2018

Relationship Between Software Development Team Structure, Ambiguity, Volatility, and Project Failure

Dominic Martinelli Saxton
Walden University

Walden University

College of Management and Technology

This is to certify that the doctoral study by

Dominic Martinelli Saxton

has been found to be complete and satisfactory in all respects,
and that any and all revisions required by
the review committee have been made.

Review Committee
Dr. Scott Burrus, Committee Chairperson, Doctor of Business Administration Faculty

Dr. Jaime Klein, Committee Member, Doctor of Business Administration Faculty

Dr. Reginald Taylor, University Reviewer, Doctor of Business Administration Faculty

Chief Academic Officer
Eric Riedel, Ph.D.

Walden University
2018

Abstract

Relationship Between Software Development Team Structure, Ambiguity, Volatility, and Project Failure

by

Dominic Martinelli Saxton

MS, Troy State University, 1999

BS, Tusculum College, 1995

Doctoral Study Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Business Administration

Walden University

December 2018

Abstract

Complex environments like the United States Air Force's advanced weapon systems are highly reliant on externally developed software, which is often delivered late, over budget, and with fewer benefits than expected. Grounded in Galbraith's organizational information processing theory, the purpose of this correlational study was to examine the relationship between software development team structure, ambiguity, volatility and software project failure. Participants included 23 members of the Armed Forces Communications and Electronics Association in the southeastern United States who completed 4 project management surveys. Results of multiple regression analysis indicated the model as a whole was able to predict software project failure, $F(3,19) = 10.838$, $p < .001$, $R^2 = 0.631$. Software development team structure was the only statistically significant predictor, $t = 2.762$, $p = .012$. Implications for positive social change include the potential for software development company owners and military leaders to understand the factors that influence software project success and to develop strategies to enhance software development team structure.

Relationship Between Software Development Team Structure, Ambiguity, Volatility, and Project Failure

by

Dominic Martinelli Saxton

MS, Troy State University, 1999

BS, Tusculum College, 1995

Doctoral Study Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Business Administration

Walden University

December 2018

Dedication

This study is dedicated to my loving wife, Faye, for her endurance, sacrifice, and support that has allowed me to complete a dream. Also, my children, Dominique and Dominic Jr., thank you for never complaining over the years about having to share my attention with my career and academic pursuits without their consent. I would also like to dedicate this work to my mother and father, Raymon and Mary, for their love and for giving me an unstoppable work ethic. I thank you and love you all.

Acknowledgments

I would like first to thank God for giving me the strength, perseverance, and resolution to take on and complete this endeavor. I would also like to acknowledge and thank all my instructors at Walden University who have encouraged and supported me throughout this program. Specifically, I thank Dr. Scott Burrus, chair, and Dr. Jaime Klein, committee member, for guiding me through the Walden process. I would also like to thank Dr. Reginald Taylor, university research reviewer, for his guidance and patience. To Dr. Zin Htway, thank you for providing me the statistical guidance, patience, and encouraging words needed when I was at a point of frustration and uncertainty.

Thank you to my wife, Faye E. Saxton, for giving me the support and time to complete my study. Thank you to my daughter, Dominique, and my son, Dominic Jr., for secretly giving me the strength to continue when I was on the verge of quitting the program. Thank you to all my family and friends for being there throughout this process; without your guidance, patience, and support this process would not have been possible. I also appreciate the many others not mentioned by name for the contributions made in their respective ways. Thank you all.

Table of Contents

List of Tables

List of Figures

Section 1: Foundation of the Study

U.S. Air Force (USAF) business systems, national security systems, and weapon systems are highly reliant on software, which underlies the system. As personnel in the USAF fluctuate, the levels of software development expertise fluctuate. To manage the changes in expertise, the USAF outsources most software project development to commercial software development companies. Although the USAF has outsourced the development of software to external companies, the software is still susceptible to schedule slips, cost overruns, resource misallocations, and cancellations (Hagen & Sorenson, 2013). Although commercial systems are not designed to defend the country, these systems remain susceptible to the same levels of software project success or failure.

Background of the Problem

Software affects businesses directly through an employed software application or indirectly through customer or supplier interfaces (Sabbineni & Rao, 2015). Approximately 40% of software development projects worldwide have an on-time delivery within the prescribed budget (Eberendu, 2015). Approximately 60% of worldwide software development projects do not meet the traditional definition of project success, which is a project completed within schedule, within budget, and with all specifications (Eberendu, 2015). Kannan, Mahalakshmi, and Sujatha (2014) stated that incomplete requirements, lack of user involvement, lack of resources, and lack of information technology management are factors contributing to software development projects' failure. Leveson (2013) listed lack of documentation, lack of code verification, lack of clearly defined and managed requirements, and sacrificing quality to meet schedules as factors contributing to software development projects' failure.

The definition of software project success has evolved from a traditional triad definition to a multifaceted definition with objective and subject factors (McLeod, Doolin, & MacDonell, 2012). An earlier definition of software project success involved specifications, schedule, and budget; however, current definitions include factors such as scope, client satisfaction, and team satisfaction (da Silva et al., 2013). The evolution of the definition of project success has increased researchers' interest in why software development projects fail (Müller & Jugdev, 2012). However, few studies have addressed the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

Problem Statement

Complex environments like the USAF advanced weapon systems are highly reliant on externally developed software, which researchers have noted often is delivered late, over budget, and with fewer benefits than expected (Locatelli, Mancini, & Romano, 2014). From 2008 to 2012, the USAF reported $5 billion in mishap losses and 9000 lost workdays mostly attributed to software failure somewhere in the system (Foreman, Favaró, Saleh, & Johnson, 2015). The general business problem was software project failure negatively impacts the technological advances, financial stability, and operational viability of a company. The specific business problem was some software development company owners contracted by the USAF do not know the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

Purpose Statement

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. The three independent variables were software development team structure, ambiguity, and volatility in software specifications. The dependent variable was software project failure. The target population was software development companies contracted by the USAF who were members of the Armed Forces Communications and Electronics Association (AFCEA) in the Southeastern United States geographical area. The contributions to positive social change included increasing software success, reducing tax dollar waste, and increasing the public’s trust in national defense.

Nature of the Study

I used quantitative methodology for this study. Researchers use the quantitative method to test relationships and differences among variables to test hypotheses (O’Leary, 2017). A quantitative methodology was appropriate for this study because the goal was to test the relationships between independent and dependent variables. Qualitative methodology was not appropriate for this particular study because researchers use the qualitative method to provide a thick description of a phenomenon (O’Leary, 2017). The mixed-methods approach requires researchers to employ both quantitative and qualitative methods (O’Leary, 2017). Because the purpose of this study was not to identify themes emerging from exploration of a phenomenon, the mixed-methods approach was not appropriate.

Researchers use a correlational design to examine relationships to predict or explain variation between independent and dependent variables (O'Leary, 2017). Researchers use a descriptive research design to describe a variable systematically to explain a phenomenon (O'Leary, 2017). The intent of this study was to examine the relationship between independent and dependent variables; therefore, a descriptive design was not suitable for this study. Researchers use quasi-experimental and experimental designs to control or randomize the treatment when examining potential cause-effect relationships (O'Leary, 2017). I determined that the topic and settings of this correlational study did not require random assignment of treatment to determine cause-effect relationships among variables. Therefore, quasi-experimental and experimental designs were not suitable for this study.

Research Question and Hypotheses

The overarching research question was the following: What is the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure?

H_0: There is no relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

H_a: There is a relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

Theoretical Framework

The organizational information processing (OIP) theory was the theoretical framework in this study. Galbraith developed the OIP theory in 1973 (Galbraith, 1973). The constructs of OIP theory include information-processing needs, information-processing capability, and interoperability between information-processing needs and information-processing capability (Galbraith, 1973). Common measurements of the OIP constructs are ambiguity and volatility (Galbraith, 1974). Galbraith (1973) defined *ambiguity* as users having different frames of references, which creates multiple perspectives. Galbraith (1973) stated that volatility reflects changes occurring over time. Quality information provides organizational leaders with the skills to enhance their decision-making and the capacity to address environmental uncertainty (Galbraith, 1974). Organizational leaders implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty (Galbraith, 1973).

Operational Definition

In this study, I used technical terms that were relevant to the examination of the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure in companies that developed software for the USAF. Operational definitions include the following:

Software project success: Software project success is a software development project completed within schedule, within budget, meeting all specifications, and with the proper team composition and client satisfaction (da Silva et al., 2013).

Assumptions, Limitations, and Delimitations

Assumptions constrained the scope of this study, which I assumed were true but could not verify. Limitations indicated known weaknesses of the study. Delimitations indicated the boundaries of the study.

Assumptions

Ellis and Levy (2010) defined *assumptions* as claims accepted as true within a study, but without evidence. An assumption in this study was that all respondents would answer survey questions candidly and honestly. A second assumption was that participants would respond based on their experiences.

Limitations

Limitations are factors that are beyond the researcher’s control and can affect the outcome of the study or create improper interpretations of the results (Ellis & Levy, 2010). A limitation of this study was the information obtained through the survey was accurate at the time but did not account for possible future changes. Another limitation was the quantitative findings reflected a narrow view of occurrences or circumstances and did not include the depth of understanding provided by a qualitative study.

Delimitations

Delimitations are boundaries the researcher has chosen to narrow the focus of the study (O’Leary, 2017). A delimitation of this study was the focus on USAF software development companies, which were members of the Armed Forces Communications and Electronics Association (AFCEA) in the Southeastern United States. The AFCEA membership in the Southeastern United States included a small group of software development companies that specialized in developing, modernizing, and sustaining Defense Business Systems. Readers should take into consideration the delimitations of this study and exercise caution in interpreting and generalizing findings. However, this study focused on aspects of the system development process that were common across geographical areas and other business settings.

Significance of the Study

The significance of the study included three elements. The first element was how the findings would be valuable to software development companies that work for the USAF. The second element was how the results may contribute to the improvement of software development. Lastly, findings may contribute to positive social change in broader ways.

Contribution to Business Practice

The findings of this doctoral study may contribute to businesses by improving the way business owners organize software development teams to enhance the teams' decision-making and capacity to cope with environmental uncertainty, thereby increasing software projects' success rates. Three common measures researchers employed to determine the success of software programs were (a) software delivered on time, (b) software costs were within budget, and (c) software worked as intended (Kaur & Sengupta, 2013). The study of software development team structure and the effects on software project failure may be significant to the U.S. Government, foreign and domestic businesses, and foreign governments because software has become prevalent and software applications constitute a significant portion of total development expenses (see

Sabbineni & Rao, 2015). For example, in December 2012, the USAF canceled a major logistics software program when customer-required changes to the logistics processes, tools, and languages became unmanageable, even after $1 billion was spent over 8 years of development (Hagen & Sorenson, 2013). With the dependence on software and the rising cost of software, a software project failure could be financially detrimental to businesses’ and the military’s viability (Kaur & Sengupta, 2013).

Implications for Social Change

The results of this study may contribute to positive social change by improving the success rates of USAF software projects to provide better weaponry for the protection of the country. Better weaponry may enable the U.S. warfighter to remain dominant on the battlefield while deterring other enemies and preventing hostile actions. An additional potential contribution to positive social change may be through the reduction of the tax burden to the U.S. citizen. The improvement of success rates of USAF software projects may reduce software project cancellations, thereby reducing government spending.

A Review of the Professional and Academic Literature

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. I reviewed the literature on the OIP theory, which was the theoretical framework for this study. A tenet of the OIP theory was that the information-processing capability of a team would match the levels of ambiguity and volatility in the information the team must process (Galbraith, 1973). When a team possesses quality information, there is low ambiguity and volatility of the information within that team (Galbraith, 1973). I included ambiguity and volatility as independent variables to determine whether these variables were related to the dependent variable of software project failure. Honoring the team's structure leads to activities that enhance team cohesiveness and improve team performance (Cheruvelil et al., 2014). I also included the independent variable of software development team structure.

This literature review consists of six sections: the purpose of the study; hypotheses; OIP theory; the software development team structure, ambiguity, and volatility independent variables; software project failure dependent variable; and the measurement of variables. The first section addresses the purpose of this quantitative correlational study, followed by a restatement of the hypotheses. The next section provides an in-depth analysis and synthesis of OIP theory literature, including other supporting and contrasting theories. The following section addresses the relevance of OIP theory through a critical analysis and synthesis of the independent and dependent variables. A review of the measurement of the independent and dependent variables follows the examination of the independent and dependent variables. The final section provides a comparison and contrast of differing points of view and the relationship to previous research and findings.

I searched the literature using electronic databases and other online materials, focusing on peer-reviewed journal articles. I searched the databases in the Walden University library, including EBSCO and ProQuest Library. I also used Google Scholar and searched other Internet sources. I used key words and Boolean parameters to search for relevant literature, including *software development*, *organizational information* *processing theory*, *OIP theory*, *software development team structure*, *ambiguity*, *volatility*, *ambiguity in software development*, *volatility in software development*, *software project success definition*, *software project success factors*, *software ambiguity*, and *software volatility*. These literature searches yielded 1,546 references. I identified 78 relevant references from the search results and included them in this literature review. These 78 references included 67 peer-reviewed articles. Sixty-nine references had a publication date between 2014 and 2017, which ensured that a minimum of 85% of the references were peer-reviewed and published within 5 years of the anticipated completion of the study.

Purpose of the Study

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure in software development companies that developed software for the USAF. This quantitative correlational study included three independent variables: team structure, ambiguity, and volatility of SW specifications. The dependent variable was software project failure. The target population was software development company owners contracted by the USAF. I conducted a simple random sample of software development company owners contracted by the USAF who were members of the Armed Forces Communications and Electronics Association (AFCEA) and were located in the Southeastern United States. AFCEA membership included software company owners who provided software development for the military, the federal government, and state governments. Contributions to positive social change include increasing software success, reducing tax dollar waste, and increasing the public’s trust in national defense.

Hypotheses

H_0: There is no relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

H_a: There is a relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

OIP Theory

Software development companies rely on information to eliminate confusion, improve productivity, set team structure, and disseminate the company’s vision and goals. Company owners have a desire to achieve higher performance, and through understanding the link between information-processing requirements and information-processing capacities, they can position the company to obtain the goal of higher performance (Restuccia, Brentani, Legoux, & Ouellet, 2016). Galbraith developed the organizational information processing (OIP) theoretical framework, with the constructs of (a) information-processing needs, (b) information-processing capability, and (c) the interoperability between information-processing needs and information-processing capability (AlMarzouq, Grover, & Thatcher, 2015; Galbraith, 1973, 1974). Galbraith (1974) identified typical measurements of the OIP constructs as ambiguity and volatility. Quality information can eliminate ambiguity and volatility and can improve productivity to increase the chances of software project success (AlMarzouq et al., 2015).

Reducing the effects of ambiguity and volatility and the team structure can have a significant impact on the success of software development projects. Organizations implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty (AlMarzouq et al., 2015; Galbraith, 1973, 1974). OIP theory implies that organizations require more information-processing capacity when executing tasks involving more uncertainty and more levels of interdependence (Dutot, Bergeron, & Raymond, 2014; Srinivasan & Swink, 2015). Researchers suggested that extensive planning is necessary to reduce uncertainty and improve information flows and processing, especially in tasks that are interdependent such as software development projects.

Review of Independent Variables

The review of independent variables for this study contains a critical analysis and synthesis of literature pertaining to the variables software development team structure, ambiguity, and volatility.

Software development team structure. Software development team structure affects the team's success in software project development. Galbraith (1973) stated that team structure refers to how organizations implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty. Knowledge sharing in software development companies occurs through knowledge exchange and knowledge combination (Ghobadi, 2015). Social knowledge develops through relationships within close-knit organizational groups (Ghobadi, 2015). Organizations that have a hierarchical structure have a lower information-processing capability than organizations that have a flatter organizational structure (Galbraith, 1973). Software development companies with a loosely defined team structure employ less knowledge sharing, whereas companies with a well-defined team structure can process information more efficiently.

Uncertainty in software development teams often materializes because the team's structure does not promote an environment of information sharing. Uncertainty in an organization drives the need for more information to achieve higher organizational performance (Galbraith, 1973; Restuccia et al., 2016). One of the most common reasons why managers increase organizational complexity is to expand the division of labor in their organization (Galbraith, 1973; Ghobadi, 2015). Four strategies to improve organizational structure problems include the creation of slack resources, self-contained tasks, investment in vertical information systems, and the creation of lateral relations (Dutot et al., 2014). Teams often believe communications mean endless meetings, excessive emails, and the reading of other team member minds, rather than the sharing of relevant information.

Although Galbraith addressed team structure from the technical information-processing aspect, other authors suggested that management structure influences team structure. The importance of having a proper team structure applies to the teams working on information-intensive tasks such as software development teams (Açıkgöz & Günsel, 2016). Researchers have viewed the software development process primarily from a technical perspective, but the emerging view on software development process centers on the sociotechnical aspects of the process, indicating that organizational and human aspects play critical roles (Too & Weaver, 2014). Although the effectiveness of the team's ability to process information depends on the structure of that team, the overarching management structure also influences the team structure.

In waterfall development methodology, project managers lead software development teams. However, infrequently, functional managers can share the leader responsibilities or lead the team (Project Management Institute, 2017). One of the modern approaches of structuring software development teams is to use agile project management (Conforto, Salum, Amaral, da Silva, & de Almeida, 2014). Leadership, personnel innovation, and collaboration anchor the agile project management approach rather than leader command and control (Brhel, Meth, Maedche, & Werder, 2015; Conforto et al., 2014). Decentralized management is another distinguishing characteristic of the agile management approach, which contrasts with the autocratic, hierarchical management style of the traditional management approach (Brhel et al., 2015). Waterfall methodology requires more management involvement and more responsibilities placed on the project manager, while agile allows a flatter management structure.

Using the right methodological approach will assist software development leaders in defining the proper team structure for the highest chances of software project success. The depth and breadth of the software development team depend on the complexity of the project and the methodological approach of the project (Brhel et al., 2015; Liu, Kong, & Chen, 2015; Project Management Institute, 2017). Traditional, agile, extreme, and emertxe project management are four of the conventional approaches to software project development, with each approach requiring a different team structure (Liu et al., 2015; Project Management Institute, 2017). Complexity, scope clarity, and uncertainty determine the management approach best suited for successful software project development (Brhel et al., 2015; Conforto et al., 2014; Dikert, Paasivaara, & Lassenius, 2016; Liu et al., 2015). The organizational culture, location, and scope of the project can vary the software development team's composition (Drury-Grogan, 2014; Hoch & Kozlowski, 2014). Team structure is a multifaceted topic, with the organizational structure being one of the tenants heavily involved in the decision of which best suits the team structure.

Organizational structure has a substantial influence on the software development team's structure. Some of the typical organizational structures software development company leaders choose are functional, matrixed, and projectized (Mir & Pinnington, 2014; Project Management Institute, 2017). Within the functional, matrixed, and projectized structured organizations, a dedicated team, hybrid team, virtual team, or part-time team are optional substructures of the software development team (Fernandes, Ward, & Araújo, 2014; Mir & Pinnington, 2014; Project Management Institute, 2017). The project manager and functional manager's authorities and responsibilities can vary widely depending on the organizational context (Jiang, Chang, Chen, Wang, & Klein, 2014; Mir & Pinnington, 2014; Project Management Institute, 2017). In a traditional hierarchical organization and a strong matrix organization, project managers have the authority and responsibility of managing their teams, but the functional manager has more authority in a weak matrix organization (Fernandes et al., 2014; Project

Management Institute, 2017). The software development team's structure depends on the parent organization's structure so the project aligns with the organization's vision and goal.

The global reach of organizations creates a need for team diversity and different organizational structuring. A method for software project managers to ensure software project team diversity is to embrace a virtual team structure to allow the team to operate in multiple geographical locations (Hoch & Kozlowski, 2014; Project Management Institute, 2017). Many organizations combine several of the organizational structures into a composite organizational structure, depending on the timing, urgency, and complexity of the project (Project Management Institute, 2017; Stapel & Schneider, 2014). Virtual organizations incorporate video telecom capabilities, virtual meetings, and an assortment of IT tools that allow the program management office to operate in a virtual environment to reach a more capable talent pool and combine the diversity from multiple countries.

Ambiguity. Ambiguity in a software development project increases the confusion in requirements and multiple interpretations of the purpose and scope of the project. The degree to which various interpretations of the information exist in the specifications defines ambiguity (Giezen, Salet, & Bertolini, 2015; Martens & Van Weelden, 2014; Pelikan, Stikova, & Vrana, 2017). Reducing requirement ambiguity is an essential factor for project success (Colomo-Palacios, Casado-Lumbreras, Soto-Acosta, García-Peñalvo, & Tovar, 2014; Giezen et al., 2015). The only way for a software requirement to be unambiguous is for there to be only one interpretation (Colomo-Palacios et al., 2014; Martens & Van Weelden, 2014). However, an unambiguous requirement rarely occurs.

Ambiguity occurs in multiple dimensions within software requirements management. Researchers identified four dimensions of ambiguity: lexical, syntactic, semantic, and pragmatic (Christophe et al., 2014; Colomo-Palacios et al., 2014; Sajjadi, Rassouli, Abbaszadeh, Majd, & Zendehdel, 2014; Wang, Chen, & Chen, 2016). Lexical ambiguity occurs when a single word has multiple meanings. Syntactic ambiguity occurs when a requirement has multiple parts (Christophe et al., 2014; Colomo-Palacios et al., 2014). Semantic ambiguity occurs when a requirement has several meanings (Christophe et al., 2014; Colomo-Palacios et al., 2014). Lastly, pragmatic ambiguity occurs when a requirement has several context-dependent meanings (Colomo-Palacios et al., 2014; Sajjadi et al., 2014). To eliminate or reduce ambiguity in software requirement management, project managers must be aware of the multiple dimensions of ambiguity before they can increase the probabilities of software development project success.

Ambiguity within a software development project can lead to the failure of the project. The impact of ambiguity on the software development process can be cost overrun, delays, or project cancellation (Colomo-Palacios et al., 2014; Martens & Van Weelden, 2014; Van den Hoek, Brugnach, Mulder, & Hoekstra, 2014). The magnitude of the impacts determines whether the ambiguity is a contributing factor to software project success (Van den Hoek et al., 2014). Colomo-Palacios et al. (2014) found the levels of ambiguity in the requirements for a project did not correlate with the project's success. Colomo-Palacios et al. (2014) also reported ambiguity did not increase the number of defects. Although ambiguity in a software development project can lead to confusion within the project, researchers reveal certain levels of ambiguity may not affect software project success.

Ambiguity is dependent on the perspective of the individuals involved in the project. Ambiguity also refers to the frame of mind, which is relevant for an actor concerning a decision, issue, or event (Van den Hoek et al., 2014). Multiple actors may have multiple views despite having the same information, thus increasing the ambiguity in the project (Martens & Van Weelden, 2014; Van den Hoek et al., 2014). The effects of the high levels of ambiguity can cause indecisiveness or conflict (Colomo-Palacios et al., 2014; Van den Hoek et al., 2014). Van den Hoek et al. (2014) states two of the most critical aspects of ambiguity is the potential impact and relevance for the actors. An engineer and a project manager on the same project receive the same scope definition, but they can come away with separate meanings of that scope definition.

Ambiguity in a project does not automatically mean there is a negative against the project. Ambiguity in a project can be beneficial with early conflict confrontation (Martens & Van Weelden, 2014). Martens and Van Weelden (2014) state all large projects have a high degree of contested information. The high degree of contested information presents the characteristics of ambiguity (Martens & Van Weelden, 2014; Van den Hoek et al., 2014). Software developers expect unambiguous requirements, but customers often write requirements in a simplistic language to ease understanding; however, this simple language seldom leads to a singular interpretation (Colomo-Palacios et al., 2014; Van den Hoek et al., 2014). Thus, requirements written in a simplistic format, introduce ambiguity into the project (Colomo-Palacios et al., 2014). Ambiguities early in a project promote clarity in communications and therefore increase the agreed understanding as the project progresses to the critical stages.

At times software project developers view requirements ambiguity as inconsequential instead of releasing the slightest ambiguity can have a tremendous effect. When actors consistently address a particular ambiguity, it implies the ambiguity may have significant relevance and not a mere inconvenience (Van den Hoek et al., 2014). When a project has sufficient agreed information, there is little ambiguity, and therefore codification, storage, and transference of the software project can readily occur (Peng, Heim, & Mallick, 2014). The uniqueness of a project increases a lack of information about the potential markets and target customers, thus increasing the ambiguity of the project (Pelikan et al., 2017; Peng et al., 2014). New ideas are central causes in software project ambiguity (Gutiérrez, 2014). Ideas and decisions must be wholly defined to ensure the success of software project development (Gutiérrez, 2014). Ideas not fully understood or opposing opinions increase the ambiguity in a software development project.

Global changes and technological advancements impact software programs as the software becomes older; thus forcing the software programs to become convoluted. Customers value IT products with multiple components, and this project complexity increases the project ambiguity (Kandjani, Tavana, Bernus, Wen, & Mohtarami, 2015; Peng et al., 2014). With a complex project, there is a need to share vast amounts of information, therefore unwittingly increasing the ambiguity of the project (Kandjani et al., 2015; Peng et al., 2014). Collaboration, brainstorming, debate, and clarifying information among software project team members help to eliminate ambiguity (Peng et al., 2014; Yang, Lu, Yao, & Zhang, 2014). Through the development of ideas, project managers allow the software project team to understand the purpose and reveal benefits to the project, thus reducing project ambiguity (Gutiérrez, 2014; Yang et al., 2014). Although software programs continue to become more complex, continuous clarifying actions can reduce ambiguity.

The film industry also experiences ambiguity, because there are dual leaders in the director and producer, which at times conflict. The director is culpable for the artistic aspects of the film, and the producer is responsible for the commercial aspects of the film (Ebbers & Wijnberg, 2017). These dual leaders often do not have a clear idea of their boundaries, which tasks and responsibilities are part of their role, and often provide information, which conflicts with their roles (Ebbers & Wijnberg, 2017). To eliminate the role ambiguity between the director and the producer a rigid structure must be in place, with descriptions of all roles and only one superior per role (Ebbers & Wijnberg, 2017). Ambiguity in requirements and roles occur in various industries and if left unresolved can be detrimental to the success of the project.

Volatility. Requirements volatility in software projects occurs on a regular basis and without management of the volatility, the volatility can be detrimental to a software project. Volatility refers to the extent of changes occurring over time (Galbraith, 1973). Ramasubbu, Bharadwaj, and Kumar (2015) equate volatility as the degree and frequency of changes surrounding requirements. Volatility is a measure of uncontrolled changes in software specifications occurring throughout the software development lifecycle (Al-

Saiyd, 2016; Bayona-Oré, Calvo-Manzano, Cuevas, & San-Feliu, 2014; Islam, Mouratidis, & Weippl, 2014). Requirements volatility or scope creep can occur when changes in software requirements occur after an agreed scope determination.

Incomplete, erroneous, or inconsistent requirements create an environment for software project failure. Peña & Valerdi (2015) stated requirements volatility is a significant problem in software development projects, often resulting in project delays and cost overruns. Receiving reliable, complete, consistent, and high-quality requirements often are not a reality for software development projects and create a breeding environment for volatility in the project (Al-Saiyd, 2016; Islam et al., 2014). Uncontrolled requirements volatility can create unfavorable results in a software development project.

Volatility in a software development project can affect the success of the project, program, and organization. Experienced team leadership, software volatility, and environmental volatility have an impact on software project success (Bayona-Oré et al., 2014). Software companies must continually change to stay abreast of changing customer needs and technological advances (Al-Saiyd, 2016). Changing software components and the dynamic software company’s business environment causes volatility (Peña & Valerdi, 2015). Therefore, volatility is a constant part of a dynamic software company’s business climate because there is a continuous stream of improvements to the software products usability and value to the customer (Al-Saiyd, 2016). Consumers demand rapid software changes, which demands the software development companies respond quickly, often creating volatility in the project.

Volatility in a software project often results in cancellation of the project. Customers cancel or abandoned roughly 60% to 70% of the global software projects attempted (Al-Saiyd, 2016; Hornstein, 2015). Volatility in requirements can cause software project failures due to the ambiguous understanding of the things driving the needed changes and the consequences of these changes (Al-Saiyd, 2016; Ebad, 2017). Changing requirements during the software maintenance phase of a project is a costly and time-consumed endeavor because the software developer must ensure the new software adapts to the new environment, business, and end user’s needs (Al-Saiyd, 2016; Wickboldt et al., 2015). The four major factors creating requirement volatility are organizational factors, project factors, development process factors, and project stakeholder factors (Al-Saiyd, 2016). No matter the factors of volatility, unchecked volatility creates uncertainty in the project.

Volatility in a software development project reaches beyond the project and can influence the underlying infrastructure and the parent organization. Uncontrolled volatility affects software architecture in a negative way (Mehta, Hall, & Byrd, 2014). Requirements volatility not only affects a development project from the project management perspective but volatility also affects the development project from the software architecture design perspective (Ebad, 2017; Peña & Valerdi, 2015). Design stability is one of the requirements to eliminate software architecture volatility (Mehta et al., 2014). The quality of the software architecture is in direct correlation to the quality of the software project design; therefore, volatility in design stability has an impact on software architecture (Mehta et al., 2014). Although there is an association between project size, project performance, and volatility, the relationship is not linear (Bayona-Oré et al., 2014). Volatility in software design ultimately affects the stability of the final product.

Not all volatility in software development projects has negative consequences; volatility can have a positive effect as well. Ways to reduce volatility in software projects include; pair programming, code reviews, and automated testing (Verner, Brereton, Kitchenham, Turner, & Niazi, 2014). Depending on the level of volatility in the development process, a software development team could use one of the following common software approaches; traditional project management, agile project management, extreme project management, and emertxe project management to control the amount of volatility (Li, Lu, Kwak, & Dong, 2015; Marinho, Sampaio, Lima, & Moura, 2014; Liu et al., 2015). The level of certainty as it pertains to the goal and the final solution helps the software development team determine the best approach (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). The higher levels of certainty equate to lower levels of volatility (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). Software developers have developed several approaches, which can address volatility in software development.

A project manager's method of addressing varying levels of certainty assists the project manager in controlling volatility in the software requirements. With traditional project management, the software development team needs to have a very stable goal and solution (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). On the other end of the spectrum if there is much volatility in both the goal and the solution, then extreme project management is the preferred approach (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). In the extreme project management approach, volatility allows the greatest flexibility in development and outcome (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). The middle approaches between traditional project management and extreme project management are the agile project management and emertxe project management, where volatility is tolerated in with the goal axis or the solution axis but not both (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). The agile project management and emertxe project management approaches allow the customers to change their goals or solutions to fit their needs as the environment or technology changes (Li et al., 2015; Liu et al., 2015; Marinho et al., 2014). With every evolving levels of volatility, a project manager must have various choices to approach the software development project.

Review of Dependent Variable

The dependent variable for this quantitative study is software project failure. Scholars define software project success in many ways with multiple factors determining success. A lack of budget overrun, lack of time overrun, complete requirement coverage, and high customer satisfaction, defines project success (Martens & Van Weelden, 2014; Pelikan et al., 2017; Van den Hoek et al., 2014). The most straightforward definition of software project success is a project, which meets the customer’s needs (Bayona-Oré et al., 2014; Joslin & Müller, 2015; Project Management Institute, 2017). Project Management Institute (2017) argued a software project, may meet many of the traditional definitions of success but does not satisfy the customer, therefore that project is not a success. The Office of the Under Secretary of Defense for Acquisition, Technology, and

Logistics (OUSD[AT&L], 2015) definition of software success has a baseline from the traditional triad of software success factors. The triad of software project success is software delivered within scope, on time, and within budget (OUSD[AT&L], 2015). The premise is the Department of Defense (DoD) customers define the scope, schedule, and budget of the project, therefore if the project delivery meets the scope, schedule, and budget that meets the customer's needs and is thereby a success (OUSD[AT&L], 2015). No matter the definition of software project success, the satisfaction of the customer remains the primary focus.

There are multitudes of software development frameworks to choose from, but the agile framework stands out due to the focus on customer satisfaction. The agile manifesto's first principle is to satisfy the customer (Inayat, Salim, Marczak, Daneva, & Shamshirband, 2015). The agile program management framework places more focus on customer satisfaction rather than the traditional conformance standards (Inayat et al., 2015; Serrador & Pinto, 2015). In an agile framework, the customer's needs, rather than the scope, schedule, and budget determine when the project is successful (Serrador & Pinto, 2015). Although many frameworks have different success factors, customer satisfaction remains the anchor of each framework.

The agile framework concentrates on delivering value to satisfy the customer. A project manager delivering to a set scope, schedule, and budget, which no longer meets the customers' needs, does not enjoy success (Joslin & Müller, 2015). Many project managers trained in meeting scope, schedule, and budget find frustration the customer's satisfaction level changed since the project inception (Mir & Pinnington, 2014).

Technology, environment, and world events affect the customers' expectations and satisfaction (Joslin & Müller, 2015). The agile framework focuses on customer satisfaction, accelerated delivery of functionality, and bringing value to the customer.

The traditional mindset is the training many older project managers received. The conformance standards became the norm for evaluations of project managers and therefore became the standard for how to measure software project success (Bayona-Oré et al., 2014; Joslin & Müller, 2015; Project Management Institute, 2017). Many of the traditional measures included scope, schedule, budget, performance parameters, software recycle rates, and error rates (Mir & Pinnington, 2014). Due to the length of time between requirements definition and product delivery, the waterfall methodology often involves scope creep and customer dissatisfaction.

There are tradeoffs a traditional mindset project manager can make when determining what is successful in a software development project. Stakeholders and customers must be in constant communication and agree to any trade-offs (Bayona-Oré et al., 2014; Joslin & Müller, 2015; Project Management Institute, 2017). In software projects, a project manager can accept risk in the projects through the tradeoffs made (Joslin & Müller, 2015). If a project needs additional time in the schedule, then a project manager can decide to increase the budget to get the project back on schedule or decrease the scope to eliminate the portion making the schedule longer (Aziz & Wong, 2015). The project manager can decide to apply the same risk acceptance or aversion criteria to threats to software project scope increase or budget decrease (Aziz & Wong, 2015). If there is a reduction in the budget, then the project manager can decrease the scope to match the new budget or lengthen the schedule to accomplish the same items with a reduced budget (Estler, Nordio, Furia, Meyer, & Schneider, 2014; Mir & Pinnington, 2014). Likewise, if there is an increase in scope the project manager can increase the schedule to allow the scope increase or increase the budget to accommodate the scope increase while maintaining the same budget (Estler et al., 2014; Mir & Pinnington, 2014). Software development projects completed on schedule, within budget, meeting all specifications, with the proper team composition, and client satisfaction is the primary goal of traditional software development project managers.

Measurement of Variables

This measurement of variables section addresses a critical analysis and synthesis of the measurement of the independent and dependent variables. This section also addresses the reliability of each of the measurement instruments. In addition, the review of the measurement of the variables section describes the original purpose of the measurement instrument and facilitates the decision to use the instrument in this study.

Software development team structure. Software development team structure can affect the software development project’s success; however, I must measure to determine if a relationship exists. Galbraith (1973) stated software development team structure refers to how organizations implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty. I adapted a part of the measurement of software development team structure from a differentiated replication 60-item scale survey designed by Lindsjørn, Sjøberg, Dingsøyr, Bergersen, and Dybå (2016). Lindsjørn et al. (2016) reported the internal consistency of .81 for the team leader's effectiveness construct, as measured using Cronbach's Alpha. Lindsjørn et al. (2016) divided the original Hoegl and Gemuenden (2001) survey into two surveys to study agile and traditional team performance. My adaptation of the Lindsjørn et al. survey allows me to measure of software development team structure for this quantitative study.

Ambiguity. Ambiguity within software projects can affect the success of a software development project; however, I must measure to determine if a relationship exists. Galbraith (1973) defines ambiguity as users having different frames of references, therefore, creating multiple perspectives. Kahn, Wolfe, Quinn, Snoek, and Rosenthal (1964) indicated role ambiguity results from the organization's size and complexity. Rizzo, House, and Lirtzman (1970) stated classical organization theory and role theory is the basis for role ambiguity research. The role ambiguity question Rizzo et al. sought to answer was "Is role ambiguity associated with decreased satisfaction and organizational effectiveness" (p. 154). Adaptation of ambiguity in software specifications measurement comes from a 30-item scale survey known as the Rizzo, House, and Lirtzman (RHL) scale (Rizzo et al., 1970). John Rizzo, Robert House, and Sidney Lirtzman developed the RHL scale to study ambiguity, which is a seven-point Likert-type scale (Schuler, Aldag, & Brief, 1977). Using Kuder-Richardson internal consistency the reliabilities for two samples with Spearman-Brown correction were .780 and .808 for the 13 questions of the RHL measure (Rizzo et al., 1970).

Rizzo et al. (1970) studied two samples for the same population. Sample A included 199 participants and sample B included 91participants. The RHL scale questionnaire includes 30 items; however, only the even number questions deal with role ambiguity (Rizzo et al., 1970). Of the 15 ambiguity related questions, two of the questions were duplicates, which resulted in the elimination of one of the questions (Rizzo et al., 1970). One other question required elimination because Rizzo et al. (1970) determined the question did not adequately measure ambiguity. My adaptation of the Rizzo et al. survey allows me to measure ambiguity within a software project for this quantitative study.

Volatility. Volatility can affect the software development project's success; however, I must measure to determine if a relationship exists. The volatility variable refers to the extent to which changes occur over time Galbraith (1973). This study will use an 8-item scale survey developed by Zowghi and Nurmuliani (2002). The Zowghi and Nurmuliani instrument assesses the impact of requirements volatility on software project performance. The Cronbach Alpha reliability value for the first three questions of the Zowghi and Nurmuliani survey is .74 for and .77 for the last three questions.

Software project success. Software project success is essential to a software development company's viability. The definition of software project success is a software development project completed within schedule, within budget, meeting all specifications, with the proper team composition, and client satisfaction (da Silva et al., 2013). To measure software project success da Silva et al. (2013) used a revised questionnaire called the G questionnaire, originally developed by Haggerty (2000), which focused on project success factors. The revised G questionnaire originally was a three-point Likert-type scale in the original study, which requires conversion to a five-point

Likert-type scale for greater granularity (da Silva et al., 2013). The original G questionnaire was a three-point ordinal questionnaire with a single question cost, time, scope, team satisfaction, client satisfaction, and project manager satisfaction. The revised G questionnaire contains two questions each about cost, time, scope, team satisfaction, client satisfaction, and project manager satisfaction; with an added two questions about project success. The rationale for two questions was to ask a positive and a negative question for each main question to increase the reliability. The 14 questions of the G questionnaire were answered twice by each participant, once for a successful project and once for a not successful project. The overall reliability of the da Silva et al. survey was .0900. Da Silva et al. (2013) reported Cronbach alpha reliability of each sub-dimension of the project success measure, costs 0.884; implementation date 0.793; scope 0.750; team satisfaction 0.649; user satisfaction 0.761; project manager satisfaction 0.712; and overall project success 0.855.

Review of Alternatives to the OIP Theoretical Framework

As a researcher reviews literature to support a particular theory, the researcher discovers other rival theories, which provide different viewpoints of the research topic. The study of the organizational information processing theory uncovered alternate theories to assist in understanding software project success. While bureaucratic management theory, organization theory, information processing theory have relevance to understanding software project success; the most relevant is contingency theory.

Bureaucratic management theory. The theory of bureaucratic management reflects that an organization should be structured based on hierarchy and the governance of its members defined with rational-legal decision-making rules (Gregory & Keil, 2014; Johnsen, 2016; Vittikh, 2016). Max Weber authored the theory of bureaucratic management in a book he published in 1922. Within the organization is a hierarchy defined by the level of authority both above and below the organization (Gregory & Keil, 2014). Each level of leadership in the organization understands the authority above and below within the organization with a central leader at the top (Gregory & Keil, 2014). The bureaucratic organization makes decisions through predefined rules with explicit objectives, documented in policies and procedures (Johnsen, 2016; Vittikh, 2016). The bureaucratic management theory is in direct conflict with modern forms of software project development. Agile development methodologies rely on collaboration between the various members of the project management office and functional management office (Brhel et al., 2015). The traditional waterfall and other older software development methodologies were better suited for the theory of bureaucratic management (Conforto et al., 2014; Johnsen, 2016).

Organization theory. Organization theory is not a single theory but a collection of studies about organizational designs and structures, fashioned to study organizational behaviors. The organization theory centrally concerns the relationship of organizations, the organization’s external environment, behaviors of managers, and the behaviors of other organizational leaders (Davis, 2015). Through the organization theory, theorist described alternative ways an organization can cope with rapid change (Davis, 2015; Örtenblad, Putnam, & Trehan, 2016). These alternative approaches to rapid change help describe how an organization processes information to develop alternative courses of action (Lounsbury & Beckman, 2015). The resulting alternative courses of actions determine the design of the organizational structure to deal with the rapidly changing environment.

Organizational theory is similar to the organization information processing theory in that information is the center of the theory, but the organization theory suggests ways an organization can cope with rapid change (Rowlinson et al., 2014). Although organization theory is similar to organization information processing theory, it fails to prescribe an organizational structure to determine software project success (Örtenblad et al., 2016). Studying managerial decision-making and information-processing is very useful to all organizations but cannot determine the organization's primary activity success or failure.

Information processing theory. The information processing theory is the combination of two central ideas. In 1956, George Miller combined the ideas of the capacity of short-term memory and information-processing into the theory known as the information processing theory (Gurbin, 2015; Maity, Dass, & Malhotra, 2014; Wong, Lai, Cheng, & Lun, 2015). The capacity of short-term memory refers to a concept Miller called chunking. The central idea of this thought is a person's short-term memory only can hold five to nine chunks of information (Gurbin, 2015; Wong et al., 2015). The second central idea is the human mind operates much like a computer (Gurbin, 2015; Wong et al., 2015). Whereas, like a computer, the human mind receives information, processes the information, stores the information, and creates responses to the information received (Gurbin, 2015; Wong et al., 2015).

Jay Galbraith used George Miller’s information processing theory as a baseline for the organizational information processing theory (Zand, Solaimani, & van Beers, 2015). The fundamental idea in Miller’s information processing theory and Galbraith’s organizational information processing theory is the flow of information is essential for the organization (Zand et al., 2015). Miller’s information processing theory mainly defines the human aspects to information processing with some relationship to the organization (Gurbin, 2015; Maity et al., 2014; Wong et al., 2015). When assessing organizational structure in software projects Miller’s information processing theory does not go into depth about the organizational structure (Zand et al., 2015). To reach the depth needed for this study I must use Galbraith’s organizational information processing theory to obtain an operational perspective towards analyzing a software development company’s structure.

Contingency theory. Contingency theory proposes there is no best way to design organizational structures. Joan Woodward originated the contingency theory in 1958. The external and internal environment of the organization determines the best way of organizing the organization (McAdam, Miller, & McSorley, 2016). Each organization faces a different set of internal and external environmental factors; therefore, the environmental factors will dictate the preferred organizational structure of the organization (McAdam et al., 2016). The challenge for the organization is to take into account the uncertain internal and external environmental influences to design the proper organizational structure for success in handling future uncertainties (Furlan Matos Alves, Lopes de Sousa Jabbour, Kannan, & Chiappetta Jabbour, 2017). Although the general aspects of the contingency theory seem plausible, theorists attempted to generalize organizational structures to apply to similar technologies (Otley, 2016; Suddaby, 2015).

The thought behind the generalization of the contingency theory was technology determines organizational attributes like the span of control, centralization of authority, and rules (Kim, Chung, Lee, & Preis, 2015). While some technologies are similar in organization, the internal and external environmental influences prescribe differences in organizational structure (Kim et al., 2015; Otley, 2016; Suddaby, 2015). While the contingency theory has validity, it does not apply when determining software project success. Software development companies are structured less formally due to the continually evolving nature of software technologies. Researchers determined less formal organizational structures are useful when uncertainty is continuously present in the internal and external organizational environments (Mikes & Kaplan, 2015).

In addition, researchers discovered a decentralized organizational structure allows organizations in less stable environments to rely on mutual adjustments between various departments in the company (Mikes & Kaplan, 2015; Whalen et al., 2016). Companies who operate in unstable environments are more effective when the tasks within the company are differentiated, yet heavily integrated (Furlan Matos Alves et al., 2017; Whalen et al., 2016). Conversely, companies in stable environments operate better if there is a formalized organizational structure with centralized decision-making (Furlan Matos Alves et al., 2017; Whalen et al., 2016).

Transition

The material presented in Section 1 includes (a) an overview of the background of the problem, (b) a review of the business problem, and (c) the purpose of the study. In addition, in Section 1 I presented the nature of the study, the research question, hypotheses, the theoretical framework, study definitions, assumptions, limitations, and delimitations. Lastly, Section 1 included a comprehensive literature review, which comprises of critical analysis and synthesis of literature sources and a critical review of the literature related to software development team structure, ambiguity, and volatility and software project failure. The subsections of the literature review included the purpose of the study, hypotheses, the analysis and synthesis of OIP theory literature, the critical analysis and synthesis of independent variables, the critical analysis and synthesis of the dependent variable, the measurement of variables, and alternatives to the OIP theoretical framework.

The material and data I present in Section 2 will include (a) an overview of the project, (b) the purpose statement, (c) the role of the researcher, (d) the participants, and (e) include an outline of the research method and design. In addition, the material I present in Section 2 detail (a) the population and sampling method, (b) ethical research, (c) the data collection instruments, (d) data collection techniques, (e) data analysis methods, and (f) study validity. Section 3 embodies information about the (a) a presentation of the study findings, (b) the application of research findings to professional practice, (c) implications of the study for social change, (d) recommendations for action and future research, (f) reflections, (g) a summary, (h) and study conclusions.

Section 2: The Project

Purpose Statement

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure in software development companies that developed software for the USAF. This quantitative correlational study included the three independent variables: team structure, ambiguity, and volatility of SW specifications. The dependent variable was software project failure. The target population was software development company owners contracted by the USAF. The simple random sample consisted of software development company owners contracted by the USAF who were members of the Armed Forces Communications and Electronics Association (AFCEA) in the Southeastern United States. AFCEA membership included software company owners who provided software development for the military, the federal government, and state governments. Contributions to positive social change included increasing software success, reducing tax dollar waste, and increasing the public’s trust in national defense.

Role of the Researcher

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure in software development companies that developed software for the USAF. Yilmaz (2013) stated that researchers conducting a quantitative study must take on the role of an objective observer. As an objective observer, the researcher does not influence or participate in the research (O'Leary, 2017). My role as the researcher was to (a) research, adapt, and design the survey instrument; (b) distribute the survey via SurveyMonkey; (c) recruit and communicate through e-mail with the survey respondents; (d) use SPSS to analyze the survey data; and (e) interpret and document the results.

I had extensive knowledge and experience in software development and software development project management. I was an acquisition program manager in the USAF providing programmatic assistance to 52 software programs. I have worked in the USAF as a program manager for the past 20 years. I spent 10 years as an active-duty officer working aircraft hardware systems before retiring and returning to work 10 years in business systems software acquisitions. Also, I was familiar with the target population that included members of the AFCEA in the Southeastern United States. The AFCEA members represented the companies that contracted with the USAF to develop software for the USAF software programs. AFCEA membership included business leaders of software companies that provided software development for the military, the federal government, and state governments.

As a researcher, I was responsible for maintaining ethical standards and protecting the participants in this study. Prior to the collection of data, I obtained approval from Walden University's institutional review board (IRB) (Walden University, n.d.). I completed an IRB application and obtained approval from Walden University's IRB before contacting the respondents about this study or collecting data. The purpose of the Belmont Report was to provide researchers with guidelines for ethical conduct in human subject research (U.S. National Commission, 1978). Approval of the IRB data collection procedure met the established ethical standards and guidelines of the National Institute of Health (NIH), the Belmont Report, and Walden University's IRB. Upon receiving approval from the IRB, I sent the survey participants an e-mail message with informed consent and confidentiality statements, along with an invitation to participate in the survey on a voluntary basis.

Participants

Participants were software development company owners contracted by the USAF who are members of AFCEA in the Southeastern United States. AFCEA membership included software company owners who provided software development for the military, the federal government, and state governments. I sent an invitation to participate to approximately 150 software development company owners contracted by the USAF who are members of AFCEA in the Southeastern United States. To gain access to the list of potential participants, I used the AFCEA corporate member's list, which was publicly accessible.

The members of AFCEA in Southeastern United States developed software code, projects, and programs for approximately 162 USAF software programs. Because of my employment role, I interacted with the AFCEA members on a continuous basis through conferences, symposiums, meetings, and government-sponsored vendor industry days. Many of the AFCEA members were aware of my efforts to obtain a doctoral degree. Because the AFCEA members had personal knowledge of my intentions, they had previously offered to participate in a survey. Therefore, I was confident I would obtain a high survey response rate. To initiate the formal study relationship with the respondents, I sent an e-mail to each participant. The e-mail contained an informed consent statement, a link to SurveyMonkey, and instructions for completing the survey.

Research Method and Design

I chose a quantitative method and correlational design. The quantitative method allowed me to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. I used a correlational design to examine relationships between the independent variables (software development team structure, ambiguity, and volatility in software specifications) so I could explain variation in relation to software project failure.

Research Method

The chosen methodology for this doctoral study was quantitative. Researchers use quantitative methods to collect numerical data to test hypothesis based on an existing theory (Bernard, 2011; O'Leary, 2017; Williams, 2007). In addition, researchers use quantitative methods to generalize the results of the data analysis across groups of people or to explain a particular phenomenon (Bernard, 2011; Palinkas et al., 2015; Teddlie & Tashakkori, 2003). Researchers also use quantitative methods to achieve a breadth of understanding of predictors for dependent variables (Leedy & Ormrod, 2015; O'Leary, 2017; Teddlie & Tashakkori, 2003). The quantitative method was appropriate for this doctoral study because I used it to collect numerical data, test hypotheses, and generalize findings across the population of software development company owners.

Researchers use qualitative methods to obtain results that are naturalistic, interpretative, and rich (Leedy & Ormrod, 2015; O'Leary, 2017; Ritchie, Lewis, Nicholls, & Ormston, 2013). Because I did not seek thick, rich knowledge but rather a breadth of understanding, qualitative methods were not suitable for my study. A qualitative researcher relies on knowledge of the social world, coupled with his or her interpretations and understanding of the phenomenon (Bernard, 2011; Corbin & Strauss, 2015; Ritchie et al., 2013). Researchers use mixed methods to employ both quantitative and qualitative methods (Kemper, Stringfield, & Teddlie, 2003; O'Leary, 2017; Williams, 2007). Because the purpose of this study was not to explore themes emerging from exploring a qualitative phenomenon, the mixed-methods approach was not appropriate.

Research Design

The chosen design for this study was correlational because researchers use a correlational design to examine predictive or explanatory relationships between independent and dependent variables. Quantitative researchers use a correlational design to examine the differences between the constructs of the group under study (Chen, 2012; Leedy & Ormrod, 2015; Thompson, Diamond, McWilliam, Snyder, & Snyder, 2005). A rationale for a correlational study is to use correlation as a statistical test to establish patterns between two or more variables (Barker & Pistrang, 2016; Leedy & Ormrod, 2015; Thompson et al., 2005). Correlational studies may be cross-sectional, which enables the researcher to make measurements at the same point in time. Correlational designed studies can also be longitudinal in which the researcher makes measurements at different points in time (Håkansson, 2013; Leedy & Ormrod, 2015; Thompson et al., 2005).

Other possible quantitative designs are descriptive, experimental, and quasi-experimental. To explain a phenomenon systematically, a researcher would use a descriptive design (De Vaus, 2013; Håkansson, 2013; O'Leary, 2017). The researcher uses experimental and quasi-experimental designs to control as many variables as possible in the search for cause and effect (De Vaus, 2013; O'Leary, 2017; Park & Park, 2016). The purpose of this correlational study was to examine the relationship between the independent variables to predict the variation in the dependent variable; therefore, descriptive, experimental, and quasi-experimental designs were not appropriate for this study.

Population and Sampling

This section includes a description of the population from which the sample was obtained. The section also includes a justification of the sampling method and the strengths and weaknesses associated with the chosen sampling method. I performed a power analysis to justify the sample size, including justification for the effect size, alpha, and power levels.

Population

The population for this study was software development companies. The random sample drawn from the population of software development companies included software development company owners contracted by the USAF who are members of AFCEA. Southeastern United States is the location for the development of many USAF software programs. Due to the location of the USAF software program management offices, many software development companies positioned their offices in the Southeastern United States to be in close proximity. These software development companies were also members of AFCEA. AFCEA membership included software company owners who provided software development for the military, the federal government, and state governments. I limited the random sample to AFCEA members who have or were currently working on an Air Force software development programs or project.

Sampling

The two broad categories of sampling are probabilistic and nonprobabilistic (Barker & Pistrang, 2016; Bornstein, Jager, & Putnick, 2013; Palinkas et al., 2015). The main difference between probabilistic and nonprobabilistic sampling is that all participants in a population have a chance of being selected in probabilistic sampling (Barker & Pistrang, 2016; Bornstein et al., 2013; Palinkas et al., 2015). To determine whether probabilistic or nonprobabilistic sampling is appropriate for a study, the researcher must examine the availability of the population, cost of obtaining the sample, and time to obtain the sample (Barker & Pistrang, 2016; Bornstein et al., 2013; Palinkas et al., 2015). When choosing to use probabilistic sampling, a researcher must also determine whether the contact information for the entire population is current and available (Barker & Pistrang, 2016; Bornstein et al., 2013; Palinkas et al., 2015). Nonprobabilistic sampling is appropriate when a comprehensive population list is not available, the sample is not random, or the results cannot be generalized to the entire population (Barker & Pistrang, 2016; Bornstein et al., 2013; Palinkas et al., 2015).

Types of probabilistic sampling include simple random sampling, stratified sampling, systematic sampling, and cluster sampling (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Researchers use simple random sampling to give every participant in a population an equal chance of participating in the sample (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). When a researcher uses stratified sampling, the researcher must divide the population into separate groups and choose randomly from each group (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Systematic sampling occurs when the researcher selects the first participant and then at preplanned intervals selects the remainder of the participants for the sample (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Cluster sampling involves selecting random clusters of participants from a population (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013).

Types of non-probabilistic sampling include availability, purposive, quota, and snowball (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). In availability sampling, the researcher selects participants from the population based on the availability or convenience to the researcher (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Purposive sampling occurs when the researcher selects participants based on predetermined criteria of inclusivity to fit the study (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Researchers use quota sampling when they need to subdivide the population into exclusive groups and request participation from within those groups until the desired number is reached (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013). Snowball sampling occurs when researchers select participants based on referrals from participants previously selected from the study (Barker & Pistrang, 2016; Bornstein et al., 2013; Uprichard, 2013).

The broad category of sampling for this quantitative study was probabilistic. One of the appropriate sampling strategies for quantitative studies is simple random sampling, which gives every participant in the population an equal chance of selection (Ivarsson, Anderson, Johnson, & Lindwall, 2013; Palinkas et al., 2015). Simple random sampling is a technique in which a smaller subset of participants originates from a larger population and every member of the larger population has an equal chance of being included in the smaller subset (Ivarsson et al., 2013; Kessler et al., 2015; Palinkas et al., 2015). Researchers value simple random sampling because of the ease of assembling the sample and the representativeness of the population (Barker & Pistrang, 2016; Kessler et al., 2015; Uprichard, 2013). The representativeness of the population makes it easier for the researcher to generalize the results to the population (Kessler et al., 2015; Palinkas et al., 2015; Uprichard, 2013). The weakness of simple random sampling is the researcher must have a complete list of all members of the population (Kessler et al., 2015; Palinkas et al., 2015; Uprichard, 2013).

I used the G*Power 3 program to determine the minimum sample size needed for my study. G*Power 3 is a free statistical analysis program that researchers use to determine sample size (Faul, Erdfelder, Lang, & Buchner, 2007; Faul, Erdfelder, Buchner, & Lang, 2009). In the G*Power 3 program, the researcher is required to select the alpha error, beta error, and effect size to determine the appropriate sample size.

Selecting a priori as the power analysis allows the researcher to estimate the minimum required sample size before data collection (Faul et al., 2007; Faul et al., 2009).

Selecting a priori in the G*Power 3 program requires the researcher to provide the alpha error, beta error, and effect size. Avoiding or reducing Type I and Type II errors in quantitative studies determines the need for probabilistic sampling (Ivarsson et al., 2013; Kessler et al., 2015; Palinkas et al., 2015). Alpha (α) is the threshold level for the rate of probability of having a Type I error, and the accepted value is 0.05 (Ivarsson et al., 2013, Kessler et al., 2015; Palinkas et al., 2015). Beta (β) is the threshold level of probability of having a Type II error, and a commonly accepted value is 0.20 (Ivarsson et al., 2013, Kessler et al., 2015; Palinkas et al., 2015). Effect size emphasizes the correlation between the two groups (Cohen, 1992). Cohen (1992) generalizes effect size as small (0.10), medium (0.30), and large (0.50). Cohen (1992) further stipulates a medium effect size between observed groups is noticeable to the naked eye. The effect size chosen for this study was $f^2 = 0.15$, which indicates the observed differences between my variables, was small and not readily noticeable. With an a priori power analysis, assuming an effect size (f^2 =0 .15), $\alpha = 0.05$, and three predictor variables, a minimum sample size of 77 participants were required to achieve a power of .80 (β=.20). Therefore, I required at least 77 participants for this quantitative study (Table 1).

Although I used published data collection procedures in this study, the number of samples received was only 23. The community of software development company owners contracted by the USAF who are members of AFCEA within the Southeastern United States geographical area included approximately 150 software development company owners contracted by the USAF. Of the 150 AFCEA members I sent the survey, I believed I would receive at least 77 samples. Since my sample size resulted in 23 complete responses instead of the 77 expected responses, I employed bootstrapping 95% confidence intervals using 2,000 samples, to address the possible impact of assumption violations. There were no indications why more than 23 AFCEA members did not respond to the survey. None of the 150 AFCEA members contacted me to address why they did not respond, and the survey was completely anonymous, so I could not determine who responded and who did not.

Table 1

Linear Multiple Regression Sample Size

Input parameter		Output parameter	
Effect size f^2	.015	Non centrality parameter	11.5500000
α err prob	0.05	Critical F	2.7300187
Power (1-β err prob)	0.80	*Df*	73
# of predictors	3	Total sample size	77
		Actual power	0. 8017655

Ethical Research

I collected data for this doctoral study from voluntary participants who are members of AFCEA, in accordance with the guidelines of the National Institute of Health (NIH) Office of Extramural Research and Walden University's IRB. I emailed an invitation to participate in the survey to the participants. The invitational email contains information about informed consent, study withdrawal procedures, procedures for the ethical protection of information, and a statement about the storage of data for five years to protect the anonymity of participants. There are no promised or implied incentives for participation in this study.

The participants indicated understanding and consent by going to the SurveyMonkey.com and completing the survey (see Appendix A for the survey). Participants withdrew from the survey any time they preferred. Withdrawal from the initial stages of the survey throughout the survey completion, the participants who choose not to respond, did not complete the survey.

I retrieved the survey data from SurveyMonkey.com, stored the data on a thumb drive, encrypted the thumb drive with US DoD approved AES encryption protocols, and I maintained the only encryption key. Once I started using the thumb drive, it remained locked in my personal fireproof safe when not in use. Once approved, I will use the original email listing to email an executive summary of the final study to the participants. Schneider (2013) states methods of destruction thumb drives include degaussing, physical destruction through crushing or overwriting stored data. To protect the participant’s anonymity, five years after the studies published date I will destroy the survey results by crushing the thumb drive with a hammer as described by Schneider (2013). The Walden University IRB approval number for this doctoral study is 04-20-18-0553975.

Instrumentation

In the instrumentation section, I provided the name and year for the instruments I used to create my survey instrument for the quantitative correlation study. I also provided the developer’s name for each instrument. I outlined the concept of measurement for each instrument as well as a detail description of each. To accompany the description of the instruments, I included the reliability measure for each of the instrument. Lastly, I included information about the validity and reliability of the instruments.

Survey Instrument

I adapted the data collection instrument of this study from four published data collection instruments. The original scoring of the instruments was a five-point scale ranging from 1 - strongly disagree to 5 - strongly agree, a seven-point scale ranging from very false to very true, a five-point scale ranging from -2 - strongly disagree to +2 - strongly agree, and a five-point scale ranging from 1 - strongly disagree to 5 - strongly agree. Therefore, readers are to judge the results with caution, as the instrument has been altered from its original scoring scale.

Using a standardized survey instrument indicates the survey instrument has psychometric validation (De Vaus, 2013). A valid survey instrument is a survey which receives consistent responses, measures the construct intended to be measured, and differentiates between the good and bad qualities of the construct (Fowler & Cosenza, 2009). When a researcher alters the survey instrument, the researcher risks invalidating the reliability and validity of the original instrument. When a researcher changes the wording, dropping items from the instrument, changes available responses, changes the wording from negative or positive, or changes the language the researcher risks losing the advantages of the standardized survey. However, carefully altering of words can provide clarity to the respondents. To provide clarity, I carefully altered the survey attached to this study. However, there are acknowledge risks with altering this survey; therefore, the reader should judge the results of this study with caution.

The first part of the survey contained statements about how the software development team's structure and organization for the last project the respondent managed and completed. Part two contained statements about the software specifications for the last project the respondent managed and completed. Part three contained statements about the outcome of the last project the respondent managed and completed. Parts one, two, and three of the survey were Likert-style and asked the respondent to indicate the extent to which they: (a) 1- strongly disagree, (b) 2 - disagree, (c) 3 - disagree somewhat, (d) 4 - undecided, (e) 5 - agree somewhat, (f) 6 - agree, and (g) 7 - strongly agree. However, questions 8, 10, 12, 13, 14, 16, 18, 19, 22, and 24 were reverse coded to: (a) 1- strongly agree, (b) 2 - agree, (c) 3 - agree somewhat, (d) 4 - undecided, (e) 5 - disagree somewhat, (f) 6 - disagree, and (g) 7 - strongly disagree. Because the questions were originally asked negatively I reverse coded those questions. Appendix A contains the exact instrument for delivery via SurveyMonkey (see Appendix A for the survey).

Measures

Software development team structure. Software development team structure refers to how organizations implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty (Galbraith, 1974). I adapted the software development team structure measure from a 60-item scale survey designed by Lindsjørn et al. (2016). The Lindsjørn et al. (2016) instrument measured responses with a Likert-style scale, with responses ranging from 1 - strongly disagree to 5 - strongly agree. I altered the original instrument from a 5-point Likert-style to a 7-point Likert-style scale to increase the fidelity in the responses. Appendix B shows the list of all items for this measure (see Appendix B for the items for measures). Lindsjørn et al. (2016) reported the internal consistency of .81 for the team leader's effectiveness construct, as measured using Cronbach's Alpha.

Ambiguity. Galbraith (1973) defines ambiguity as users having different frames of references, therefore, creating multiple perspectives. I adapted the ambiguity measure from a 30-item scale survey known as the RHL scale (Rizzo et al., 1970). The psychometric properties contained in the RHL measure are widely used to study ambiguity (Schuler et al., 1977). The RHL scale measured responses with a 7-point Likert-style scale ranging from very false to very true. I kept the 7-point instrument as a 7-point Likert-style scale to ensure fidelity in the responses. Appendix B shows the list of all items for this measure (see Appendix B for the items for measures). Rizzo et al. (1970) performed a reliability analysis using the Kuder-Richardson internal consistency reliabilities with Spearman-Brown correction and measured the reliability as .808 for the ambiguity construct in the RHL measure (Rizzo et al., 1970). Although the original survey did not reverse code any of the original survey questions chosen for this survey, I chose to reverse code item 13 of this study's survey because the question was initially coded negatively and all other questions were asked positively (see Appendix B for the items for measures). The reverse coding was: (a) 1- strongly agree, (b) 2 - agree, (c) 3 - agree somewhat, (d) 4 - undecided, (e) 5 - disagree somewhat, (f) 6 - disagree, and (g) 7 - strongly disagree.

Volatility. The volatility variable refers to the extent to which changes occur over time Galbraith (1973). I adapted an 8-item scale survey developed by Zowghi and Nurmuliani (2002). The Zowghi and Nurmuliani (2002) instrument measured the responses with a Likert-style scale with responses ranging from 1 - strongly disagree to 5 - strongly agree. I altered the original instrument from a 5-point Likert-style to a 7-point Likert-style scale to increase the fidelity in the responses. Appendix B shows the list of all items for this measure (see Appendix B for the items for measures). The Cronbach Alpha reliability value for the first three statements of the Zowghi and Nurmuliani (2002) survey is .74 and .77 for the last three statements (Zowghi & Nurmuliani, 2002). Although the original survey did not reverse code any of the original survey questions chosen for this survey, I chose to reverse code item 8, 10, 12, 14, 16, and 18 of this study's survey because the original questions were coded negatively, and other questions were asked positively (see Appendix B for the items for measures). The reverse coding was: (a) 1- strongly agree, (b) 2 - agree, (c) 3 - agree somewhat, (d) 4 - undecided, (e) 5 - disagree somewhat, (f) 6 - disagree, and (g) 7 - strongly disagree.

Software project failure. Software project success is a software development project completed within schedule, within budget, meeting all specifications, with the proper team composition, and client satisfaction (da Silva et al., 2013). I adapted a 14-

item scale survey from previous studies (da Silva et al., 2013; Haggerty, 2000). The da Silva instrument was measured with a Likert-style scale with responses ranging from -2 - strongly disagree to +2 - strongly agree. I altered the original instrument from a 5-point Likert-style to a 7-point Likert-style scale to increase the fidelity in the responses. Appendix B shows the list of all items for this measure (see Appendix B for the items for measures). The internal reliability of the da Silva survey was .0900 as measured by Cronbach Alpha (da Silva et al., 2013). Although the original survey did not reverse code any of original survey questions chosen for this survey, I chose to reverse code item 19, 22, and 24 of this study's survey because the original questions were coded negatively, and other questions were asked positively (see Appendix B for the items for measures). The reverse coding was: (a) 1- strongly agree, (b) 2 - agree, (c) 3 - agree somewhat, (d) 4 - undecided, (e) 5 - disagree somewhat, (f) 6 - disagree, and (g) 7 - strongly disagree.

Data Collection Technique

The data collection instrument for this quantitative correlation study was a web-based survey using SurveyMonkey. Web-based surveys are a type of electronic survey (McPeake, Bateson, & O'Neill, 2014). Web-based surveys allow the respondent to go to a website to complete the survey (Bryman, 2012). Web-based survey services have evolved data collection techniques by making survey research easier, cheaper, and faster (Wright, 2005). Although, there is a free version of SurveyMonkey this research study required the standard plan, which was $37 per month. SurveyMonkey eases the distribution of the survey and collection of the responses in a Web-enabled fashion (SurveyMonkey, n.d.; Wright, 2005). Bojcic, Sue, Huon, Maletis, and Inacio (2014) found electronic surveys result in the collection of good-quality data, without missing responses. Petrovčič, Petrič, and Manfreda (2016) stated the presence of the researcher's authority in an email, a request for the respondents help, the sense of community, and intense community activity are positive influences on response rate.

The target population for this study was software development companies, which develop software for the USAF. A simple random sample consisted of software development companies, which develop software for the USAF, which are members of the AFCEA within the Southeastern United States geographical area. AFCEA membership includes software company owners who provide software development for the military, the federal government, and state governments. The respondent received an invitational email message with directions on how to obtain the survey and instructions for completion, to ensure a clear understanding of the intent of the survey (see Appendix A for the survey). Sauermann and Roach (2013) stated one of the ongoing problems with online surveys is low response rates. Follow on reminders effectively increase response rates (Sauermann & Roach, 2013). Fourteen days after sending the survey I sent a follow-up, email reminder, and then seven days later I sent an additional email reminder.

This study used SurveyMonkey to collect the survey responses. The data from SurveyMonkey required exportation and then importation into SPSS for analysis. Electronic surveys have become so sophisticated the results receive automatically compiling and are often available immediately (Bojcic et al., 2014). The selected version of SurveyMonkey provided data collection, team collaboration, unlimited questions and responses, 24/7 email support, skip logic, data exports, and reports (SurveyMonkey, n.d.).

Although SurveyMonkey could analyze data, the SurveyMonkey capabilities do not provide the in-depth analysis needed for this research. Therefore, SPSS is the tool used to analyze the data from the survey.

Data Analysis

To determine the statistical analyses appropriate for a quantitative study; the researcher must know the specific research questions, types of data to be collected, projected sizes of sample and groups, and the independent and dependent variables (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler & Reinhart, 2017). In this study, the overarching research question was: What is the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure?

Hypotheses

H_0: There is no relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

H_a: There is a relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure.

Types of Variables in This Study

Researchers must consider the types, and the number of variables included in a study when deciding which statistical analysis method to use to test the hypothesis (O’Leary, 2017). This study included three independent variables and one dependent variable. Researchers classify quantitative variables into two broad categories, continuous and discrete variables (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler &

Reinhart, 2017). Continuous variables are numeric variables that can hold any value between its minimum and maximum values (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler & Reinhart, 2017). Discrete variables are numeric variables that have a value from a finite set of possible values between its minimum and maximum values (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler & Reinhart, 2017). In this research study, both the independent variables and dependent variable are continuous variables and can have a value from an infinite number of possible values between the minimum and maximum values. Multiple regression analysis and structural equation modeling (SEM) are appropriate when a study contains multiple continuous independent variables and one continuous dependent variable (Henseler, 2017; Keith, 2014; Kline, 2016). This study had three continuous independent variables and one dependent variable; therefore, multiple regression analysis and SEM are possible statistical analysis method in this study.

There are four categories of scales of measurements, which are; nominal, ordinal, interval, and ratio scales (Bishop & Herron, 2015; Siddiqui, Bajwa, Elahi, & Fahim, 2016). Researchers use nominal scales of measurement when there is no order to data, but researchers place the data in logical grouping and assign numbers (Bishop & Herron, 2015; Siddiqui et al., 2016). Researchers use ordinal scales of measurement when data has no discrete measurements; instead, the display of data is in order of magnitude (Bishop & Herron, 2015; Siddiqui et al., 2016). Researchers use interval scales of measurement when the differences between the data values are quantifiable (Bishop & Herron, 2015; Siddiqui et al., 2016). Researchers use ratio scales of measurement when the scale has a fixed zero scale and allowing the comparison between values (Bishop & Herron, 2015; Siddiqui et al., 2016). The scale of measurement for this study was ordinal. Researchers typically use Likert instruments to quantify responses on different issues (Siddiqui et al., 2016). Likert instruments are ordinal due to the responses have no absolute values between them, but rather the arrangement of the responses are on order of magnitude (Siddiqui et al., 2016).

Statistical Analysis

Researchers use multiple regression methods to study the magnitude of the association and the statistical relationship between two or more independent variables and one dependent variable (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler & Reinhart, 2017). Multiple regression analysis allows the researcher to recognize the best predictors by looking at all of the independent variables at the same time (Bezzina & Saunders, 2014; Little & Rubin, 2014; Mertler & Reinhart, 2017). Researchers use SEM when the study includes a complex theoretical model with layers of mediating variables (Henseler, 2017; Keith, 2014; Kline, 2016). Researchers translate the complex model into multiple equations and can use the SEM to simultaneous test all equations (Henseler, 2017; Keith, 2014; Kline, 2016). The analysis process of SEM is complicated; therefore, the results are difficult for researchers and readers to interpret (Bezzina & Saunders, 2014; Henseler, 2017; Kline, 2016). In this study, multiple regression analysis was more appropriate than SEM because the theoretical model in this study had only one set of independent variables that I can translate into a single equation.

Data Cleaning and Missing Data

Data cleaning is the process of discovering errors in data and then fixing those errors (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). Researchers use data cleaning to determine inaccurate, incomplete, or unreasonable data (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). Researchers also use data cleaning to improve the quality by correcting detected errors and omissions (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016).

Missing data occurs when participants do not respond to certain statements in a survey or observation (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). Missing data is a common occurrence and can affect conclusions drawn from the data (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). There are several methods to handle missing data to include ignoring the missing data; deletion of the features, which contain the missing values; and inputting the missing data (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). Only inputting the missing data is the most reliable (Ales & Marko, 2017; Kulkarni & Bakal, 2014; Sessa & Syed, 2016). To avoid missing data, I formatted SurveyMonkey to require a response to each statement before the participant could go to the next statement. Once the participants completed imputation of their responses in SurveyMonkey, I used IBM SPSS Statistics Program, version 24, to perform needed statistical analysis.

Statistical Analysis Assumptions

The required statistical analysis for this study was multiple regression analysis, and the use of multiple regression analysis required some assumptions. The critical assumptions for multiple regression analysis are outliers, normality, linearity, multicollinearity, and independence of residuals (Amin, Akbar, & Manzoor, 2015; Ernst & Albers, 2017; Williams, Gómez Grajales, & Kurkiewicz, 2013). To test the outliers, normality, linearity, and independence of residuals assumptions, I used a Normal Probability Plot (P-P) of the Regression Standardized Residual. A normal P-P is a graphical tool to test normality assumptions (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). The normal P-P is more precise than a histogram, and too much or too little power does not affect the test (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013).

Outliers distort significance test and relationships (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). Outliers can occur due to coding errors, as a result of measurement errors, unintended participants outside the intended population, or typographical mistake (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). To test this assumption, I used a normal P-P so I can identify any outliers. To deal with the outliers, I data cleaned the outliers to reduce the probability of Type I and Type II errors.

An additional assumption is normality, whereas all variables have normal distributions (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). Normality is required for multiple regression and to test this assumption I used a normal P-P. I also examined the skewness and kurtosis of each variable to estimate the shape of the distribution. I adopted previously tested and validated measures to reduce the risk of other than normal distribution, and therefore the occurrence is unlikely to happen. However, I examined kurtosis, skewness, and normal P-P to ensure a normal distribution.

When computing multiple regression analysis, a researcher must assume linearity in the relationships between dependent and independent variables (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). If non-linearity is present, the multiple regression analysis will result in an under-estimation of a relationship (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). If the relationship is indeed linear, the residuals have a random distribution in all ranges of the estimated value — known as homoscedasticity (Ernst & Albers, 2017; Williams et al., 2013; Yang & Yuan, 2016). Significant heteroscedasticity can lead the researcher to a distortion of findings and increase the possibility of Type I errors (Ernst & Albers, 2017; Williams et al., 2013; Yang & Yuan, 2016).

Multicollinearity is the presence of correlations between more than two predictors (Williams et al., 2013; Yang & Yuan, 2016; Yoo et al., 2014). While the correlation within the study is accepted, correlation with the other variables is not (Williams et al., 2013; Yang & Yuan, 2016; Yoo et al., 2014). Multicollinearity can lead to unstable estimates of the coefficients for individual predictors (Williams et al., 2013; Yang & Yuan, 2016; Yoo et al., 2014). Overinflating the standard errors can make some of the variables statistically insignificant when they should be significant (Williams et al., 2013; Yang & Yuan, 2016; Yoo et al., 2014). To test for multicollinearity, I measured the predictors using the variance inflation factor.

The results of multiple regression analysis are an estimated value for each actual dependent variable value a respondent has provided. Residuals are the differences between those actual and estimated values. To test the independence of residuals, I used a normal P-P. The normal P-P allows the researcher to determine if randomly dispersal is present in the residuals and if they have a constant variance (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). The best situation is if the data points fall randomly on both sides of zero (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Errors found in in the results should be independent of the estimated dependent variable values (Ernst & Albers, 2017; Williams et al., 2013; Yang & Yuan, 2016). However, when the residuals are dependent on each other, it is usually because of the study design and not the distribution of the data (Ernst & Albers, 2017; Williams et al., 2013; Yang & Yuan, 2016).

Interpretation of Multiple Regression Analysis

To determine the fit of the model to my data, I checked the goodness-of-fit statistics produced in SPSS using the adjusted R^2, *F*-statistic, and the *p*-value of the *f*-statistic. Adjusted R^2 represents the percentage of variation between the model and the data (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). When determining the best fit between the model and the data, the desire is for a higher adjusted R^2 (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Although the R^2 measured how well the model fit my data, I checked the residual plots to cross check if the model fits the assumptions. To statistically determine how well the model fit the data, I used the *p*-value of the *f*-statistic. A p-value $< .05$ indicates the model is statistically significant in estimated in the dependent variable (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Consequently, a p-value > 0.05 signals the researcher that any changes in the independent variable will not correlate to a change in the dependent variable (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Beta (β) is the threshold level of probability of having a Type II error, and a commonly accepted value is 0.20 (Ivarsson et al., 2013, Kessler et al., 2015; Palinkas et al., 2015).

The key outputs of multiple regression analysis include for each independent variable, the beta coefficient (β), *t*-statistic, and the *p*-value of the *t*-statistic (Keith, 2014; Krzywinski & Altman, 2015; Nimon & Oswald, 2013). To determine if the relationship between the independent variables and the dependent variable is statistically significant, I checked the *p*-value for the independent variable. The *p*-value tests the null hypothesis. A *p*-value < 0.05 signals the researcher to reject the null hypothesis (Keith, 2014; Krzywinski & Altman, 2015; Nimon & Oswald, 2013). The β are standardized coefficients for independent variables (Keith, 2014; Krzywinski & Altman, 2015; Nimon & Oswald, 2013). The β represents the relative strength of association between the independent variable and the dependent variable (Keith, 2014; Krzywinski & Altman, 2015; Nimon & Oswald, 2013). Due to the standardization of the β, when compared, the researcher will know if one independent variable has a stronger association with the dependent variable than another independent variable in the same equation (Keith, 2014; Krzywinski & Altman, 2015; Nimon & Oswald, 2013).

The unstandardized coefficient (*B*) represents the extent of changes in the dependent variable if the researcher changes the independent variables (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Researchers find unstandardized coefficients useful when viewing the original units and standardized coefficients when viewing normalized units (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015).

Researchers use the standard errors (*SE*) of the standardized and unstandardized (*SE B*) coefficients to test the hypothesis and construct confidence intervals (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). Another output excepted is the 95% Confidence Interval (*CI*) for *B* (Amin et al. 2015; Keith, 2014; Krzywinski & Altman, 2015). I was able to determine from the 95% *CI* for *B* that I am 95% confident the interval contains the population mean.

Study Validity

This quantitative correlation study was a nonexperimental design and threats to external and internal validity were not applicable. Researchers seek external and internal validity when exploring causal relationships in experimental or quasi-experimental designs. However, threats to statistical conclusion validity, which inflates the Type I error and Type II error rates, are of concern in nonexperimental designs. To reduce the threats of statistical conclusion validity, (a) the survey instrument selected for this study was reliable, (b) I conducted a pre-study power analysis with G*Power 3 to determine appropriate sample size, and (c) avoided violation of the assumptions for multiple regression analysis.

Statistical Conclusion Validity

To ensure statistical conclusion validity researchers must make sure they use, (a) appropriate sampling methods, (b) correct statistical tests, and (c) ensure the reliability of the instrument (Anestis, Anestis, Zawilinski, Hopkins, & Lilienfeld, 2014; Becker, Rai, Ringle, & Völckner, 2013; Stantchev, Colomo-Palacios, Soto-Acosta, & Misra, 2014). Unobserved heterogeneity biases can lead to Type I and Type II errors and are a threat to statistical conclusion validity (Anestis et al., 2014; Becker et al., 2013; Stantchev et al., 2014). However, heterogeneous samples can cause high standard errors, low effect sizes, and influences to the power of tests (Anestis et al., 2014; Becker et al., 2013; Stantchev et al., 2014). The three conditions I covered in this study were: (a) reliability of the instrument, (b) data assumptions, and (c) sample size.

Reliability of the instrument. To determine the reliability of the data collection instrument used in this study, I conducted an internal reliability check to see how close the Cronbach's alpha reliability coefficients reported in the Data Collection Instrumentation section of this study against my calculated reliability coefficients. Researchers prefer a Cronbach's alpha reliability coefficient greater than 0.70; however, coefficients equal to 0.70 are acceptable (Bonett & Wright, 2015; Dunn, Baguley, & Brunsden, 2014; Nardelli et al., 2015). After data collection, I computed the Cronbach's alpha values using IBM SPSS Statistics Program, version 24. Within SPSS there is a procedure labeled as Analyze/Scale/Reliability Analysis, which I used to calculate the Cronbach's alpha reliability coefficients. Once I calculate the Cronbach's alpha values, I reported the results in Section 3, Presentation of Findings of this study.

In this study, I modified the survey instrument to provide clarity for the respondents, and I acknowledge the risks with altering this survey; therefore, the results are to be judged with caution. While the changes did not seem to affect the outcomes of this study; the implications could have invalidated reliability and validity. When changing the wording, dropping items from the instrument, changing available responses, changing the wording from negative or positive, or changing the language the researcher risks losing the reliability and validity of the original survey. I would recommend future researchers use the original survey instrument in their original state without any changes to preserve reliability and validity.

Data assumptions. Some of the typical data assumptions are; normality, homogeneity of variances, and statistical independence. Data normality refers to data that has a normal distribution (Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). Normality is required for multiple regression and to test this assumption; I used a Normal P-P. Data homogeneity of variances refers to the same variance exists in data from multiple groups. Unequal variances lead to false positives and lead to incorrect conclusions. To test for homogeneity of variances, I used the *F*-test to compare the variation in any two data sets. Statistical independence of data elements means there is not a relationship between the data elements. To check for statistical independence, I performed a Chi-Square test for association.

Sample size. To control bias and ensure accuracy the sample must be the appropriate size (Hawkins, Gallacher, & Gammell, 2013; Hoffman, 2013; Williams et al., 2013). Controlling Type I and Type II errors are residual effects of choosing the appropriate sample size (Button et al., 2013; Hawkins et al., 2013; Williams et al., 2013). An additional result of choosing the appropriate sample size is achieving the correct effect size, power level, and confidence level (Button et al., 2013). Tarhan & Yilmaz (2014) states a researcher must increase the sample size beyond the minimum pre-study size to better generalize the findings to similar populations.

The primary barrier to sample building is invalidated reliability and validity. When changing the wording, dropping items from the instrument, changing possible responses, changing the wording from negative or positive, or changing the language the researcher risks losing the reliability and validity of the original survey.

Statistical tests. To ensure statistical conclusion validity, the researcher must choose the appropriate statistical test. Researchers use multiple regressions analysis to respond to research questions, which involve multiple independent variables and a single dependent variable (Baird & Bieber, 2016; Keith, 2014; Mertler & Reinhart, 2017). When researchers use multiple regressions analysis, the researcher must also make a few assumptions (Baird & Bieber, 2016; Keith, 2014; Mertler & Reinhart, 2017). The Data Analysis section of this study address the assumptions the researcher must address when using multiple regressions analysis.

Bootstrapping is a statistical technique of resampling the original sample and is a substitute for inferential statistics (Elvarsson, Taylor, Trenkel, Kupca, & Stefansson, 2014). The bootstrapping technique requires the researcher to run hundreds of calculations, and because of the volume of work, it is best to use a computer program to assist the researcher (Elvarsson et al., 2014). When the researcher runs the bootstrapping calculations, duplication of some of the data points occurs while replacement of others will occur (Elvarsson et al., 2014). However, researchers usually use bootstrapping when sample sizes are less than 40 data elements (Elvarsson et al., 2014). I used the bootstrapping technique to resample the data to address assumption violations since my sample size resulted in 23 complete responses instead of the 77 expected responses.

Transition and Summary

In Section 2, I outlined a methodical process for participant selection, research collection, research methods and design, and data analysis. Additional material presented in Section 2 includes (a) a detailed view of the data collection techniques and survey instrument, (b) the purpose statement, (c) the role of the researcher, (d) the population and sampling method, (e) ethical research, (f) the survey instruments, (g) research analysis methods, (h) research reliability, and (i) study validity. The information included in Section 3 contains (a) a presentation of findings, (b) the application of research findings to professional practice, (c) implications of the study for social change, (d) recommendations for action and future research, (f) reflections, (g) a summary, (h) and study conclusions.

Section 3: Application to Professional Practice and Implications for Change

Introduction

The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. The independent variables were software development team structure, ambiguity, and volatility in software specifications. The dependent variable was software project failure. The model was significantly able to predict software project failure, $F(3,19) = 10.838$, $p < .001$, $R^2 = 0.631$. The R^2 (.631) value indicated approximately 63% of variations in software project failure were accounted for by combined predictor variables. Software development team structure was the only statistically significant predictor ($t = 2.762$, $p = .012$). However, readers are to judge the results with caution due to the small sample size and instrument alteration.

Presentation of the Findings

In this section, I discuss the reliability of the variables, testing of the assumptions, descriptive statistics, and interpretation of the findings. I conclude with a concise summary. I used the bootstrapping technique to resample the data to address assumption violations because my sample size resulted in 23 complete responses instead of the 77 expected responses (see Elvarsson et al., 2014). Low sample sizes often result in low statistical power and could imply that any statistically significant finding will not reflect a true effect (Button et al., 2013). Where appropriate, I employed bootstrapping 95% confidence intervals using 2,000 samples to address the possible impact of assumption violations.

Reliability of the Variables

To measure how close the relationship was in the constructs, I used Cronbach's alpha to measure the internal consistency. The alpha coefficient for the four constructs was > .700, suggesting the items had relatively high internal consistency (Table 2). Alpha coefficients of .700 or higher are desirable; however, an alpha coefficient with .600 is the lowest acceptable threshold (Aslan, Cinar, & Yavuz, 2012; Field, 2013).

Table 2

Reliability Statistics for Study Constructs

Variables	Cronbach's alpha
Software Development Team Structure	.807
Ambiguity	.767
Volatility in Software Specifications	.790
Software Project Failure	.694

Note. $N = 23$.

Tests of Assumptions

I evaluated the assumptions of outliers, normality, linearity, multicollinearity, and independence of residuals. I used bootstrapping with 2,000 samples to focus on the impact of possible assumption violations.

Multicollinearity. I evaluated multicollinearity by viewing the correlation coefficients among the predictor variables. All bivariate correlations were small to medium (Table 3); therefore, the violation of the assumption of multicollinearity was not evident.

Table 3

Correlation Coefficients Among Study Predictor Variables

Variable	Software Development Team	Ambiguity	Volatility in Software
Software Development Team	1.00	.415	.353
Ambiguity	.415	1.00	.548
Volatility in Software	.353	.548	1.00

Note. *N* = 23.

Outliers, normality, linearity, homoscedasticity, and independence of residuals. I evaluated outliers, normality, linearity, homoscedasticity, and independence of residuals by examining the normal probability plot (P-P) of the regression standardized residual (Figure 1) and the scatterplot of the standardized residuals (Figure 2). My examination revealed no major violations of outliers, linearity, homoscedasticity, and independence of residuals. However, there was a violation of the assumption of normality.

Figure 1 depicts the normality results of the distribution around the fit line. My study was positively skewed; Figure 1 shows an even distribution along the fitted distribution line, with most points deviating from the fitted distribution line. Eight of the 23 points touch the fitted distribution line, and the other 15 points flowed along the fitted distribution line but deviated from normality. The tendency of the points to deviate from a reasonably straight line, diagonal from the bottom left to the top right, provided evidence that the assumption of normality had been violated (see Amin et al., 2015; Ernst & Albers, 2017; Williams et al., 2013). I examined the scatterplot of the standardized residuals and found there was no pattern to support the assumption being met. I used SPSS to compute 2,000 bootstrapping samples to combat any possible influence of assumption violations, and reported 95% confidence intervals based on the bootstrap samples where appropriate.

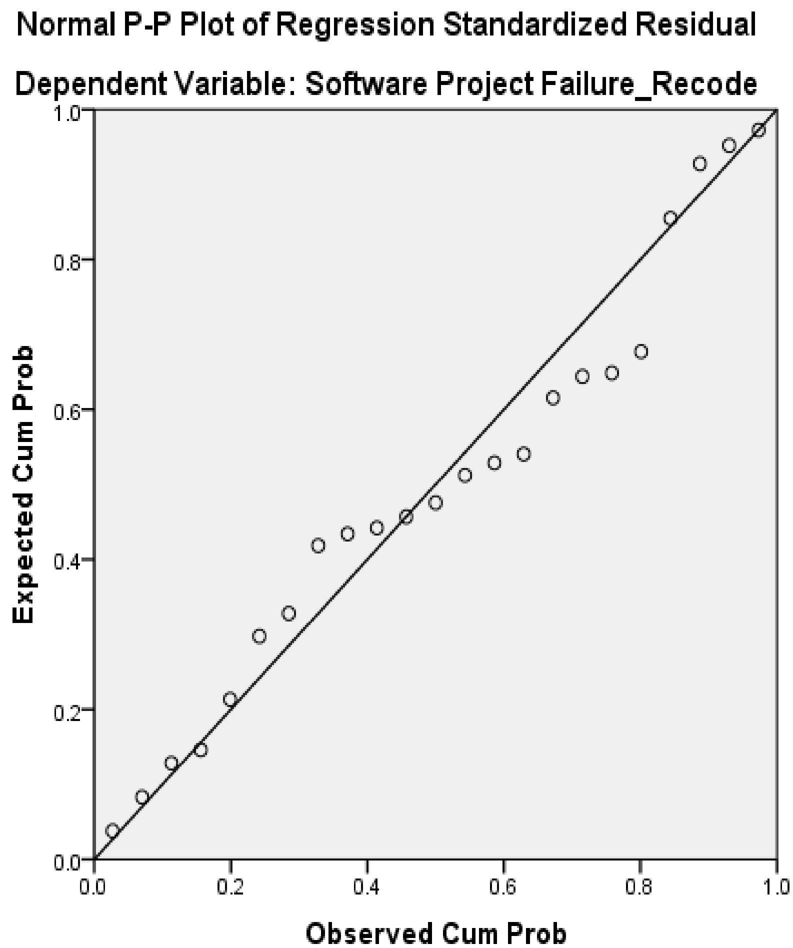

Figure 1. Normal P-P plot of the regression standardized residuals.

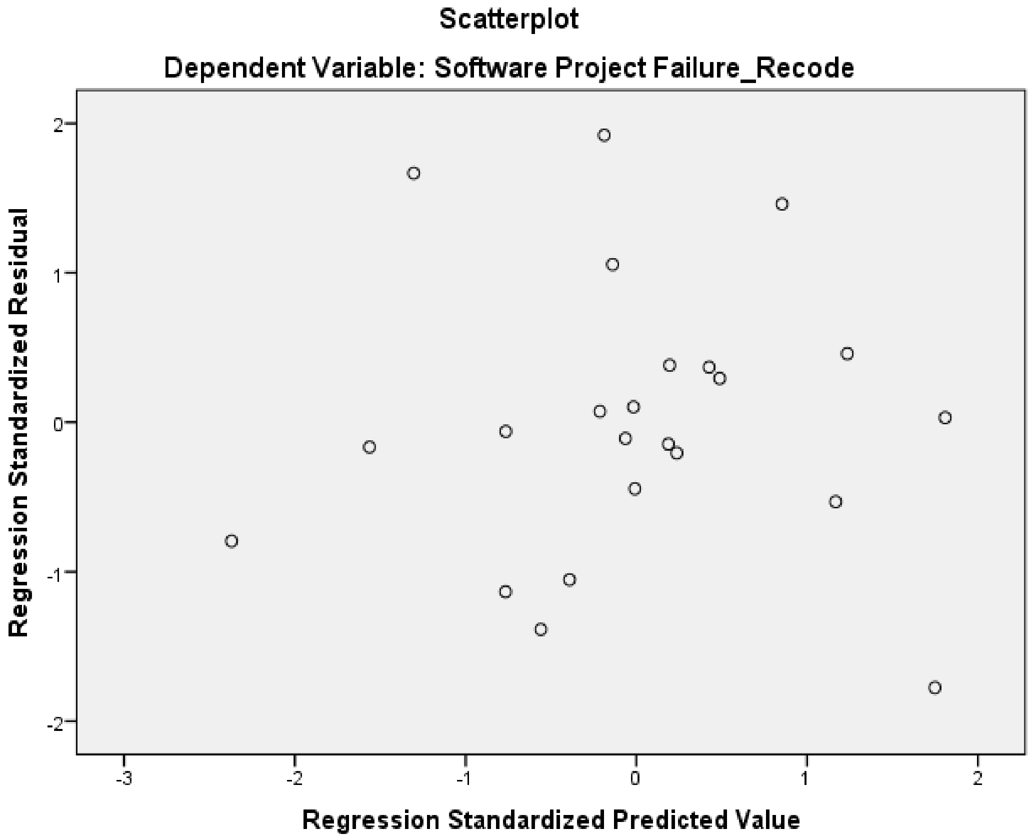

Figure 2. Scatterplot of the standardized residuals.

Descriptive Statistics

I received 28 responses to my survey. I eliminated five records due to missing data, resulting in 23 records for analysis. The target population included approximately 150 software development company owners contracted by the USAF, and the expected response rate was 77 responses. Of the expected 77 responses, 23 valid responses were received. Low response rates can affect the demographic representativeness of age, race, gender, income, and education (Holbrook, Krosnick, & Pfent, 2007). The small sample size resulted in nonresponse bias in the study. Nonresponse bias is an error that results from an insufficient number of responses to the study survey by the target population

(Fincham, 2008). Nonresponse bias negatively affects the reliability and validity of the study findings (Fincham, 2008). Because nonresponse bias was present in the survey data due to the small sample size, the reader should judge the study results with caution. Table 4 presents descriptive statistics of the independent and dependent variables.

Table 4

Descriptive Statistics for Independent and Dependent Variables

Variable	*M*	*M 95% Bootstrap CI*	*SD*	*SD 95% Bootstrap CI*
Software Development Team Structure	34.870	[33.088, 36.608]	4.404	[2.946, 5.347]
Ambiguity	23.652	[21.522, 25.609]	5.096	[3.621, 6.149]
Volatility in Software Specifications	16.739	[14.261, 19.608]	6.412	[3.843, 8.299]
Software Project Failure	31.696	[29.348, 33.826]	5.700	[3.914, 7.089]

Note. *N* = 23. Bootstrap results are based on 2000 bootstrap samples.

Inferential Results

Of the expected 77 responses, I received only 23 valid responses. The small sample size resulted in nonresponse bias in the study. Nonresponse to the survey can contribute to an increase in the total variance of estimates and can introduce bias in estimates (Särndal & Lundström, 2005). Although I used bootstrapping of 2000 samples in the analysis of the data, the small sample size resulted in significant underpowering and nonresponse bias.

I used standard multiple linear regression to examine software development team structure, ambiguity, and volatility in software specifications in predicting software project failure. The independent variables were software development team structure, ambiguity, and volatility in software specifications. The dependent variable was software project failure. The null hypothesis was there is no relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. The alternative hypothesis was there is a relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. I conducted a preliminary analysis to assess whether the assumptions of outliers, normality, linearity, multicollinearity, and independence of residuals were met; no serious violations were noted. The model as whole was able to significantly predict software project failure, $F(3,19) = 10.838$, $p < .001$, $R^2 = 0.631$. The R^2 (.631) value indicated approximately 63% of variations in software project failure accounted for the linear combination of the predictor variables (software development team structure, ambiguity, and volatility in software specifications). In the final model, software development team structure was the only statistically significant predictor ($t = 2.762$, $p = .012$). Ambiguity and volatility in software specifications did not explain any significant variation in software project failure.

Because my sample size resulted in 23 complete responses instead of the 77 expected responses, I employed bootstrapping 95% confidence intervals using 2,000 samples with the intent to address the possible impact of assumption violations. However, the extremely small original data set introduced large variability in the confidence interval: Software Development Team Structure (085, 1.018); Ambiguity (.118, .777); Volatility in Software Specifications (.075, .713). Although the intent of bootstrapping was to assign a measure of accuracy to my original sample, having an extremely small original dataset had the inverse effect (see Hall, 2013). As the number of resampled data sets decreases, the result is the introduction of more variability into the confidence interval estimation (Haukoos & Lewis, 2005). Table 5 presents a summary of the regression analysis.

Table 5

Multiple Regression Analysis Summary for Predictor Variables

Variable	*B*	*SE B*	*B* 95% Bootstrap CI	*β*	*t*	*p*
Software Development Team Structure	.555	.201	[.085, 1.018]	.429	2.762	.012
Ambiguity	.285	.194	[-.118, .777]	.255	1.468	.158
Volatility in Software Specifications	.287	.150	[-.075, .713]	.322	1.908	.072

Note. *N* = 23. Bootstrap results are based on 2000 bootstrap samples.

Software development team structure. The positive slope for software development team structure (0.555) as a predictor of software project failure indicated there was a .555 increase in software project failure for each additional one-unit increase in software development team structure. In other words, software project failure tended to increase as software development team structure increased.

Analysis summary. The purpose of this quantitative correlational study was to examine the relationship between software development team structure, ambiguity, and volatility in software specifications and software project failure. I used multiple linear regression to examine the ability of software development team structure, ambiguity, and volatility in software specifications to predict software project failure. I assessed the assumptions surrounding multiple regression and noted a violation of the assumption of normality. The model including software development team structure, ambiguity, and volatility in software specifications was able to significantly predict software project failure, $F(3,19) = 10.838$, $p < .001$, $R^2 = 0.631$. Software development team structure provided useful predictive information about software project failure. The bootstrap computation relies on the sampling distribution when calculating confidence intervals, but using extremely small samples will interfere with the validity of the computation (Elvarsson et al., 2014). The small original data set was less likely to represent the intended population. Extremely small data sets make it difficult to compute valid confidence intervals (Meeker & Escobar, 2014). A small data set may not provide demographic representativeness of age, race, gender, income, and education (Holbrook et al., 2007). The conclusion from this analysis was software development team structure was significantly associated with software project failure.

Theoretical discussion of findings. Software project success is a multifaceted construct, but not all factors are present in every project (McLeod et al., 2012). The Air Force definition of software project success includes software projects delivered within specifications, within budget, and on time (OUSD[AT&L], 2015). Other researchers also define successful projects in a similar method; but include team structure, ambiguity, customer satisfaction, code verification, and engineering practices as possible other variables (Lindsjørn et al., 2016). Reducing the effects of ambiguity, volatility, and team structure can have a significant impact on the success of software development projects (AlMarzouq et al., 2015). Galbraith (1974) stated quality information could eliminate ambiguity, volatility, and improve productivity to increase the chances of software project success. Honoring the team's structure leads to activities, which enhance team cohesiveness and improve team performance (Cheruvelil et al., 2014).

The results of this study support Galbraith's (1973) theory of OIP. Galbraith (1973) stated team structure refers to how organizations implement structural mechanisms to enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty. Organizations that have a hierarchical structure have a lower information-processing capability than organizations that have a flatter organizational structure (Galbraith, 1973). Software development companies with a loosely defined team structure employ less knowledge sharing; whereas companies with a well-defined, flatter team structure can process information more efficiently. Uncertainty in a software development teams often materializes because the team's structure does not promote an environment of information sharing.

While Galbraith addresses team structure from the technical information-processing aspect, other authors suggest management structure influences team structure. The importance of having a proper team structure applies to the teams working on information-intensive tasks such as software development teams (Açıkgöz & Günsel, 2016). Traditionally, researchers have viewed the software development process primarily from a technical perspective; but the emerging view on software development process centers on the socio-technical aspects of the process, indicating the organizational and human aspects both play critical roles (Too & Weaver, 2014). Although the effectiveness of the team's ability to process information depends on the structure of that team, the overarching management structure also influences the team structure.

Applications to Professional Practice

The findings of this study have applicability to the software development professional practice by providing software development company owners with information to better safeguard against software project failure. I chose three predictor variables; and then analyzed the predictor variables with the dependent variable, software project failure. I discovered of the three-predictor variables only software development team structure was significant in predicting software project failure. Managers who understand the factors of software success position themselves to manage those factors to lessen the impact, and better assure the chances of software project success (McLeod et al., 2012). Management of the software project success factors is not a passive activity but rather an active endeavor, which requires constant monitoring and adjustment (Müller & Jugdev, 2012).

The way software development company owners design software development teams will lessen the uncertainty, hence positioning the software development project for success. The organizational culture, location, and scope of the project can vary the software development team's composition (Drury-Grogan, 2014; Hoch & Kozlowski, 2014). Team structure is a multifaceted topic, with the organizational structure being one of the tenants heavily involved in the decision of what best suits the team structure. By properly organizing the structure of the software development team, software development company owners can improve the company's profitability, by ensuring more successful software development projects. With the dependence companies and the military have on software and the rising cost of software, a software project failure can be financially detrimental to their viability (Kaur & Sengupta, 2013). The cost of software is continuously rising, which motivates companies and the military to purchase services from qualified software development companies who have structured their software development teams for success.

Implications for Social Change

The implications for positive social change include the potential to include higher success rates for USAF software development projects. Better USAF software means better weaponry for the protection of society. The purpose of the U.S. warfighter is to deter the enemy before the fight begins and if a battle ensues then the U.S. warfighter can win that battle decisively. Providing better weaponry to the U.S. warfighter enable the warfighter to remain dominant on the battlefield while deterring other U.S. enemies and preventing hostile actions. An additional implication for positive social change includes the potential of the reduced tax burden to the American citizen. Reducing the failure rates of USAF software projects can reduce software project cost increases or cancellations, thus reducing excessive government spending.

Practical implications are business leaders, as well as military leaders, can apply the results of this study to gain a better understanding of methods to improve software development project success and develop strategies to improve software development team structure. Software development project failure affects organizations, companies, communities, individuals, and the economy. Putting strategies into place to improve software development team structure will increase the chances of software project success and can keep software development companies employed, hence keeping employees employed. Keeping employees employed is significant in reducing the unemployment rates and boosting the economy.

Recommendations for Action

Software is vital to the majority of businesses as well as the U.S. military. When a company’s software project fails, the company pays for the failure in time, performance, and with money (Estler et al., 2014). I recommend software development company owners (a) ensure software development team structure is flat, (b) enhance information flows and information-processing capability while employing buffers to reduce the potential effects of uncertainty, (c) employ an agile development methodology to eliminate hierarchical structures, (d) embrace virtual teams over brick and mortar teams, and (e) encourage team diversity. This recommendation extends to any organization who manages development projects, which produces a product developed by teams.

Software development company owners should concentrate on training their managers to restructure software development teams for better software development success. Project Management Institute (PMI) hosts training sessions, which specialize in structuring software development teams (Project Management Institute, 2017). Other avenues of information about software development team structure are Gartner conferences, PMI conferences, and AFCEA conferences. At these conferences, there are usually alternative sessions, which focus on the software development team structure. Through effective software development team structure, employees can assist software development company owners in producing viable products.

Using the right methodological approach like traditional, agile, extreme, or emertxe project management, for example, will assist software development leaders in defining the proper team structure for the highest chances of software project success. Adopting a traditional, agile, extreme, or emertxe project management structure depends on the project and the needs of the software development team. Traditional, agile, extreme, or emertxe project management are four of the conventional approaches to software project development, with each approach demanding a different team structure (Liu et al., 2015; Project Management Institute, 2017). Complexity, scope clarity, and uncertainty determine the management approach best suited for successful software development team structure (Brhel et al., 2015; Conforto et al., 2014; Dikert, Paasivaara, & Lassenius, 2016; Liu et al., 2015). The organizational culture, location, and scope of the project can vary the software development team's composition (Drury-Grogan, 2014; Hoch & Kozlowski, 2014). Team structure is a multifaceted topic, with the organizational structure being one of the tenants heavily involved in the decision of which best suits the team structure.

Recommendations for Further Research

A limitation of this study was the information obtained through the survey was correct at the time and does not account for future changes. Software development is dynamic and influenced by new technology, new development process, software development company maturity, and personnel capability. Software development companies continuously seek the newest process, latest technology, and best employees to give their company an edge in the expanding numbers of software development companies. Since software development companies are always changing, the survey responses, which were correct when the respondents answered the survey, may not be accurate in a future software development environment. Future research project respondents' answers will reflect the technologies, development processes, software development company maturity, and personnel capabilities of that time.

An additional limitation was the findings of the study focused on the responses received, and I could not derive the depth of understanding a qualitative study could have provided. Using the mixed method approach requires researchers to employ both quantitative and qualitative methods (O'Leary, 2017). Researchers use the qualitative method to produce results, which are naturalistic, interpretative, and a rich depth of information, while researchers use the quantitative method to collect numerical data, test and confirm hypotheses, and then generalize those findings (Leedy & Ormrod, 2015; O'Leary, 2017; Ritchie et al., 2013). Using a mixed methods research methodology to study software project success can provide the future research a rich depth of information and the numerical information to support generalization (Kemper et al., 2003).

A recommendation for further research includes increasing the variables in future studies. Software project success factors are numerous and complex and far exceed the three chosen for this study. Throughout this study, I have identified schedule, scope, client satisfaction, team satisfaction, budget overrun, time overrun, complete requirement coverage, high customer satisfaction, performance parameters, software recycle rates, and error rates as other variables which influence software project success. With so many different software project success factors, researchers could develop a comprehensive study, which could provide substantial information to software development company owners.

Furthermore, since the target population for this study was software development companies contracted by the USAF who were members of AFCEA within the Southeastern United States geographical area; I recommend further research on software project success in other geographic locations using a more extensive study population. The target population of this study is a tiny specific audience in comparison to the software development companies located within the state, country, and world. In addition, future studies could include more participants then the software development company owners this study included. The additional participants could include developers, programmers, project managers, testers, configuration managers, and others with the companies. Examining more abundant and diverse populations could add richer data to the future study.

I also have recommendations to deal with the altering of my survey instrument and the small sample size I received. In this study, I modified the survey instrument to provide clarity for the respondents, and I acknowledge the risks with altering this survey; therefore, the results are to be judged with caution. While the changes did not seem to affect the outcomes of this study; the implications could invalidate reliability and validity. When changing the wording, dropping items from the instrument, changing possible responses, changing the wording from negative or positive, or changing the language the researcher risks losing the reliability and validity of the original survey. I would recommend future researchers use the original survey instrument in their original state without any changes to preserve reliability and validity.

In addition, although I used published data collection procedures in this study, the number of samples received was only 23. The community of software development company owners contracted by the USAF who are members of AFCEA within the Southeastern United States geographical area included approximately 150 software development company owners contracted by the USAF. Of the 150 AFCEA members I sent the survey, I believed I would receive at least 77 samples. Since my sample size resulted in 23 complete responses instead of the 77 expected responses, I employed bootstrapping 95% confidence intervals using 2,000 samples, to address the possible impact of assumption violations. I would recommend future researchers use a population that quadruples the estimated minimum required sample size to ensure at least a 25% response rate.

Reflections

My experience in the Walden University DBA Doctoral program was an ordeal, which brought me to the brink of quitting. I have always been an excellent student in brick and mortar institutions. My DBA study was my first foray into a full-time web-based university and the frustrations of feeling helpless and sense of being alone in the online education process where a new experience and almost overwhelming. The Walden University environment, DBA rubric requirements, APA, IRB review process, and various faculty members contributed to my frustrations. I became frustrated because it seemed the required changes often contradicted the previous researcher’s requests.

However, I took time to reflect on my military career and determined it was no different from the military personnel evaluations systems. In the military personnel evaluations systems, it is common to go through 10 or more reviewers, each with their own opinions. It was during this period I learned to keep every variant of the evaluation because I would be able to use the same information I had previously removed. Once I made this realization, I applied the same concept to my DBA Study and found my frustration levels significantly lessened.

At several points in my time at Walden University I considered quitting. Several discussions with friends, co-workers, family, and Walden University faculty convinced me to continue with the program. Once I decided to stay with the DBA program, I decided I would focus on networking with other students and try to maintain a positive outlook. Helping other students cope with their frustrations allowed me to cope with my frustrations. In addition, I was constantly challenged to balance my church job, my work for the U.S. Air Force, building a retirement home, Walden University coursework, and family life. Through Gods will and perseverance, I succeeded.

My work with U.S. Air Force software programs resulted in me seeing some projects' fail while others succeeded. These failures and successes caused me to wonder and develop some suspicions about the cause of the failures and what was going right to result in the successes. One of my perceptions coming into this study was the experience level of the Project Manager reflected directly on the success of the software development project. However, the literature reviewed did not list experience level as one of the factors of success. I also perceived there was a single cause for software project failure. The literature reviewed suggested software project failure was usually the results of a combination of factors.

Conclusion

The success of a software project affects the budgets of the USAF as well as civilian companies. Software project success is dependent on many variables or combinations of variables. The variables of software project success are very numerous and worthwhile for further study. In this study, I found software development team structure had a relationship with software project success. However, I was surprised to find there were no relationships with volatility or ambiguity and software project failure. Although there is a relationship between software development team structure and software project failure, I believe there are additional relationships amongst the abundant software success factors.

Software programs fail a rate of 60% to 70%, resulting in U.S. companies spending over $81 billion on failed software projects annually (Eberendu, 2015). Unstable requirements of software applications are a direct cause in 70% of the failed software systems (Khan, Khalid, & Haq, 2013). Software costs are expected to rise significantly in future years (Melo, Tavares, Marinho, Nogueira, & Sousa, 2015). Civilian markets as well as military markets experience massive software program failures and must focus on factors, which contribute to software project success.

References

Açıkgöz, A., & Günsel, A. (2016). Individual creativity and team climate in software development projects: The mediating role of team decision processes. *Creativity and Innovation Management*, *25*, 445-463. doi:10.1111/caim.12173

Ales, G., & Marko, P. (2017). Missing value imputation using contemporary computer capabilities: An application to financial statements data in large panels. *Economic and Business Review*, *19*, 97-119. doi:10.15458/85451.38

AlMarzouq, M., Grover, V., & Thatcher, J. B. (2015). Taxing the development structure of open source communities: An information processing view. *Decision Support Systems*, *80*, 27-41. doi:10.1016/j.dss.2015.09.004

Al-Saiyd, N. A. (2016). A multistep approach for managing the risks of software requirements volatility. *Computer and Information Science*, *9*(1), 101-112. doi:10.5539/cis.v9n1p101

Amin, M., Akbar, A., & Manzoor, M. A. (2015). Fitting regression model with some assumptions for the identification of cotton yield factors. *Pakistan Journal of Life and Social Sciences, 13,* 86-90. Retrieved from http://www.pjlss.edu.pk

Anestis, M. D., Anestis, J. C., Zawilinski, L. L., Hopkins, T. A., & Lilienfeld, S. O. (2014). Equine-related treatments for mental disorders lack empirical support: A systematic review of empirical investigations. *Journal of Clinical Psychology*, *70*, 1115-1132. doi:10.1002/jclp.22113

Aslan, I., Cinar, O., & Yavuz, S. (2012). Analyzing survey questions according to departments and checking the reliability of them for an organization.

International Journal of Business and Management, *7*, 185-201. doi:10.5539/ijbm.v7n6p185

Aziz, R. A., & Wong, B. (2015). The interplay between requirements relationships knowledge and requirements change towards software project success: An assessment using partial least square (PLS). *Procedia Computer Science*, *46*, 732-741. doi:10.1016/j.procs.2015.02.138

Baird, G. L., & Bieber, S. L. (2016). The goldilocks dilemma: Impacts of multicollinearity--A comparison of simple linear regression, multiple regression, and ordered variable regression models. *Journal of Modern Applied Statistical Methods*, *15*(1), 332-357. doi:10.22237/jmasm/1462076220

Barker, C., & Pistrang, N. (2016). *Research methods in clinical psychology: An introduction for students and practitioners* (3rd ed.) [Wiley Online Library]. doi:10.1002/9781119154082

Bayona-Oré, S., Calvo-Manzano, J. A., Cuevas, G., & San-Feliu, T. (2014). Critical success factors taxonomy for software process deployment. *Software Quality Journal*, *22*, 21-48. doi:10.1007/s11219-012-9190-y

Becker, J. M., Rai, A., Ringle, C. M., & Völckner, F. (2013). Discovering unobserved heterogeneity in structural equation models to avert validity threats. *MIS Quarterly*, *37*, 665-694. Retrieved from http://www.misq.org

Bernard, H. R. (2011). *Research methods in anthropology: Qualitative and quantitative approaches*. Lanham, MD: Rowman & Littlefield Publishers, Inc.

Bezzina, F., & Saunders, M. (2014). The pervasiveness and implications of statistical misconceptions among academics with a special interest in business research methods. *Electronic Journal of Business Research Methods, 12,* 107-120. Retrieved from http://www.ejbrm.com

Bishop, P. A., & Herron, R. L. (2015). Use and misuse of the Likert item responses and other ordinal measures. *International Journal of Exercise Science, 8*, 297-302. Retrieved from https://digitalcommons.wku.edu/ijes

Bojcic, J. L., Sue, V. M., Huon, T. S., Maletis, G. B., & Inacio, M. C. (2014). Comparison of paper and electronic surveys for measuring patient-reported outcomes after anterior cruciate ligament reconstruction. *Permanente Journal*, *18*(3), 22-26. doi:10.7812/tpp/13-142

Bonett, D. G., & Wright, T. A. (2015). Cronbach's alpha reliability: Interval estimation, hypothesis testing, and sample size planning. *Journal of Organizational Behavior, 36*(1), 3-15. doi:10.1002/job.1960

Bornstein, M., Jager, J., & Putnick, D. (2013). Sampling in developmental science: Situations, shortcomings, solutions, and standard. *Developmental Review, 33*, 357-370. doi:10.1016/j.dr.2013.08.003

Brhel, M., Meth, H., Maedche, A., & Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology*, *61*, 163-181. doi:10.1016/j.infsof.2015.01.004

Bryman, A. (2012). *Social research methods* (5th ed.). Oxford, UK: Oxford University Press.

Button, K., Ioannidis, J., Mokrysz, C., Nosek, B., Flint, J., Robinson, E., & Munafo, M.

(2013). Power failure: Why small sample size undermines the reliability of neuroscience. *Nature Reviews Neuroscience, 14*, 365-376. doi:10.1038/nrn3475

Chen, H. X. (2012). *Approaches to quantitative research* [ProQuest Ebook Central]. Retrieved from http://ebookcentral.proquest.com/lib/waldenu/detail.action?docID=1620892

Cheruvelil, K. S., Soranno, P. A., Weathers, K. C., Hanson, P. C., Goring, S. J., Filstrup, C. T., & Read, E. K. (2014). Creating and maintaining high-performing collaborative research teams: The importance of diversity and interpersonal skills. *Frontiers in Ecology and the Environment*, *12*, 31-38. doi:10.1890/130001

Christophe, F., Mokammel, F., Coatanéa, E., Nguyen, A., Bakhouya, M., & Bernard, A. (2014). A methodology supporting syntactic, lexical and semantic clarification of requirements in systems engineering. *International Journal of Product Development*, *19*, 173-190. doi:10.1504/ijpd.2014.062973

Cohen, J. (1992). Statistical power analysis. *Current Directions in Psychological Science*, *1*, 98-101. doi:10.1111/1467-8721.ep10768783

Colomo-Palacios, R., Casado-Lumbreras, C., Soto-Acosta, P., García-Peñalvo, F. J., & Tovar, E. (2014). Project managers in global software development teams: A study of the effects on productivity and performance. *Software Quality Journal*, *22*, 3-19. doi:10.1007/s11219-012-9191-x

Conforto, E. C., Salum, F., Amaral, D. C., da Silva, S. L., & de Almeida, L. F. M. (2014). Can agile project management be adopted by industries other than software development?. *Project Management Journal*, *45*(3), 21-34.

doi:10.1002/pmj.21410

Corbin, J., & Strauss, A. (2015). *Basics of qualitative research: Techniques and procedures for developing grounded theory* (4th Ed). Thousand Oaks, CA: SAGE Publications, Inc.

da Silva, F. Q., França, A. C. C., Suassuna, M., de Sousa Mariz, L. M., Rossiley, I., de Miranda, R. C., …Espindola, E. (2013). Team building criteria in software projects: A mix-method replicated study. *Information and Software Technology*, *55,* 1316-1340. doi:10.1016/j.infsof.2012.11.006

Davis, G. F. (2015). Celebrating organization theory: The after-party. *Journal of Management Studies*, *52,* 309-319. doi:10.1111/joms.12094

De Vaus, D. (2013). *Surveys in social research* (6th ed.) [Taylor & Francis eBooks]. doi:10.4324/9780203519196

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, *119,* 87-108. doi:10.1016/j.jss.2016.06.013

Drury-Grogan, M. L. (2014). Performance on agile teams: Relating iteration objectives and critical decisions to project management success factors. *Information and Software Technology*, *56*, 506-515. doi:10.1016/j.infsof.2013.11.003

Dunn, T. J., Baguley, T., & Brunsden, V. (2014). From alpha to omega: A practical solution to the pervasive problem of internal consistency estimation. *British Journal of Psychology, 105,* 399-412. doi:10.1111/bjop.12046

Dutot, V., Bergeron, F., & Raymond, L. (2014). Information management for the internationalization of SMEs: An exploratory study based on a strategic alignment perspective. *International Journal of Information Management, 34*, 672-681. doi:10.1016/j.ijinfomgt.2014.06.006

Ebad, S. A. (2017). Towards measuring software requirements volatility: A retrospective analysis. *Malaysian Journal of Computer Science, 30*, 99-116. Retrieved from http://mjcs.fsktm.um.edu.my/

Ebbers, J. J., & Wijnberg, N. M. (2017). Betwixt and between: Role conflict, role ambiguity and role definition in project-based dual-leadership structures. *Human Relations, 70*, 1342-1365. doi:10.1177/0018726717692852

Eberendu, A. C. (2015). Evaluation of software project failure and abandonment in tertiary institutions in Nigeria. *Information and Knowledge Management, 5*(4), 29-40. Retrieved from http://iiste.org/

Ellis, T. J., & Levy, Y. (2010, June). *A guide for novice researchers: Design and development research methods.* Paper presented at Informing Science & IT Education Conference (InSITE), Cassino, Italy. Retrieved from http://proceedings.informingscience.org/ InSITE2010/InSITE10p107-118Ellis725.pdf

Elvarsson, B., Taylor, L., Trenkel, V. M., Kupca, V., & Stefansson, G. (2014). A bootstrap method for estimating bias and variance in statistical fisheries modelling frameworks using highly disparate datasets. *African Journal of Marine Science, 36*, 99-110. doi:10.2989/1814232x.2014.897253

Ernst, A. F., & Albers, C. J. (2017). Regression assumptions in clinical psychology research practice-A systematic review of common misconceptions. *PeerJ*, *5*, e3323. doi:10.7717/peerj.3323

Estler, H. C., Nordio, M., Furia, C. A., Meyer, B., & Schneider, J. (2014). Agile vs. structured distributed software development: A case study. *Empirical Software Engineering*, *19*, 1197-1224. doi:10.1007/s10664-013-9271-y

Faul, F., Erdfelder, E., Buchner, A., & Lang, A.-G. (2009). Statistical power analyses using G*Power 3.1: Tests for correlation and regression analyses. *Behavior Research Methods*, *41*, 1149-1160. doi:10.3758/brm.41.4.1149

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*, 175-191. doi:10.3758/bf03193146

Fernandes, G., Ward, S., & Araújo, M. (2014). Developing a framework for embedding useful project management improvement initiatives in organizations. *Project Management Journal*, *45*(4), 81-108. doi:10.1002/pmj.21441

Field, A. (2013). *Discovering statistics using SPSS*. Thousand Oaks, CA: Sage.

Fincham, J. E. (2008). Response rates and responsiveness for surveys, standards, and the journal. *American Journal of Pharmaceutical Education*, *72*, 43. doi:10.5688/aj720243

Foreman, V. L., Favaró, F. M., Saleh, J. H., & Johnson, C. W. (2015). Software in military aviation and drone mishaps: Analysis and recommendations for the investigation process. *Reliability Engineering & System Safety, 137*, 101–111. doi:10.1016/j.ress.2015.01.006

Furlan Matos Alves, M. W., Lopes de Sousa Jabbour, A. B., Kannan, D., & Chiappetta Jabbour, C. J. (2017). Contingency theory, climate change, and low-carbon operations management. *Supply Chain Management: An International Journal*, *22*, 223-236. doi:10.1108/SCM-09-2016-0311

Galbraith, J. R. (1973). *Designing complex organizations*. Reading, MA: Addison-Wesley Longman.

Galbraith, J. R. (1974). Organization design: An information processing view. *Interfaces*, *4*(3), 28-36. doi:10.1287/inte.4.3.28

Ghobadi, S. (2015). What drives knowledge sharing in software development teams: A literature review and classification framework. *Information & Management*, *52*, 82-97. doi:10.1016/j.im.2014.10.008

Giezen, M., Salet, W., & Bertolini, L. (2015). Adding value to the decision-making process of mega projects: Fostering strategic ambiguity, redundancy, and resilience. *Transport Policy*, *44,* 169-178. doi:10.1016/j.tranpol.2015.08.006

Gregory, R. W., & Keil, M. (2014). Blending bureaucratic and collaborative management styles to achieve control ambidexterity in IS projects. *European Journal of Information Systems*, *23*, 343-356. doi:10.1057/ejis.2013.3

Gurbin, T. (2015). Enlivening the machinist perspective: Humanising the information processing theory with social and cultural influences. *Procedia-Social and Behavioral Sciences*, *197*, 2331-2338. doi:10.1016/j.sbspro.2015.07.263

Gutiérrez, E. (2014). Managing ambiguity when evaluating and selecting new ideas in project portfolio management. *International Journal of Innovation and*

Technology Management, *11*, 1450030. doi:10.1142/s0219877014500308

Hagen, C., & Sorenson, J. (2013). Delivering military software affordably. *Defense AT&L, 42*(2), 30-34. Retrieved from http://www.dau.mil/

Haggerty, N. (2000). Understanding the link between IT project manager skills and project success research in progress. *Proceedings of the 2000 ACM SIGCPR Conference on Computer Personnel Research, Chicago, IL*, 192-195. doi:10.1145/333334.333398

Håkansson, A. (2013, January). Portal of research methods and methodologies for research projects and degree projects. In *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'13)*. Symposium conducted at The 2013 World Congress in Computer Science, Computer Engineering, and Applied Computing WORLDCOMP 2013, Las Vegas, NV.

Hall, P. (2013). *The bootstrap and edgeworth expansion*. New York City, NY: Springer Science & Business Media.

Haukoos, J. S., & Lewis, R. J. (2005). Advanced statistics: Bootstrapping confidence intervals for statistics with "difficult" distributions. *Academic Emergency Medicine, 12,* 360-365. doi:10.1197/j.aem.2004.11.018

Hawkins, D., Gallacher, E., & Gammell, M. (2013). Statistical power, effect size and animal welfare: Recommendations for good practice. *Animal Welfare, 22,* 339-344. doi:10.7120/09627286.22.3.339

Henseler, J. (2017). Bridging design and behavioral research with variance-based structural equation modeling. *Journal of Advertising*, *46*, 178-192. doi:10.1080/00913367.2017.1281780

Hoch, J. E., & Kozlowski, S. W. (2014). Leading virtual teams: Hierarchical leadership, structural supports, and shared team leadership. *Journal of Applied Psychology*, *99*, 390-403. doi:10.1037/a0030264

Hoegl, M., & Gemuenden, H. G. (2001). Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization Science*, *12*, 435-449. doi:10.1287/orsc.12.4.435.10635

Hoffman, C. (2013). Research articles in dreaming: A review of the first 20 years. *Dreaming*, *23*, 216-231. doi:10.1037/a0032905

Holbrook, A., Krosnick, J. A., & Pfent, A. (2007). The causes and consequences of response rates in surveys by the news media and government contractor survey research firms. *Advances in Telephone Survey Methodology (pp.* 499-528). doi:10.1002/9780470173404.ch23

Hornstein, H. A. (2015). The integration of project management and organizational change management is now a necessity. *International Journal of Project Management*, *33*, 291-298. doi:10.1016/j.ijproman.2014.08.005

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in Human Behavior*, *51*, 915-929. doi:10.1016/j.chb.2014.10.046

Ingham-Broomfield, R. (2014). A nurses' guide to quantitative research. *Australian Journal of Advanced Nursing, 32*(2), 32-38. Retrieved from http://www.ajan.com.au

Islam, S., Mouratidis, H., & Weippl, E. R. (2014). An empirical study on the implementation and evaluation of a goal-driven software development risk management model. *Information and Software Technology*, *56*, 117-133. doi:10.1016/j.infsof.2013.06.003

Ivarsson, A., Andersen, M. B., Johnson, U., & Lindwall, M. (2013). To adjust or not adjust: Nonparametric effect sizes, confidence intervals, and real-world meaning. *Psychology of Sport and Exercise*, *14*, 97-102. doi:10.1016/j.psychsport.2012.07.007

Jiang, J. J., Chang, J. Y., Chen, H. G., Wang, E. T., & Klein, G. (2014). Achieving IT program goals with integrative conflict management. *Journal of Management Information Systems*, *31*(1), 79-106. doi:10.2753/mis0742-1222310104

Johnsen, C. G. (2016). Romanticizing the market: Jean-Jacques Rousseau and the pedagogy of post-bureaucratic management. *Culture and Organization*, *22*, 88-105. doi:10.1080/14759551.2015.1092970

Joslin, R., & Müller, R. (2015). Relationships between a project management methodology and project success in different project governance contexts. *International Journal of Project Management*, *33*, 1377-1392. doi:10.1016/j.ijproman.2015.03.005

Kahn, R. L., Wolfe, D. M., Quinn, R. P., Snoek, J. D., & Rosenthal, R. A. (1964). *Organizational stress: Studies in role conflict and ambiguity*. New York, NY: John Wiley Oxford, England: John Wiley

Kandjani, H., Tavana, M., Bernus, P., Wen, L., & Mohtarami, A. (2015). Using extended axiomatic design theory to reduce complexities in global software development projects. *Computers in Industry*, *67*, 86-96. doi:10.1016/j.compind.2014.10.008

Kannan, M., Mahalakshmi, V., & Sujatha, J. (2014). A review on requirement analysis in software companies. *International Journal of Applied Engineering Research, 9,* 20637-20647. Retrieved from http://www.ripublication.com/

Kaur, R., & Sengupta, J. (2011). Software process models and analysis on failure of software development projects. *International Journal of Scientific & Engineering Research, 2*(2), 1-4. Retrieved from http://www.ijser.org/

Keith, T. Z. (2014). *Multiple regression and beyond: An introduction to multiple regression and structural equation modeling* (2nd ed.). New York, NY: Routledge.

Kemper, E. A., Stringfield, S., & Teddlie, C. (2003). *Major issues and controversies in the use of mixed methods in the social and behavioral sciences. Handbook of Mixed Methods in Social & Behavioral Research* (pp. 273-296). Thousand Oaks, CA: SAGE Publications, Inc.

Kessler, L. G., Barnhart, H. X., Buckler, A. J., Choudhury, K. R., Kondratovich, M. V., Toledano, A., ... & Sullivan, D. C. (2015). The emerging science of quantitative imaging biomarkers terminology and definitions for scientific studies and regulatory submissions. *Statistical Methods in Medical Research, 24*, 9-26. doi:10.1177/0962280214537333

Khan, M. N., Khalid, M., & ul Haq, S. (2013). Review of requirements management issues in software development. *International Journal of Modern Education and Computer Science, 5*(1), 21-27. doi:10.5815/ijmecs.2013.01.03

Kim, M. J., Chung, N., Lee, C. K., & Preis, M. W. (2015). Motivations and use context in mobile tourism shopping: Applying contingency and task–technology fit theories. *International Journal of Tourism Research, 17*, 13-24. doi:10.1002/jtr.1957

Kline, R. B. (2016). *Principles and practice of structural equation modeling* (4th ed.). New York, NY: The Guilford Press.

Krzywinski, M., & Altman, N. (2015). Points of significance: Multiple linear regression. *Nature Methods*, *12*, 1103-1104. doi:10.1038/nmeth.3665

Kulkarni, P. S., & Bakal, J. W. (2014). Survey on data cleaning. *International Journal of Engineering Science and Innovative Technology*, *3*(4), 615-620. Retrieved from http://www.ijesit.com.

Leedy, P., & Ormrod, J. (2015). Practical research: Planning and design (11th ed). Essex, England: Pearson Education Unlimited.

Leveson, N. G. (2013). Learning from the past to face the risks of today. *Communications of the ACM, 56*(6), 38-42. doi:10.1145/2461256.2461270

Li, Y., Lu, Y., Kwak, Y. H., & Dong, S. (2015). Developing a city-level multi-project management information system for Chinese urbanization. *International Journal* *of Project Management*, *33*, 510-527. doi:10.1016/j.ijproman.2014.06.011

Lindsjørn, Y., Sjøberg, D. I., Dingsøyr, T., Bergersen, G. R., & Dybå, T. (2016). Teamwork quality and project success in software development: A survey of agile development teams. *Journal of Systems and Software*, *122*, 274-286. doi:10.1016/j.jss.2016.09.028

Little, R. J., & Rubin, D. B. (2014). *Statistical analysis with missing data* (2nd Ed.). Hoboken, NJ: John Wiley & Sons, Inc..

Liu, L., Kong, X., & Chen, J. (2015). How project duration, upfront costs and uncertainty interact and impact on software development productivity? A simulation approach. *International Journal of Agile Systems and Management*, *8*, 39-52. doi:10.1504/ijasm.2015.068605

Locatelli, G., Mancini, M., & Romano, E. (2014). Systems engineering to improve the governance in complex project environments. *International Journal of Project Management*, *32*, 1395–1410. doi:10.1016/j.ijproman.2013.10.007

Lounsbury, M., & Beckman, C. M. (2015). Celebrating organization theory. *Journal of Management Studies*, *52*, 288-308. doi:10.1111/joms.12091

Maity, M., Dass, M., & Malhotra, N. K. (2014). The antecedents and moderators of offline information search: A meta-analysis. *Journal of Retailing*, *90*, 233-254. doi:10.1016/j.jretai.2014.03.001

Marinho, M. L., Sampaio, S. C., Lima, T. L., & Moura, H. P. (2014). A guide to deal with uncertainties in software project management. *International Journal of Computer Science and Information Technology*, *6*(5), 1–20.

doi:10.5121/ijcsit.2014.6501

Martens, K., & Van Weelden, P. (2014). Decision-making on transport infrastructure and contested information: A critical analysis of three approaches. *European Planning Studies*, *22*(3), 648-666. doi:10.1080/09654313.2013.783665

McAdam, R., Miller, K., & McSorley, C. (2016). Towards a contingency theory perspective of quality management in enabling strategic alignment. *International Journal of Production Economics*. doi:10.1016/j.ijpe.2016.07.003

McLeod, L., Doolin, B., & MacDonell, S. G. (2012). A perspective-based understanding of project success. *Project Management Journal*, *43*(5), 68-86. doi:10.1002/pmj.21290

McPeake, J., Bateson, M., & O'Neill, A. (2014). Electronic surveys: How to maximise success. *Nurse Researcher*, *21*(3), 24-26. doi:10.7748/nr2014.01.21.3.24.e1205

Meeker, W. Q., & Escobar, L. A. (2014). *Statistical methods for reliability data*. Hoboken, NJ: John Wiley & Sons.

Mehta, N., Hall, D., & Byrd, T. (2014). Information technology and knowledge in software development teams: The role of project uncertainty. *Information & Management*, *51*, 417-429. doi:10.1016/j.im.2014.02.007

Melo, A., Tavares, E. A. G., Sousa, E., & Marinho, M. (2015). Dependability approach for evaluating software development risks. *IET Software*, *9*, 17-27. doi:10.1049/iet-sen.2013.0196

Mertler, C. A., & Reinhart, R. V. (2017). *Advanced and multivariate statistical methods: Practical application and interpretation* (6th ed.). New York, NY: Routledge;

Taylor & Francis Group

Mikes, A., & Kaplan, R. S. (2015). When one size doesn't fit all: Evolving directions in the research and practice of enterprise risk management. *Journal of Applied Corporate Finance*, *27*(1), 37-40. doi:10.1111/jacf.12102

Mir, F. A., & Pinnington, A. H. (2014). Exploring the value of project management: Linking project management performance and project success. *International Journal of Project Management*, *32*, 202-217. doi:10.1016/j.ijproman.2013.05.012

Müller, R., & Jugdev, K. (2012). Critical success factors in projects: Pinto, Slevin, and Prescott-The elucidation of project success. *International Journal of Managing Projects in Business*, *5*, 757-775. doi:10.1108/17538371211269040

Nardelli, M., Valenza, G., Cristea, I. A., Gentili, C., Cotet, C., David, D., ... & Scilingo, E. P. (2015). Characterizing psychological dimensions in non-pathological subjects through autonomic nervous system dynamics. *Frontiers in Computational Neuroscience*, *9*(37),1-12. doi:10.3389/fncom.2015.00037

Nimon, K. F., & Oswald, F. L. (2013). Understanding the results of multiple linear regression: Beyond standardized regression coefficients. *Organizational Research Methods*, *16*, 650-674. doi: 10.1177/1094428113493929

Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics (OUSD[AT&L]). (2015, January 7). Operation of the defense acquisition system (DoD Instruction 5000.02). Washington, DC: Author.

O'Leary, Z. (2017). *The essential guide to doing your research project* (3rd ed.). London, UK: SAGE Publications.

Örtenblad, A., Putnam, L. L., & Trehan, K. (2016). Beyond Morgan's eight metaphors: Adding to and developing organization theory. *Human Relations*, *69,* 875-889. doi:10.1177/0018726715623999

Otley, D. (2016). The contingency theory of management accounting and control: 1980–2014. *Management Accounting Research*, *31*, 45-62. doi:10.1016/j.mar.2016.02.002

Palinkas, L. A., Horwitz, S. M., Green, C. A., Wisdom, J. P., Duan, N., & Hoagwood, K. (2015). Purposeful sampling for qualitative data collection and analysis in mixed method implementation research. *Administration and Policy in Mental Health and Mental Health Services Research*, *42,* 533-544. doi:10.1007/s10488-013-0528-y

Park, J., & Park, M. (2016). Qualitative versus quantitative research methods: Discovery or justification?. *Journal of Marketing Thought*, *3*(1), 1-7. doi:10.15577/jmt.2016.03.01.1

Pelikan, M., Stikova, H., & Vrana, I. (2017). Detection of resource overload in conditions of project ambiguity. *IEEE Transactions on Fuzzy Systems, 25,* 868-877. doi:10.1109/tfuzz.2016.2584645

Peña, M., & Valerdi, R. (2015). Characterizing the impact of requirements volatility on systems engineering effort. *Systems Engineering*, *18*, 59-70. doi:10.1111/sys.21288

Peng, D. X., Heim, G. R., & Mallick, D. N. (2014). Collaborative product development: The effect of project complexity on the use of information technology tools and new product development practices. *Production and Operations Management*, *23,* 1421-1438. doi:10.1111/j.1937-5956.2012.01383.x

Petrovčič, A., Petrič, G., & Manfreda, K. L. (2016). The effect of email invitation elements on response rate in a web survey within an online community. *Computers in Human Behavior*, *56,* 320-329. doi:10.1016/j.chb.2015.11.025

Project Management Institute. (2017). *A guide to the project management body of knowledge (PMBOK guide)* (6th ed.). Newton Square, PA: Project Management Institute, Inc.

Ramasubbu, N., Bharadwaj, A., & Kumar Tayi, G. (2015). Software process diversity: Conceptualization, measurement, and analysis of impact on project performance. *MIS Quarterly, 39*, 787-807. doi:10.2139/ssrn.2627173

Restuccia, M., Brentani, U., Legoux, R., & Ouellet, J. F. (2016). Product life-cycle management and distributor contribution to new product development. *Journal of Product Innovation Management*, *33,* 69-89. doi:10.1111/jpim.12261

Ritchie, J., Lewis, J., Nicholls, C. M., & Ormston, R. (Eds.). (2013). *Qualitative research practice: A guide for social science students and researchers*. London, England: SAGE Publications, Inc.

Rizzo, J. R., House, R. J., & Lirtzman, S. I. (1970). Role conflict and ambiguity in complex organizations. *Administrative Science Quarterly, 15,* 150-163. doi:10.2307/2391486

Rowlinson, M., Hassard, J., & Decker, S. (2014). Research strategies for organizational history: A dialogue between historical theory and organization theory. *Academy* *of Management Review*, *39*, 250-274. doi:10.5465/amr.2012.0203

Sabbineni, S., & Rao, K. R. (2015). A comparitive study on reliability and cost estimation for architecture-based software. *International Journal of Applied Engineering Research*, *10*, 19729-19736. Retrieved from https://www.ripublication.com/

Sajjadi, M., Rassouli, M., Abbaszadeh, A., Majd, H. A., & Zendehdel, K. (2014). Psychometric properties of the persian version of the Mishel's uncertainty in illness scale in patients with cancer. *European Journal of Oncology Nursing*, *18*, 52-57. doi:10.1016/j.ejon.2013.09.006

Särndal, C.E., & Lundström, S. (2005). *Estimation in surveys with nonresponse*. doi:10.1002/0470011351

Sauermann, H., & Roach, M. (2013). Increasing web survey response rates in innovation research: An experimental study of static and dynamic contact design features. *Research Policy*, *42*, 273-286. doi:10.1016/j.respol.2012.05.003

Schneider, K. N. (2013). Improving data security in small businesses. *Journal of Technology Research*, *4*, 1-8. Retrieved from http://www.aabri.com/

Schuler, R. S., Aldag, R. J., & Brief, A. P. (1977). Role conflict and ambiguity: A scale analysis. *Organizational Behavior and Human Performance*, *20*, 111-128. doi:10.1016/0030-5073(77)90047-2

Serrador, P., & Pinto, J. K. (2015). Does agile work?—A quantitative analysis of agile project success. *International Journal of Project Management*, *33*, 1040-1051. doi:10.1016/j.ijproman.2015.01.006

Sessa, J., & Syed, D. (2016, December). *Techniques to deal with missing data.* Paper presented at *2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA),* Ras Al Khaimah, UAE. doi:10.1109/ICEDSA.2016.7818486

Siddiqui, K. A., Bajwa, I. A., Elahi, M. A., & Fahim, S. M. (2016). Heuristics of applying statistical tests using appropriate measurement scales. *Science International*, 2*8,* 4433-4438. Retrieved from http://www.sci-int.com

Srinivasan, R., & Swink, M. (2015). Leveraging supply chain integration through planning comprehensiveness: An organizational information processing theory perspective. *Decision Sciences, 46*, 823-861. doi:10.1111/deci.12166

Stantchev, V., Colomo-Palacios, R., Soto-Acosta, P., & Misra, S. (2014). Learning management systems and cloud file hosting services: A study on students' acceptance. *Computers in Human Behavior*, *31*, 612-619. doi:10.1016/j.chb.2013.07.002

Stapel, K., & Schneider, K. (2014). Managing knowledge on communication and information flow in global software projects. *Expert Systems*, *31*, 234-252. doi:10.1111/exsy.649

Suddaby, R. (2015). Can institutional theory be critical?. *Journal of Management Inquiry*, *24*, 93-95. doi:10.1177/1056492614545304

SurveyMonkey. (n.d). SurveyMonkey special pricing for students and educators. Retrieved from https://www.surveymonkey.com/pricing/?ut_source=edu_faq&e=edu_faq

Tarhan, A., & Yilmaz, S. G. (2014). Systematic analyses and comparison of development performance and product quality of incremental process and agile process. *Information and Software Technology, 56*, 477-494. doi:10.1016/j.infsof.2013.12.002

Teddlie, C., & Tashakkori, A. (2003). *Major issues and controversies in the use of mixed methods in the social and behavioral sciences. Handbook of Mixed Methods in Social & Behavioral Research* (pp. 3-50). Thousand Oaks, CA: SAGE Publications, Inc.

Thompson, B., Diamond, K. E., McWilliam, R., Snyder, P., & Snyder, S. W. (2005). Evaluating the quality of evidence from correlational research for evidence-based practice. *Exceptional Children*, *71*, 181-194. doi:10.1177/001440290507100204

Too, E. G., & Weaver, P. (2014). The management of project management: A conceptual framework for project governance. *International Journal of Project Management*, *32*, 1382-1394. doi:10.1016/j.ijproman.2013.07.006

Uprichard, E. (2013). Sampling: Bridging probability and nonprobability designs. *International Journal of Social Research Methodology*, *16*, 1-11. doi:10.1080/13645579.2011.633391

U.S. National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research. (1978). *The Belmont report: Ethical principles and guidelines for the protection of human subjects of research* (SuDoc No. Y3.H88:2B41). Washington, DC: Government Printing Office.

Van den Hoek, R. E., Brugnach, M., Mulder, J. M., & Hoekstra, A. Y. (2014).

Uncovering the origin of ambiguity in nature-inclusive flood infrastructure projects. *Ecology and Society*, *19*(2), 520-538. doi:10.5751/ES-06416-190251

Verner, J. M., Brereton, O. P., Kitchenham, B. A., Turner, M., & Niazi, M. (2014). Risks and risk mitigation in global software development: A tertiary study. *Information and Software Technology*, *56*, 54-78. doi:10.1016/j.infsof.2013.06.005

Vittikh, V. A. (2016). Evergetics: Science of intersubjective management processes in everyday life. *International Journal of Management Concepts and Philosophy*, *9*, 63-72. doi:10.1504/ijmcp.2016.077763

Walden University. (n.d.). Walden University Institutional Review Board for Ethical Standards in Research http://academicguides.waldenu.edu/researchcenter/orec

Wang, K. W. K., Chen, I. H., & Chen, M. (2016). Uncertainty of parents with children undergoing herniorrhaphy. *Clinical Nursing Research*. doi:10.1177/1054773816676942

Whalen, P., Uslay, C., Pascal, V. J., Omura, G., McAuley, A., Kasouf, C. J., … Deacon, J. (2016). Anatomy of competitive advantage: Towards a contingency theory of entrepreneurial marketing. *Journal of Strategic Marketing*, 24, 5–19. doi:10.1080/0965254x.2015.1035036

Wickboldt, J. A., De Jesus, W. P., Isolani, P. H., Both, C. B., Rochol, J., & Granville, L. Z. (2015). Software-defined networking: Management requirements and challenges. *IEEE Communications Magazine*, *53*(1), 278-285. doi:10.1109/mcom.2015.7010546

Williams, C. (2007). Research methods. *Journal of Business & Economics Research*

(JBER), *5*(3), 65-72. doi:10.19030/jber.v5i3.2532.

Williams, M. M., Gómez Grajales, C. A., & Kurkiewicz, D. (2013). Assumptions of multiple regression: Correcting two misconceptions. *Practical Assessment, Research & Evaluation*, *18*(11), 1-14. Retrieved from http:// http://pareonline.net

Wong, C. W., Lai, K. H., Cheng, T. C. E., & Lun, Y. V. (2015). The role of IT-enabled collaborative decision making in inter-organizational information integration to improve customer service performance. *International Journal of Production Economics*, *159*, 56-65. doi:10.1016/j.ijpe.2014.02.019

Wright, K. B. (2005). Researching internet-based populations: Advantages and disadvantages of online survey research, online questionnaire authoring software packages, and web survey services. *Journal of Computer-Mediated Communication*, *10*(3). doi:10.1111/ j.1083-6101.2005.tb00259.x

Yang, M., & Yuan, K. (2016). Robust methods for moderation analysis with a two-level regression model. *Multivariate Behavioral Research*, *51*, 757-771. doi:10.1080/00273171.2016.1235965

Yang, Q., Lu, T., Yao, T., & Zhang, B. (2014). The impact of uncertainty and ambiguity related to iteration and overlapping on schedule of product development projects. *International Journal of Project Management*, *32*, 827-837. doi:10.1016/j.ijproman.2013.10.010

Yilmaz, K. (2013). Comparison of quantitative and qualitative research traditions: Epistemological, theoretical, and methodological differences. *European Journal of Education*, *48*, 311-325. doi:10.1111/ejed.12014

Yoo, W., Mayberry, R., Bae, S., Singh, K., He, Q. P., & Lillard Jr, J. W. (2014). A study of effects of multicollinearity in the multivariable analysis. *International Journal of Applied Science and Technology, 4*, 9-19. Retrieved from http://www.ijastnet.com/

Zand, F., Solaimani, S., & van Beers, C. (2015). A role-based typology of information technology: Model development and assessment. *Information Systems Management*, *32*, 119-135. doi:10.1080/10580530.2015.1018770

Zowghi, D., & Nurmuliani, N. (2002). A study of the impact of requirements volatility on software project performance. *Proceedings of the Ninth Asia-Pacific Software Engineering Conference, Gold Coast, Australia*, 3-11. doi:10.1109/apsec.2002.1182970

Appendix A: Survey Instrument

Part I

On a scale between 1 “Strongly Disagree” and 7 “Strongly Agree”, please mark only one item that best represents your level of agreement with statements 1 through 6, with respect to software development team’s structure and organization for the last project you managed and completed.

Statement	Strongly Disagree	Disagree	Disagree Somewhat	Undecided	Agree Somewhat	Agree	Strongly Agree
1. Going by the results of the last project, this teamwork can be regarded as successful	1	2	3	4	5	6	7
2. From the company’s perspective, all team goals are achieved	1	2	3	4	5	6	7
3. The performance of the team advances the company’s image to the customer	1	2	3	4	5	6	7
4. The teamwork result is of high quality	1	2	3	4	5	6	7
5. The product produced in the team, requires little rework	1	2	3	4	5	6	7
6. The customer is satisfied with the quality of the teamwork result	1	2	3	4	5	6	7

Part II

On a scale between 1 “Strongly Disagree” and 7 “Strongly Agree”, please mark only one item that best represents your level of agreement with statements 7 through 18, with respect to software specifications for the last project you managed and completed.

Statement	Strongly Disagree	Disagree	Disagree Somewhat	Undecided	Agree Somewhat	Agree	Strongly Agree
7. The team is certain about how much authority it has.	1	2	3	4	5	6	7
8. Requirements fluctuated in the earlier stages of the last software project completed	1	2	3	4	5	6	7
9. The team received clear, planned requirements, goals, and objectives for the last completed project.	1	2	3	4	5	6	7
10. Requirements fluctuated in the later stages last software project completed	1	2	3	4	5	6	7
11. The team members know what is expected of each other.	1	2	3	4	5	6	7

12. Difference in requirements identified at the start of the last software project completed from the final requirements	1	2	3	4	5	6	7
13. The team is uncertain as to how the last completed project is link with other projects.	1	2	3	4	5	6	7
14. It was difficult for stakeholders to reach agreement among themselves on requirements.	1	2	3	4	5	6	7
15. The team is told how well the team is doing its job.	1	2	3	4	5	6	7
16. A lot of effort had to be spent in incorporating the requirements of various users	1	2	3	4	5	6	7
17. The team knew the end product would be acceptable according to the requirements.	1	2	3	4	5	6	7
18. It was difficult to customize software to one set of users without reducing support for other users	1	2	3	4	5	6	7

<u>Part III</u>

On a scale between 1 “Strongly Disagree” and 7 “Strongly Agree”, please mark only one item that best represents your level of agreement with statements 19 through 24, with respect to outcomes of the last project you managed and completed.

Statement	Strongly Disagree	Disagree	Disagree Somewhat	Undecided	Agree Somewhat	Agree	Strongly Agree
19. The project failed in achieving its cost goals, as initially planned.	1	2	3	4	5	6	7
20. The project produced its results in a timely fashion.	1	2	3	4	5	6	7
21. The project successfully achieved its scope and quality goals.	1	2	3	4	5	6	7
22. The client/customer constantly complained about the results of this project.	1	2	3	4	5	6	7
23. This project was a success.	1	2	3	4	5	6	7
24. The project failed in producing the requirements expected by the customer.	1	2	3	4	5	6	7

Appendix B: Items for Measures

Variable	Survey Statement
Software Development Team Structure	1. Going by the results of the last project, this teamwork can be regarded as successful
	2. From the company's perspective, all team goals are achieved
	3. The performance of the team advances the company's image to the customer
	4. The teamwork result is of high quality
	5. The product produced in the team, requires little rework
	6. The customer is satisfied with the quality of the teamwork result
Ambiguity in Software Specifications	7. The team is certain about how much authority it has.
	9. The team received clear, planned requirements, goals, and objectives for the last completed project.
	11. The team members know what is expected of each other.
	*13. The team is uncertain as to how the last completed project is link with other projects.
	15. The team is told how well the team is doing its job.
Volatility in Software Specifications	*8. Requirements fluctuated in the earlier stages of the last software project completed
	*10. Requirements fluctuated in the later stages last software project completed
	*12. Difference in requirements identified at the start of the last software project completed from the final requirements
	*14. It was difficult for stakeholders to reach agreement among themselves on requirements.
	*16. A lot of effort had to be spent in incorporating the requirements of various users
	*18. It was difficult to customize software to one set of users without reducing support for other users

Software Project Failure	*19. The project failed in achieving its cost goals, as initially planned.
	20. The project produced its results in a timely fashion.
	21. The project successfully achieved its scope and quality goals.
	*22. The client/customer constantly complained about the results of this project.
	23. This project was a success.
	*24. The project failed in producing the requirements expected by the customer.

*Indicates reverse coded item