HPC code. *2016 Fourth International Workshop on Software Engineering for High Performance Computing in Computational Science and Engineering (SE-HPCCSE)*. doi:10.1109/SE-HPCCSE.2016.005

Howden, W. E. (1978). Theoretical and empirical studies of program testing. *IEEE Transactions on Software Engineering*, 293-298. doi:10.1109/TSE.1978.231514

Hsu, C.-L., & Lu, H.-P. (2007). Consumer behavior in online game communities: A motivational factor perspective. *Computers in Human Behavior*, *23*(3), 1642-1659. doi:10.1016/j.chb.2005.09.001

Huang, F., & Teo, T. (2019). Influence of teacher-perceived organisational culture and school policy on Chinese teachers' intention to use technology: An extension of technology acceptance model. *Educational Technology Research and Development*, *68*, 1547-1567. doi:10.1007/s11423-019-09722-y

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

Jegers, K. (2007). Pervasive game flow: Understanding player enjoyment in pervasive gaming. *ACM Computers in Entertainment*, *5*(1), 1-11. https://doi.org/10.1145/1236224.1236238

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*, 2035-2071. doi:10.1007/s10664-015-9393-5

Kanewala, U. G. (2015). *Testing scientific software: Techniques for automatic detection of metamorphic relations* (Doctoral dissertation). Retrieved from ProQuest Dissertations and Theses database. (UMI No. 3706367)

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. (2018). Metamorphic testing: A simple yet effective approach for testing scientific software. *Computing in Science & Engineering, 21*(1), 66-72. doi:10.1109/MCSE.2018.2875368

Karagöz, G., & Sözer, H. (2017). Reproducing failures based on semiformal failure scenario descriptions. *Software Quality Journal*, *25*(1), 111-129. doi:10.1007/s11219-016-9310-1

Katzman, B., Tang, D., Santella, A., & Bao, Z. (2018). AceTree: A major update and case study in the long-term maintenance of open-source scientific software. *BMC Bioinformatics*, *19*(121). https://doi.org/10.1186/s12859-018-2127-0

Kellogg, L. H., Hwang, L. J., Gassmoller, R., Bangerth, W., & Heister, T. (2019). The role of scientific communities in creating reusable software: Lessons from geophysics. *Computing in Science & Engineering*, *21*(2), 25-35. doi:10.1109/MCSE.2018.2883326

Kelly, D., Thorsteinson, S., & Hook, D. (2011). Scientific software testing: Analysis with four dimensions. *IEEE Software*, *28*(3), 84-90. doi:10.1109/MS.2010.88

Kim, M. J., & Hall, C. M. (2019). A hedonic motivation model in virtual reality tourism: Comparing visitors and non-visitors. *International Journal of Information Management*, *46*, 236-249. https://doi.org/10.1016/j.ijinfomgt.2018.11.016

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

Koch, S., & Schneider, G. (2002). Effort, co-operation, and co-ordination in an open source software project: GNOME. *Information Systems Journal*, *12*(1), 27-42. doi:10.1046/j.1365-2575.2002.00110.x

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

Koteska, B., Mishev, A., & Pejov, L. (2018). Quantitative measurement of scientific software quality: Definition of a novel quality model. *International Journal of Software Engineering and Knowledge Engineering*, *28*(3), 407-425. doi:10.1142/S0218194018500146

Laudon, K. C., & Laudon, J. P. (2012). *Management information systems: Managing the digital firm*. Prentice Hall.

Legris, P., Ingham, J., & Collerette, P. (2003). Why do people use information technology? A critical review of the technology acceptance model. *Information & Management*, *40*, 191-204. doi:10.1016/S0378-7206(01)00143-4

Leman, J. K., Weitzner, B. D., Renfrew, P. D., Lewis, S. M., Moretti, R., Watkins, A. M., Mulligan, V. K., Lyskov, S., Adolf-Bryfogle, J., Labonte, J. W., Krys, J., Bystroff, C., Schief, W., Gront, D., Schueler-Furman, O., Baker, D., Bradley, P., Dunbrack, R., Kortemme, T., … Bonneau, R. (2020). Better together: Elements of successful scientific

software development in a distributed collaborative community. *PLoS Computational Biology*, *16*(5). https://doi.org/10.1371/journal.pcbi.1007507

Leong, L. W., Ibrahim, O., Dalvi-Esfahani, M., Shahbazi, H., & Nilashi, M. (2018). The moderating effect of experience on the intention to adopt mobile social network sites for pedagogical purposes: An extension of the technology acceptance model. *Education and Information Technologies*, *23*, 2477-2498. https://doi.org/10.1007/s10639-018-9726-2

Li, J. (2020). Blockchain technology adoption: Examining the fundamental drivers. *Proceedings of the 2020 2nd International Conference on Management Science and Industrial Engineering*, 253-260. doi:10.1145/3396743.3396750

Lidbury, C., Lascu, A., Chong, N., & Donaldson, A. F. (2015). Many-core compiler fuzzing. *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'15)*, *50*(6), 65-76. https://dx.doi.org/10.1145/2737924.2737986

Lim, Y. J., Osman, A., Salahuddin, S. N., Romle, A. R., & Abdullah, S. (2016). Factors influencing online shopping behavior: The mediating role of purchase intention. *Procedia Economics and Finance*, *35*, 401-410. doi:10.1016/S2212-5671(16)00050-2

Lin, B., Robles, G., & Serebrenik, A. (2017). Developer turnover in global, industrial open source projects: Insights from applying survival analysis. *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*, 66-75. doi:10.1109/ICGSE.2017.11

Lin, X., Simon, M., & Niu, N. (2018). Hierarchical metamorphic relations for testing scientific software. *2018 ACM/IEEE International Workshop on Software Engineering for Science.* doi:https://doi.org/10.1145/3194747.3194750

Lowry, P. B., Gaskin, J. E., Twyman, N. W., Hammer, B., & Roberts, T. L. (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. doi:10.17705/1jais.00347

Lu, Y., Mao, X., Li, Z., Zhang, Y., Wang, T., & Yin, G. (2016). Does the role matter? An investigation of the code quality of casual contributors in GitHub. *2016 23rd Asia-Pacific Software Engineering Conference*, 49-56. doi:10.1109/APSEC.2016.44

Lund Research Ltd. (2018a). *Moderator analysis (dichotomous moderator variable).* Laerd statistics. https://statistics.laerd.com/premium/spss/mcd/moderator-continuous-dichotomous-in-spss.php

Lund Research Ltd. (2018b). *Ordinal regression using SPSS Statistics*. Laerd statistics. https://statistics.laerd.com/spss-tutorials/ordinal-regression-using-spss-statistics.php

Lund Research Ltd. (2018c). *Spearman's rank-order correlation*. Laerd statistics. https://statistics.laerd.com/statistical-guides/spearmans-rank-order-correlation-statistical-guide.php

Lund Research Ltd. (2018d). *Spearman's rank-order correlation using SPSS Statistics*. Laerd statistics. https://statistics.laerd.com/spss-tutorials/spearmans-rank-order-correlation-using-spss-statistics.php

Lund Research Ltd. (2018e). *Transforming data in SPSS Statistics*. Laerd statistics. https://statistics.laerd.com/spss-tutorials/transforming-data-in-spss-statistics.php

Majchrzak, T. A. (2010). Best practices for the organizational implementation of software testing. *Proceedings of the 43rd Hawaii International Conference on System Sciences*. doi:10.1109/HICSS.2010.83

Méndez, M., Tinetti, F. G., & Overbey, J. L. (2014). Climate models: Challenges for Fortran development tools. *2014 Second International Workshop on Software Engineering for High Performance Computing in Computational Science and Engineering*. doi:10.1109/SE-HPCCSE.2014.7

Méndez, M., & Tinetti, F. G. (2017). Change-driven development for scientific software. *The Journal of Supercomputing: An International Journal of High-Performance Computer Design, Analysis, and Use*, *73*(5), 2229-2257. doi:10.1007/s11227-017-1966-1

Mesh, E. S., Burns, G., & Hawker, J. S. (2014). Leveraging expertise to support scientific software process improvement decisions. *Computing in Science & Engineering*, *16*(3), 28-34. doi:10.1109/MCSE.2014.10

Mesh, E. S., Tolar, D. M., & Hawker, J. S. (2016). Exploring process improvement decisions to support a rapidly evolving developer base. *2016 IEEE/ACM 38th IEEE International Conference on Software Engineering Companion*, 777-780. http://dx.doi.org/10.1145/2889160.2889209

Miller, G. (2006). A scientist's nightmare: Software problem leads to five retractions. *Science*, *314*(5807). doi:10.1126/science.314.5807.1856

Mockus, A. (2010). Organizational volatility and its effects on software defects. *Foundations of Software Engineering*, 117-126. doi:10.1145/1882291.1882311

Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *11*(3), 309-346. doi:10.1145/567793.567795

Moore, G. C., & Benbasat, I. (1991). Development of an instrument to measure the perceptions of adopting an information technology innovation. *Information Systems Research*, *2*(3), 192-222. https://doi.org/10.1287/isre.2.3.192

Morris, C., & Segal, J. (2009). Some challenges facing scientific software developers: The case of molecular biology. *2009 Fifth IEEE International Conference on e-Science*, 216-222. doi:10.1109/e-Science.2009.38

Morris, M. G., & Venkatesh, V. (2000). Age differences in technology adoption decisions: Implications for a changing work force. *Personnel Psychology*, *53*(2), 375-403. https://doi.org/10.1111/j.1744-6570.2000.tb00206.x

Murphy, C., Shen, K., & Kaiser, G. (2009). Automatic system testing of programs without test oracles. *Proceedings of the Eighteenth International Symposium on Software Testing and Analysis (ISSTA '09)*, 189-200. doi:10.1145/1572272.1572295

Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., & Ye, Y. (2002). Evolution patterns of open-source software systems and communities. *Principles of Software Evolution*, 76-85. doi:10.1145/512035.512055

Nanthaamornphong, A., & Carver, J. C. (2018). Test-driven development in HPC science: A case study. *Computing in Science & Engineering*, *20*(5), 98-113. doi:10.1109/MCSE.2018.05329819

Ober, I., & Ober, I. (2017). On patterns of multi-domain interaction for scientific software development focused on separation of concerns. *Procedia Computer Science*, *108*, 2298-2302. doi:10.1016/j.procs.2017.05.288

Oluwajana, D., Idowu, A., Nat, M., Vanduhe, V., & Fadiya, S. (2019). The adoption of students' hedonic motivation system model to gamified learning environment. *Journal of*

Theoretical and Applied Electronic Commerce Research, *14*(3), 156-167. doi:10.4067/S0718-18762019000300109

Oxford University Press. (2020). Proportionate stratified sampling. In *Oxford reference*. https://www.oxfordreference.com/view/10.1093/oi/authority.20110803100349910

Phillips, A. W., Reddy, S., & Durning, S. J. (2016). Improving response rates and evaluating nonresponse bias in surveys. *Medical Teacher*, *38*, 217-228. doi:10.3109/0142159X.2015.1105945

Przedzinski, T., Malawski, M., Was, Z., Carver, J., & Rouson, D. (2020). Software development strategies for high-energy physics simulations based on quantum field theory. *Computing in Science & Engineering*, *22*(4), 86-98. doi:10.1109/MCSE.2019.2947017

Qualtrics. (2020). *Security statement*. https://www.qualtrics.com/security-statement/

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

Remmel, H. (2014). *Supporting the quality assurance of a scientific framework*. [Doctoral dissertation, University of Heidelberg]. heiDOK.

Riesch, M., Nguyen, T. D., & Jirauschek, C. (2020). Bertha: Project skeleton for scientific software. *PLoS ONE*, *15*(3). https://doi.org/10.1371/journal.pone.0230557

Rilee, M., & Clune, T. (2014). Test-driven development in HPC science: A case study. *Computing in Science & Engineering*, *20*(5), 98-113. doi:10.1109/MCSE.2018.05329819

Robles, G., Koch, S., & González-Barahona, J. M. (2004). Remote analysis and measurement of libre software systems by means of the CVSAnalY tool. *26th International Conference on Software Engineering (ICSE 2004)*. doi:10.1049/ic:20040351

Rogers, E. (1995). *Diffusion of innovations*. Free Press.

Russell, S., Bennett, T. D., & Ghosh, D. (2019). Software engineering principles to improve quality and performance of R software. *PeerJ Computer Science*, *5*. doi:10.7717/peerj-cs.175

Saddler, J. A., & Cohen, M. B. (2017). EventFlowSlicer: A tool for generating realistic goal-driven GUI tests. *Automated Software Engineering*, 955-960. doi:10.1109/ASE.2017.8115711

Salkind, N. J. (2010). *Encyclopedia of research design* (Vol. 1). SAGE Publications, Inc.

Sanders, R., & Kelly, D. (2008). Dealing with risk in scientific software development. *IEEE Software*, *25*(4), 21-28. doi:10.1109/MS.2008.84

Saphira, M., & Rusli, A. (2019). Towards a gamified support tool for requirements gathering in Bahasa Indonesia. *2019 5th International Conference on New Media Studies*. doi:10.1109/CONMEDIA46929.2019.8981828

Schwittek, W., & Eicker, S. (2013). A study on third party component reuse in Java enterprise open source software. *Component-Based Software Engineering*, 75-80. doi:10.1145/2465449.2465468

Segal, J. (2008a). Models of scientific software development. *Workshop on Software Engineering in Computational Science and Engineering (SECSE 08)*. Retrieved from http://oro.open.ac.uk/17673/1/SegalICSE08R.pdf

Segal, J. (2008b). Scientists and software engineers: A tale of two cultures. *Proceedings of the Psychology of Programming Interest Group (PPIG 08)*, 44-51. Retrieved from https://oro.open.ac.uk/17671/1/PPIG_08Segal.pdf

Segal, J. (2009). Some challenges facing software engineers developing software for scientists. *2nd International Software Engineering for Computational Scientists and Engineers Workshop (SECSE '09)*, 9-14. doi:10.1109/SECSE.2009.5069156

Segars, A. H., & Grover, V. (1993). Re-examining perceived ease of use and usefulness: A confirmatory factor analysis. *MIS Quarterly*, *17*(4), 517-525. doi:10.2307/249590

Segura, S., Durán, A., Troya, J., & Ruiz-Cortés, A. (2017). A template–based approach to describing metamorphic relations. *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*, 3-9. doi:10.1109/MET.2017.3

Segura, S., Fraser, G., Sanchez, A. B., & Ruiz-Cortés, A. (2016). A survey on metamorphic testing. *IEEE Transactions on Software Engineering*, *42*(9), 805-824. doi: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. *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*. doi:10.1109/RCIS.2014.6861031

Setiawan, S. S., & Suryadibrata, A. (2019). Fitrust: Promoting healthy lifestyle through gamified mobile health application. *2019 5th International Conference on New Media Studies*. doi:10.1109/CONMEDIA46929.2019.8981840

Shahri, M. P., Srinivasan, M., Reynolds, G., Bimczok, D., Kahanda, I., & Kanewala, U. (2019). Metamorphic testing for quality assurance of protein function prediction tools. *2019*

IEEE International Conference on Artificial Intelligence Testing (AITest), 140-148. doi:10.1109/AITest.2019.00017

Shankar, A., & Kumari, P. (2019). A study of factors affecting mobile governance (mGov) adoption intention in India using an extension of the technology acceptance model (TAM). *South Asian Journal of Management*, *26*(4), 71-94. Retrieved from https://search.proquest.com/openview/4f5a1fa73bc3c28d59bf8cc47afcf989/1?pq-origsite=gscholar&cbl=46967

Shields, P. M., & Rangarajan, N. (2013). *A playbook for research methods: Integrating conceptual frameworks and project management*. New Forums Press, Inc.

Silic, M., & Lowry, P. B. (2020). Using design-science based gamification to improve organizational security training and compliance. *Journal of Management Information Systems*, *37*(1), 129-161. https://doi.org/10.1080/07421222.2020.1705512

Smith, S., Jegatheesan, T., & Kelly, D. (2016). Advantages, disadvantages, and misunderstandings about document driven design for scientific software. *2016 Fourth International Workshop on Software Engineering for High Performance Computing in Computational Science and Engineering,* 41-48. doi:10.1109/SE-HPCCSE.2016.10

Steinmacher, I., Chaves, A. P., Conte, T., & Gerosa, M. A. (2014). Preliminary empirical identification of barriers faced by newcomers to open source software projects. *2014 Brazilian Symposium on Software Engineering (SBES)*, 51-60. doi:10.1109/SBES.2014.9

Steinmacher, I., Conte, T., Gerosa, M. A., & Redmiles, D. (2015). Social barriers faced by newcomers placing their first contribution in open source software projects. *Computer Supported Cooperative Work & Social Computing*, 1379-1392. doi:10.1145/2675133.2675215

Steinmacher, I., Pinto, G., Wiese, I. S., & Gerosa, M. A. (2018). Almost there: A study on quasi-contributors in open source software projects. *Software Engineering*, 256-266. doi:10.1145/3180155.3180208

Storer, T. (2017). Bridging the chasm: A survey of software engineering practice in scientific programming. *ACM Computing Surveys*, *50*(4). doi:10.1145/3084225

Straub, D., Limayem, M., & Karahanna-Evaristo, E. (1995). Measuring system usage: Implications for IS theory testing. *Management Science*, *41*(8), 1328-1342. doi:10.1287/mnsc.41.8.1328

Subramanian, G. H. (1994). A replication of perceived usefulness and perceived ease of use measurement. *Decision Sciences*, *25*, 863-874. https://doi.org/10.1111/j.1540-5915.1994.tb01873.x

Swearngin, A., Cohen, M. B., John, B. E., & Bellamy, R. K. E. (2013). Human performance regression testing. *2013 35th International Conference on Software Engineering*, 152-161. doi:10.1109/ICSE.2013.6606561

Sweetser, P., & Wyeth, P. (2005). GameFlow: A model for evaluating player enjoyment in games. *ACM Computers in Entertainment*, *3*(3). https://doi.org/10.1145/1077246.1077253

Szajna, B. (1994). Software evaluation and choice: Predictive validation of the technology acceptance instrument. *MIS Quarterly*, *18*(3), 319-324. doi:10.2307/249621

Taylor, A. B., West, S. G., & Aiken, L. S. (2006). Loss of power in logistic, ordinal logistic, and probit regression when an outcome variable is coarsely categorized. *Educational and Psychological Measurement*, *66*(2), 228-239. doi:10.1177/0013164405278580

Taylor, S., & Todd, P. A. (1995). Assessing IT usage: The role of prior experience. *MIS Quarterly*, *19*(2), 561-570. doi:10.2307/249633

Thompson, R. L., Higgins, C. A., & Howell, J. M. (1991). Personal computing: Toward a conceptual model of utilization. *MIS Quarterly*, *15*(1), 124-143. doi:10.2307/249443

Toffola, L. D., Staicu, C.-A., & Pradel, M. (2017). Saying 'Hi!' is not enough: Mining inputs for effective test generation. *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. doi:10.1109/ASE.2017.8115617

Toronto, N., & McCarthy, J. (2014). Practically accurate floating-point math. *Computing in Science & Engineering*, *16*(4), 80-95. doi:10.1109/MCSE.2014.90

Turner, M., Kitchenham, B., Brereton, P., Charters, S., & Budgen, D. (2010). Does the technology acceptance model predict actual use? A systematic literature review. *Information and Software Technology*, *52*(5), 463-479. https://doi.org/10.1016/j.infsof.2009.11.005

Vagias, W. M. (2006). *Likert-type scale response anchors*. Clemson University. http://media.clemson.edu/cbshs/prtm/research/resources-for-research-page-2/Vagias-Likert-Type-Scale-Response-Anchors.pdf

Van der Heijden, H. (2004). User acceptance of hedonic information systems. *MIS Quarterly*, *28*(4), 695-704. doi:10.2307/25148660

Venkatesh, V., & Bala, H. (2008). Technology acceptance model 3 and a research agenda on interventions. *Decision Sciences*, *39*(2), 273-315. https://doi.org/10.1111/j.1540-5915.2008.00192.x

Venkatesh, V., Brown, S. A., Maruping, L. M., & Bala, H. (2008). Predicting different conceptualizations of system use: The competing roles of behavioral intention,

facilitating conditions, and behavioral expectation. *MIS Quarterly*, *32*(3), 483-502. doi:10.2307/25148853

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

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: Toward a unified view. *MIS Quarterly*, *27*(3), 425-478. doi:10.2307/30036540

Venkatesh, V., & Speier, C. (1999). Computer technology training in the workplace: A longitudinal investigation of the effect of the mood. *Organizational Behavior and Human Decision Processes*, *79*(1), 1-28. doi:10.1006/obhd.1999.2837

Venkatesh, V., Thong, J. Y. L., & Xu, X. (2012). Consumer acceptance and use of information technology: Extending the unified theory of acceptance and use of technology. *MIS Quarterly*, *36*(1), 157-178. doi:10.2307/41410412

Venkatesh, V., Thong, J. Y. L., & Xu, X. (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. doi:10.17705/1jais.00428

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*, 713-726. doi:10.1007/s10209-015-0443-y

Wu, J., & Du, H. (2012). Toward a better understanding of behavioral intention and system usage constructs. *European Journal of Information Systems*, *21*, 680-698. doi:10.1057/ejis.2012.15

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). doi:10.1360/jos161210

Xiang, Z., & Wu, W. (2018). The willingness to use the campus express delivery service based on UTAUT. *2018 5th International Conference on Industrial Economics System and Industrial Security Engineering (IEIS)*. doi: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. *Principles of Software Evolution*, 46-55. doi:10.1145/2804360.2804366

Yang, B., Rousseau, R., Wang, X., & Huang, S. (2018). How important is scientific software in bioinformatics research? A comparative study between international and Chinese research communities. *Journal of the Association for Information Science and Technology*, *69*(9), 1122-1133. doi:10.1002/asi

Yousafzai, S. Y., Foxall, G. R., & Pallister, J. G. (2007). Technology acceptance: A meta-analysis of the TAM: Part 1. *Journal of Modelling in Management*, *2*(3), 251-280. doi:10.1108/17465660710834453

Zaimi, A., Ampatzoglou, A., Triantafyllidou, N., Chatzigeorgiou, A., Mavridis, A., Chaikalis, T., Deligiannis, I., Sfetsos, P., & Stamelos, I. (2015). An empirical study on the reuse of third-party libraries in open-source software development. *Informatics Conference*, 1-8. doi:10.1145/2801081.2801087

Zhang, X. (2009). *Understanding conflict between developers and testers in software development: Sources and impact* (Doctoral dissertation). Retrieved from ProQuest. (UMI No. 3400169).

Zheng, Y., Zhang, X., & Ganesh, V. (2013). Z3-str: A z3-based string solver for Web application analysis. *Foundations of Software Engineering*, 114-124. doi:10.1145/2491411.2491456

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. doi:10.1109/TSE.2015.2478001

Zumbo, B. D., & Zimmerman, D. W. (1993). Is the selection of statistical methods governed by level of measurement? *Canadian Psychology*, *34*(4), 390-400. doi:10.1037/h0078865

Appendix A

Research Instrument

Default Question Block

Q1.

Consent Letter

Introduction

My name is Brittany Hoard. I am a doctoral student at Northcentral University and am conducting a research study to look at the acceptance and use of metamorphic testing (MT) among open-source software developers, as well as factors that may be associated with MT use and acceptance. The name of this research study is “Metamorphic Testing Among Open-Source Software Developers: A Quantitative Correlational Study”. I am seeking your consent to participate in this study. Your participation is completely voluntary, and I am here to address your questions or concerns at any point during the study.

Eligibility

You are eligible to participate in this research if you:

1. Are at least 18 years old
2. Speak English fluently
3. Have contributed to a project in the GitHub "Software in science" collection

I hope to include 84 people in this research.

Activities

In this study, participants will:

1. Take an online survey for approximately 5 minutes from their computer or mobile device

Please note that activities are **optional**, and participants can choose which activities they do or do not want to participate in.

Participants will be asked demographic questions about their age and gender. These questions are **optional**, and participants may skip any question at any time.

Risks

Some possible risks include minor discomforts related to taking an online survey.

To decrease the impact of these risks, you can stop participation at any time by closing your Web browser.

Benefits

If you participate, there are no direct benefits to you.

This research may increase the body of knowledge in the subject area of this study.

Privacy and Data Protection

I will secure your information with these steps:

1. Research data will be anonymized.
2. Research data will be grouped in any publications or presentations.
3. Research data will be stored on a password-protected laptop computer accessible only to the researcher.

This data could be used for future research studies or distributed to other investigators for future research studies without additional informed consent from you or your legally authorized representative.

I will securely store your data for 3 years. Then, I will delete electronic data and destroy paper data.

How the Results Will Be Used

This data will be published in a doctoral dissertation. The data may be published in other scholarly works or public presentations. All data will be grouped. Participants will not be able to be identified in the data findings.

Contact Information

If you have questions, you can contact me at: B.Hoard0115@o365.ncu.edu or 1-703-881-6989.

My dissertation chair's name is Dr. Milton Kabia. They work at Northcentral University and is supervising me on the research. You can contact them at: mkabia@ncu.edu or 1-907-351-5825.

If you have questions about your rights in the research or if a problem or injury has occurred during your participation, please contact the NCU Institutional Review Board at irb@ncu.edu or 1-888-327-2877 ext 8014.

Voluntary Participation

If you decide not to participate, or if you stop participation after you start, there will be no penalty to you: you will not lose any benefit to which you are otherwise entitled.

- ◯ Yes, I agree, begin the survey
- ◯ No, I don't agree, I do not wish to participate

Q2. The following items pertain to the metamorphic testing method. Please select the response that best represents your perception.

	Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree

	Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
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.	○	○	○	○	○	○	○

Q3. The following items pertain to the metamorphic testing method. Please select the response that best represents your perception.

	Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
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.	○	○	○	○	○	○	○

Q4. The following items pertain to the metamorphic testing method. Please select the response that best represents your perception.

	Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
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.	○	○	○	○	○	○	○

Q5. Please choose the answer that best represents how often you have used the metamorphic testing method.

- ○ Never
- ○ Rarely, in less than 10% of the chances when I could have
- ○ Occasionally, in about 30% of the chances when I could have
- ○ Sometimes, in about 50% of the chances when I could have
- ○ Frequently, in about 70% of the chances when I could have
- ○ Usually, in about 90% of the chances when I could have
- ○ Every time

Q6. Please choose your level of experience with the metamorphic testing method.

- ○ Less than 6 months
- ○ 6 months to less than 12 months
- ○ 1 to 3 years
- ○ 4 to 6 years
- ○ 7 years or more

Q7. Please enter your age in years. (Leave this field blank if you prefer not to answer.)

Q8. Please select your gender. (Select "Prefer not to answer" if you prefer not to answer.)

- ○ Male
- ○ Female
- ○ Other
- ○ Prefer not to answer

Appendix B

IRB Approval Letter

11355 N. Torrey Pines Road
La Jolla, CA 92037

Date: September 24, 2020
PI Name: Brittany Hoard
Chair Name (if applicable): Milton Kabia
Application Type: Initial Submission
Review Level: Exempt - Category 2
Study Title: Metamorphic Testing Among Open-Source Software Developers: A Quantitative Correlational Study

Approval Date: September 24, 2020

Dear Brittany:

Congratulations! Your IRB application has been approved. Your responsibilities include the following:

1. Follow the protocol as approved. If you need to make changes with your population, recruitment, or consent, please submit a modification form. Remember that we have office hours and group writing sessions to support you.
2. If there is a consent process in your research, you must use the consent form approved with your final application. Please make sure all participants receive a copy of the consent form.
3. **If there are any injuries, problems, or complaints from participants (adverse events), you must notify the IRB at IRB@ncu.edu within 24 hours.**
4. IRB audit of procedures may occur. The IRB will notify you if your study will be audited.
5. When data are collected and de-identified, please submit a study closure form to the IRB. See the IRBManager instructions on our website.
6. You must maintain current CITI certification until you have submitted a study closure form.
7. If you are a student, please be aware that you must be enrolled in an active dissertation course with NCU in order to collect data.

Best wishes as you conduct your research!

Respectfully,

Northcentral University Institutional Review Board
Email: irb@ncu.edu

Appendix C

Permission to Use Research Instrument

Copyright Clearance Center Marketplace™

M I S Quarterly - License Terms and Conditions

This is a License Agreement between Brittany Hoard ("You") and M I S Quarterly ("Publisher") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by M I S Quarterly, and the CCC terms and conditions.

All payments must be made in full to CCC.

Order Date	15-Aug-2020	Type of Use	Republish in a thesis/dissertation
Order license ID	1055790-1		
ISSN	0276-7783	Publisher	Society for Management Information Systems and Management Information Systems Research Center of the University of Minnesota
		Portion	Chart/graph/table/figure

LICENSED CONTENT

Publication Title	MIS quarterly	Country	United States of America
Author/Editor	Society for Information Management (U.S.), University of Minnesota. Management Information Systems Research Center	Rightsholder	M I S Quarterly
		Publication Type	e-Journal
		URL	http://www.misq.org
Date	12/31/1983		
Language	English		

REQUEST DETAILS

Portion Type	Chart/graph/table/figure	Distribution	Worldwide
Number of charts / graphs / tables / figures requested	1	Translation	Original language of publication
Format (select all that apply)	Print, Electronic	Copies for the disabled?	No
		Minor editing privileges?	No
Who will republish the content?	Academic institution	Incidental promotional use?	No
Duration of Use	Current edition and up to 5 years	Currency	USD
Lifetime Unit Quantity	Up to 499		
Rights Requested	Main product		

NEW WORK DETAILS

Title	Metamorphic Testing Among Open-Source Software Developers: A Quantitative Correlational Study		
		Instructor name	Milton Kabia
Institution name	Northcentral University		

ADDITIONAL DETAILS

Order reference number	N/A	The requesting person / organization to appear on the license	Brittany Hoard

REUSE CONTENT DETAILS

Title, description or numeric reference of the portion(s)	Table 16	Title of the article/chapter the portion is from	USER ACCEPTANCE OF INFORMATION TECHNOLOGY: TOWARD A UNIFIED VIEW
Editor of portion(s)	Cynthia Beath		
Volume of serial or monograph	27	Author of portion(s)	Society for Information Management (U.S.); University of Minnesota. Management Information Systems Research Center
Page or page range of portion	460		
		Issue, if republishing an article from a serial	3
Publication date of portion	2003-09-01		