2017

Improving Bespoke Software Quality: Strategies for Application and Enterprise Architects

Daniel Scott Wagner
Walden University

Walden University

College of Management and Technology

This is to certify that the doctoral study by

Daniel Wagner

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. Steven Case, Committee Chairperson, Information Technology Faculty
Dr. Gail Miles, Committee Member, Information Technology Faculty
Dr. Jon McKeeby, University Reviewer, Information Technology Faculty

Chief Academic Officer
Eric Riedel, Ph.D.

Walden University
2017

Abstract

Improving Bespoke Software Quality: Strategies for Application and Enterprise Architects

by

Dan Wagner

MSIT, Capella University, 2009

BBA, American Intercontinental University, 2006

Doctoral Study Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Information Technology

Walden University

May 2017

Abstract

Despite over 50 years of software engineering as a formal practice, contemporary developers of bespoke software follow development practices that result in low-quality products with high development and maintenance costs. This qualitative case study sought to identify strategies used by software and enterprise architects for applying architectural best practices to improve bespoke software quality and lower the total cost of ownership. The study population was application and enterprise architects associated with delivering bespoke software for the enterprise architecture team at a large enterprise in the Nashville, Tennessee metropolitan area. Interview data were collected from 7 enterprise or solution architects; in addition, 47 organizational documents were gathered. Guided by the principles of total quality management, thematic analysis was used to identify codes and themes related to management of quality in software solutions. Prominent themes included focusing on customer satisfaction, collaborating and communicating with all stakeholders, and defining boundaries and empowering people within those boundaries. The findings from this research have implications for positive social change, including improved work-life balance, morale, and productivity of software and enterprise architects through streamlining development and maintenance activities.

Improving Bespoke Software Quality: Strategies for Application and Enterprise Architects

by

Dan Wagner

MSIT, Capella University, 2009

BBA, American Intercontinental University, 2006

Doctoral Study Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctoral of Information Technology

Walden University

May 2017

Dedication

I dedicate this to my two children, Austin and Hayden. Technology is changing our world at a rapid pace. Remember that technology is supposed to make our lives easier.

Acknowledgements

I acknowledge and thank all of the following people for their involvement in this research:

Scott and Licette Duarte for getting me into IT to begin with. Without both of you, I would not be who I am today. Thank you for the introduction to and start in an I.T. capacity all of those years ago.

My employer and immediate supervisors for affording me time to complete my research. Not only have you given me time to complete my research and report, you have also provided me with motivation to complete it. I cannot express how much I appreciate your support.

All of those who participated in this research, for your dedication to my project in the midst of a high-pressure work environment. Your involvement was truly voluntary and very much appreciated. I asked for your trust and you gave it to me. I hope I used it well.

My wife and children for putting up with my odd hours, missed family nights, baseball and soccer practices, and general time together. I love you all. Thank you for your support, understanding, and motivation.

My committee chair, Dr. Steven Case, and all of my committee members, Dr. Gail Miles, and Dr. Jon McKeeby. This undertaking has challenged me academically, emotionally, and physically. Thank you, all, for your support, encouragement, and understanding. In addition, thank you to Walden's chief academic officer, Dr. Karlyn Barilovits and all of the other staff members and classmates at Walden University that I have worked and studied with these past 2.75 years.

Table of Contents

List of Tables

List of Figures

Section 1: Foundation of the Study

Background of the Problem

The first computing devices and related software started improving operations in the 1930s (Booch, 2015). Since that time, the methods, approaches, and tools used to design, develop, and deliver ISs changed drastically with goals of improving the effectiveness and efficiency of the individuals that create, maintain, and operate the ISs (Booch, 2015). Despite their efforts, improvements in the process and tools software engineers used to develop software did not result in repeatable, generalized higher quality (Ahmad, 2016; Atkinson & Benefield, 2013; Jones, 2015).

Development of contemporary ISs is a complex task that requires the involvement of a broad range of stakeholders. Customers have needs and perceptions of quality that define their efficiency and effectiveness (Liu, Chang, & Tsai, 2015). Analysts work with stakeholders to document their needs and requirements and software engineers work to implement the documented requirements as ISs (Buenen & Walgude, 2015). Architects work with all of the stakeholders to ensure a holistic view and balanced needs across all stakeholders (Ahmad, 2016). These roles must work in harmony with a process to achieve IS quality.

The perspective of quality in the software industry is shifting to that of customer satisfaction, which is a highly complex concept (Buenen & Walgude, 2015; Göransson, Gulliksen, & Boivie, 2003). The approaches to achieving satisfaction have evolved at nearly the same rate as the means of developing ISs, yet have not resulted in repeatable satisfaction (Ramasubbu, Bharadwaj, & Tayi, 2015; Singh & Jatain, 2013; Verner, Babar, Cerpa, Hall, & Beecham, 2014). Two of the critical roles in developing ISs are software engineers and solution architects, and—together—they should be able to maximize IS quality (Ahmad, 2016).

Problem Statement

Despite over 50 years of software engineering as a formal practice, contemporary developers of bespoke software follow development practices that result in low-quality products with high development and maintenance costs (Pass & Ronen, 2014). Global IT budget allocation to the quality assurance function rose from 25% to 33% between 2015 and 2016 (Buenen & Walgude, 2015) and software maintenance costs represent 90% of the total cost of bespoke software in 2012 (Dehaghani & Hajrahimi, 2013). The general IT problem is maximizing customer satisfaction with bespoke software through quality improvements. The specific IT problem is that software and enterprise architects often lack strategies for applying architectural best practices to improve bespoke software quality.

Purpose Statement

The purpose of this qualitative exploratory case study was to explore strategies used by software and enterprise architects for applying architectural best practices to improve bespoke software quality, lowering the total cost of ownership. The population for this study comprised application and enterprise architects associated with the delivery of bespoke software for the enterprise architecture team at a large enterprise in the Nashville, Tennessee metropolitan area in the United States. A wide variety of organizations can use the findings from this study to help realize and understand the benefits of having strategies architects can apply to improve the quality of bespoke software solutions. The potential social impact of this study is improved work-life balance, morale, and productivity of software and enterprise architects through streamlining development and maintenance activities.

Nature of the Study

The two primary methodologies used in scholarly research are quantitative and qualitative. Researchers use quantitative methods to identify and explain relationships between aspects of a topic using variables (Trochim & Donnelly, 2008). When a researcher can define the variables that describe the attributes of a topic and they want to determine correlation, causation, or trends between those variables, they use quantitative designs to provide the structure and processes for collecting, analyzing, and evaluating data (Trochim & Donnelly, 2008). Alternately, researchers use qualitative methods to explore and create detailed understandings of a topic or phenomenon (Trochim & Donnelly, 2008; Yin, 2014). When a researcher either is not aware of or cannot identify variables to evaluate or they simply seek to understand a concept fully, qualitative designs provide frameworks for analysis (Yin, 2014). The quantitative methodology was not appropriate for my research as I sought to create a detailed understanding of strategies architects use for improving software quality as opposed to showing a correlation between application and enterprise architect strategies and bespoke software quality.

Within the qualitative methodology, there are a few available designs: ethnography, grounded theory, phenomenology, case study, and narrative (Trochim & Donnelly, 2008; Yin, 2014). Ethnography is used to research distinct cultures or cultural groups (Trochim & Donnelly, 2008). As such, ethnography was not appropriate for my research as I was not concerned with a distinct culture or cultural group. Application and enterprise architect and engineering communities comprise members from a wide variety of cultural backgrounds. Phenomenology is used to study commonalities of participants that experienced an unusual phenomenon (Gentles, Charles, Ploeg, & McKibbon, 2015). I believed that the phenomenon of differences in the focus of architects and engineers commonly occurred, making phenomenology inappropriate for my research. Case study research is used to study complex phenomena when the researcher seeks to understand and describe the phenomena in detail (Hyett, Kenny, & Dickson-Swift, 2014; Yin, 1981, 2014). The case study design was the most appropriate qualitative design for my research as I sought to understand and describe strategies architects used to apply architectural best practices to improve bespoke software quality.

Qualitative Research Question

What strategies do application and enterprise architects use to apply architectural best practices to improve bespoke software quality?

Interview Questions

- How would you describe the distinction between an application architect and a software engineer? Please explain
- What is your understanding of architectural best practices? Please explain.
- What does quality mean to you? Please explain.
- What is your perception of the relationship between architects and software engineers regarding the pursuit of product quality? Please explain.
- What, if any, challenges do you face regarding the application of architectural best practices in delivered software products? Please elaborate.
- What strategies do you have for ensuring the highest possible quality of software products? Please elaborate.

Conceptual Framework

I used total quality management (TQM) as the lens through which I conducted my research and evaluated data. W. Edwards Deming and Joseph M. Juran started the TQM movement in the United States in the mid-1980s following a downturn in the U.S. economy during the 1970s and 1980s (Hill, 2008; Joyce, 2015). Their objective with the introduction of TQM in the United States was improving the quality of products to levels achieved by Japanese organizations (Hill, 2008; Joyce, 2015). The focal point of TQM is achieving high levels of customer satisfaction through continuous improvement and involvement of everyone in the organization (Deming, 1985; Suryanarayana, Sharma, & Samarthyam, 2015). Some contemporary organizations chose to implement TQM in the form of frameworks such as ISO-9000, Six Sigma, and CMMI (Barata & Cunha, 2015; A. Brown, 2014; Tunkelo, Hameri, & Pigneur, 2013). I summarized TQM concepts as a collection of policies, procedures, and components that directly influence customer satisfaction (see Figure 1).

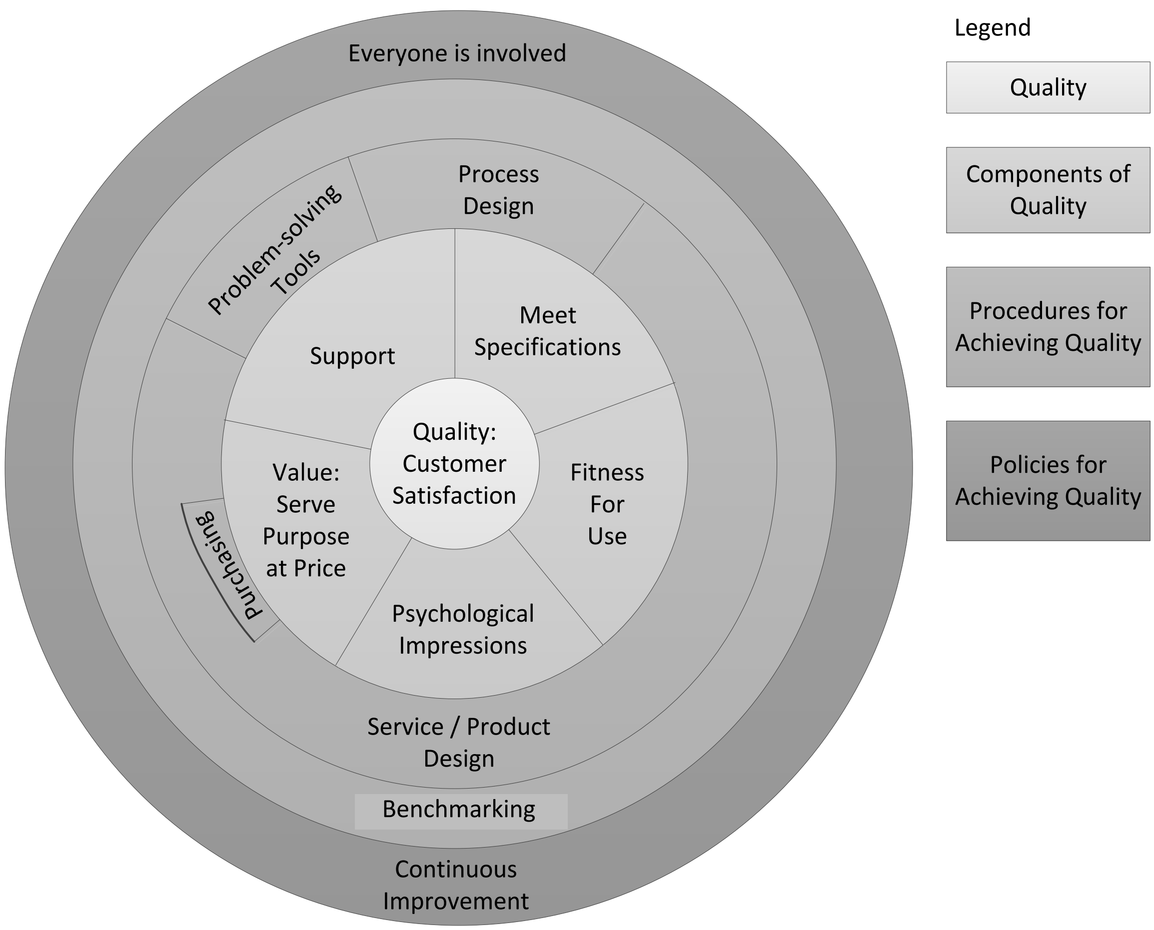

Figure 1. TQM policies, procedures, and components.

The roles of application and enterprise architect focus on design-centric activity, leadership, stakeholder focus, system-wide concerns, full lifecycle involvement, and balancing stakeholder concerns (Woods, 2014). TQM, as a conceptual framework, is significant to this research as the characteristics of application and enterprise architects overlap well with the components, procedures, and policies of TQM. Architect leadership aligns well with continuous improvement, process design, and involving everyone while design-centric activity aligns with service/product design, process design, and problem-solving tools. Stakeholder focus is central to TQM in the form of customer satisfaction, which architects achieve in part through cost controls (purpose at a price). Balancing concerns aligns with all of the components of quality defined by TQM. I summarized the overlap of TQM and architect roles and responsibilities in Table 1.

Table 1

TQM policies and Procedures Mapped to Architect Characteristics.

		Characteristics of an architect					
		Design-centric Activity	Leadership	Stakeholder Focus	System-wide concerns	Life-cycle involvement	Balancing Concerns
Policies	Customer Satisfaction			X		X	X
	Continuous Improvement		X			X	X
	Involving Everyone		X	X		X	X
Procedures	Product Design	X			X	X	X
	Process Design	X	X		X	X	X
	Problem-solving Tools	X			X	X	X
	Purchasing			X	X	X	X
	Benchmarking		X			X	X

Definition of Terms

Application architect: Application architects are architects with a technical role in delivering IT solutions that focus on design-centric activities, leadership, stakeholder focus, system-wide concerns, lifecycle involvement, and balancing concerns (Woods, 2014).

Architect: Architects are people with a role in delivering IT solutions that focus on design-centric activities, leadership, stakeholder focus, system-wide concerns, lifecycle involvement, and balancing concerns (Woods, 2014).

Bespoke software: Bespoke software is software information systems developed by an organization for its internal use (Göransson et al., 2003; Spinellis, 2014b).

Enterprise architect: Enterprise architects are architects with a business domain specialization as opposed to a technical specialization, focusing on aligning technology with business need (Woods, 2014).

Software engineer: Software engineers are people who perform the socio-technical role of developing software using complex algorithms based on experience with what works well (P. L. Li, Ko, & Zhu, 2015).

Solution architect: Solution architects are architects with a balance of business domain and technology knowledge, focusing on quality attributes (e.g. performance, security, modifiability, maintainability, availability, flexibility, reliability, and reusability) of solutions (Woods, 2014).

Assumptions, Limitations, and Delimitations

Any number of internal or external phenomena influence research and outcomes. Recognizing and documenting these phenomena is part of establishing credibility. Three categories of phenomena that occur in research are assumptions, limitations, and delimitations.

Assumptions

Assumptions are accepted truths that are unproven by measurable test (Joseph, 2013). There were a few assumptions included in this study. First, the organization participating in this case study was representative of the overall industry. Second, the number of interviews and other sources of evidence used in my research adequately represented the case regarding quality and strategies for achieving quality. Last, the interviewees adequately represented the case selected for inclusion in this study.

Limitations

Limitations are aspects of a study that are—despite the researcher's best effort—beyond their control (Teichler, 2014). The primary limitations of this study derived from the qualitative nature of the research. Interpreting dialogue and artifacts for themes is a subjective process that results in potential issues for validity and reliability due to potential bias (Yin, 2014). Additionally, due to the limited number of cases evaluated in a case study design, the generalizability of the findings is theoretical at best (Yin, 2014).

Delimitations

Delimitations are boundaries that a researcher imposes on a study to control or narrow the scope of the research (Felt, Igelsböck, Schikowitz, & Völker, 2013). There were a few primary delimitations in my research. First, I considered only organizations that develop software for internal use. Second, I considered only organizations that employed people in a role of architect that meets the industry-standard definition of an architect. Third, I included only organizations that met the first two criteria and were in the metropolitan Nashville, TN area. Fourth, I did not consider the effects of the globalization of software development (including potential multi-cultural impacts and the influence of offshored outsourcing). Fifth, architects employed at the case organization must have been full-time employees of the organization at the time of the study and must have had at least 5 years of experience as an architect at, either at the case organization or any organization over their career.

Significance of the Study

While reviewing the academic literature for research regarding strategies used by architects to influence bespoke software quality, I found no matching studies. Given the overlap of aspects of quality defined in TQM with the responsibilities assigned to an architect, there is an opportunity to enhance both academic knowledge and practice in this area. I expected that this study would lead to further research on the topic.

Contribution to Information Technology Practice

Information technology (IT) departments, like most departments of contemporary organizations, were charged with reducing costs and increasing the quality of outputs; colloquially doing more with less. Efficiency and effectiveness of operations were central to their operations. This research was significant to IT practice as it provided detailed descriptions of strategies used by architects to successfully align bespoke software solutions with customer satisfaction to improve agility and responsiveness, reduce overall costs of developing custom ISs, and align IT assets with business strategies.

Implications for Social Change

The implications for positive social change included improving the work/life balance of both IT and business workers alike. The literature I reviewed as part of this research suggested that improving the quality of software developed for internal use improves the efficiency and effectiveness of both those that develop the solutions and those that use the solutions. Increased efficiency means that developers and users accomplish their work tasks in less time, affording them more time for other tasks, either work or personal. This increased efficiency also decreases the stress levels of both IT managers and end-users of bespoke software solutions. Lowering stress levels at work helps to improve employee morale and productivity.

A Review of the Professional and Academic Literature

Three primary components of my research covered in this literature review were as follows:

1. The problems organizations faced with bespoke software development and the roles individuals filled in the development of bespoke software,
2. the use of TQM as a conceptual framework in research and as a means of achieving quality in bespoke software, and
3. the use of case study research design in bespoke software development and TQM implementations.

I searched several sources of academic and professional articles to ensure complete coverage of the topics. I used Google Scholar, ProQuest Central, Science Direct, EBSCOHost Academic Search Complete, IEEE Explore, IEEE Computer Society Digital Library, and ACM Digital Library as the primary search locations. I also used the reference lists of articles I found in these sources as alternate sources.

While searching, I reviewed and considered seminal and other appropriate articles from all timeframes, focusing on recent (2013 and newer) articles to ensure a contemporary perspective of the topics. The approach I took to searching evolved over time, beginning with distinct primary search criteria for each of the themes in my literature review. For TQM, search terms evolved to include the following: *Total quality management, quality management system,* and *quality management.*

For bespoke software development, search terms evolved to include the following: *bespoke software issues, bespoke software problems, bespoke software quality*, and *tailor-made software*. The literature retrieved for both TQM and bespoke software development provided a foundation for the use of case study as a research design.

I reviewed over 250 articles and discarded more than 150 as either irrelevant to the scope or duplicative. In total, I included 103 articles in this review with 88 published within the last 5 years (2013 and newer; 85.44%). There are 86 articles from peer-reviewed sources (83.50%) and four doctoral dissertations or theses. According to Walden's policy, up to 10% of peer reviewed sources could be dissertations, resulting in four of the dissertations counting toward peer-reviewed totals. The adjusted total count of peer-reviewed articles is 90, or 87.38%.

While the development of case studies and the use of a case survey design have been around for a while longer, I identified Robert Yin's (1981) article supporting case study as a valid research methodology as the seminal article on single case studies (now referred to as a design in the qualitative methodology). Similarly, while there was previous work on case study design by Robert Stake, I identified the seminal work on multiple case studies as a report on case studies in science education (Stake, 1978). I chose this as the seminal work on multiple case studies as Stake (1978) noted that part of the intent of the report was to show the efficacy and value of multiple case studies.

Similarly, while the topics that drove TQM started with Frederick Winslow Taylor in 1911 (Khalil, Stockton, Alkaabi, & Manyonge, 2015), the formal definition of TQM did not occur until much later. In 1985, the U.S. Department of Defense (DoD) was formalizing their quality initiatives with the guidance of Deming and Juran and associated the TQM moniker with the initiative. I identified Houston and Dockstader's (1997) report on the DoD's implementation in 1985 as the seminal work on TQM, though authors in the reviewed literature frequently cite Deming and Juran as the leading experts on TQM (Hill, 2008).

I began this literature review with an analysis of literature regarding bespoke software development. I presented how the literature defined bespoke software and the two prominent themes found in the literature: the software development process and requirements engineering. Usability was a requirement that had a significant amount of coverage in the literature as well. As such, I included one section dedicated to usability as it relates to the perception of quality. I concluded the section on bespoke software with a review of the literature on the failures of bespoke software, methods to improve bespoke software, and a review of research designs used in research of bespoke software in the literature.

Following the analysis of bespoke software development, I presented the evolution of TQM followed by a review of contemporary implementations, supporting conceptual models, and contrasting conceptual models. Because TQM was a practice implemented in the industry to control product quality, its use in this research was more than just that of a conceptual framework. In order to provide a foundation for the basis of this research (the distinction between software engineers and solution architects in terms of quality focus and the strategies used by architects to influence quality), it is important to understand the key concepts of TQM and the aspects of products or services that comprise or affect satisfaction. For that reason, I continued the literature review with analysis of how researchers define TQM, customer satisfaction, the policies of TQM, and the procedures of TQM. I concluded this section with a summary of the research designs used in research of TQM to substantiate my use of case study. I concluded the literature review with a section on differences between two prominent roles involved in bespoke software development and a critical evaluation of themes found in the literature.

Bespoke Software Development

When considering software for use in business operations, organizations seek bespoke solutions because commercial software is not always a good fit. Göransson, Gulliksen, and Boivie (2003) defined bespoke software as software that is developed in-house for work-related use. Spinellis (2014a, 2014b) mentioned that organizations choose to implement bespoke solutions as they are tailored to fit unique organizational needs and support the organization's ability to enhance them on their timeline; aspects of aligning software to business needs that are not always possible with commercial software. Atkinson and Benefield (2013) added that businesses typically treat software development as a necessary evil; as a cost center instead of a value creation center. While there may be justification, the choice to develop bespoke solutions comes with its challenges.

The Process of Bespoke Software Development

One of the considerations when deciding to develop bespoke solutions is the process the organization will use. Göransson et al. (2003) noted that the three primary phases in the course of developing and delivering bespoke software are requirements analysis, iteratively developing the software, and deploying the software (see Figure 2). Alternately, Galster, Weyns, Tofan, Michalik, and Avgeriou (2014) noted that effective development processes include requirements gathering, requirements engineering, architecture, solution design, evaluation of solution design, development, code reviews, system integration, and testing the complete system. No matter which steps the organization includes, the collection of steps in a software development process is known as the software development life cycle (SDLC) is. The process used to develop products (e.g. the SDLC) is believed to have a profound impact on product quality (Deming, 1982; Ishikawa, 1984; Juran & Godfrey, 1999).

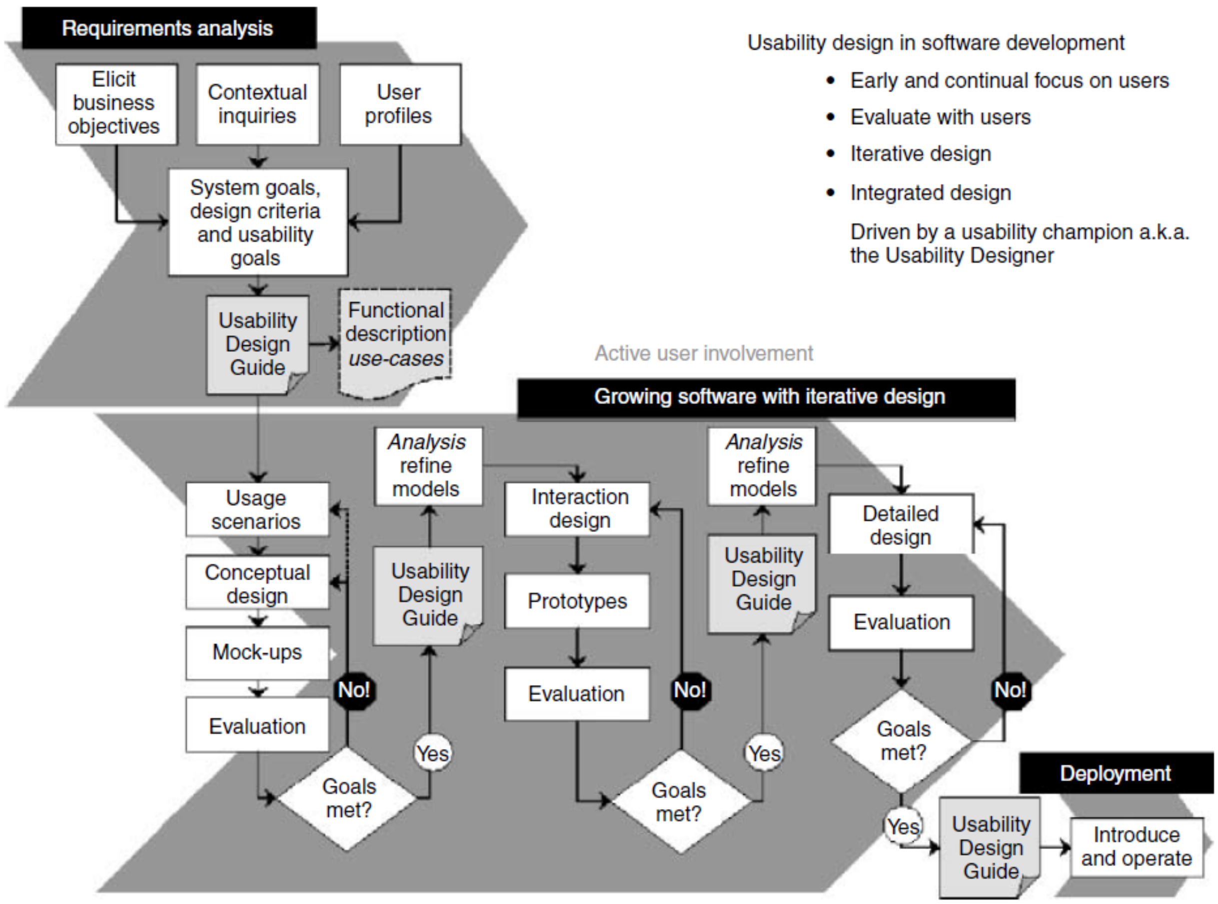

Figure 2. A process for including usability design in software development (Göransson et al., 2003).

There are two contemporary SDLCs in use, each with their issues. Misra, Fernández, and Colomo-Palacios (2014) and Mitchell (2012) noted that common processes for software development include Waterfall and Agile. Mitchell (2012) stated that Waterfall became a formalized methodology in the 1970s, driven by W. Royce, and Shah (2014) criticized it as being overly process-centric. Atkinson and Benefield (2013) added that it results in inefficiency and waste through loss of knowledge at each handoff. Booch (2015) stated that waterfall was appropriate for its time due to the nature of software at the time, though changes in the programming industry that arose in the 1980s forced a change.

Agile SDLCs address some of the issues associated with Waterfall. Despa (2015) indicated that Agile methodologies focus on adaptability and communication. Göransson et al. (2003) and Huckabee (2015) reported that Agile lifecycles are geared toward delivering working software over comprehensive documentation, yet Despa (2015) and Johann (2015) believed that Agile methodologies do not scale well in larger organizations and do not support adequate architectural planning. However, Poort (2014) believed that updated models such as Scaled Agile Framework now include architecture in the process. Ramasubbu, Bharadwaj, and Tayi (2015) believed that the volatility of requirements should be considered when selecting a development process, adding that Agile processes tend to work better on projects with higher requirement volatility while plan-based processes (e.g. waterfall) tend to work better when requirement volatility is lower or non-existent. Continuous improvement of processes also positively affects product quality (Deming, 1982; Feigenbaum, 1985).

The SDLC may need to vary throughout the project to maximize the quality of the outputs. Ramasubbu et al. (2015) noted that, while the process used to produce bespoke software is critical to quality, a single process may not be sufficient and the SDLC may need to vary by development phase, team maturity, and customer involvement. However, Chen and Huang (2009) stated that complications and unpredictability in the software development process negatively affect the maintainability of software. Having a stable SDLC may be critical to quality, yet Verner et al. (2014) stated that software development managers and software developers do not always agree on which aspects of the software development process are essential. These themes suggest that team members should work together to implement the best process.

However, having high-quality products requires more than an effective process. Shah (2014) noted that, across all of the phases of software engineering, people are just as significant to quality and success as technical and process considerations. Involving everyone is believed to have a positive effect on quality (Feigenbaum, 1985; Ishikawa, 1984).

There are a few architectural quality drivers that should be in every development process. Cleland-Huang, Hanmer, Supakkul, and Mirakhorli (2013), Lyu and Liang (2014) and Shanmugasundaram and Vikram (2015) stated that, regardless of software development methodology, solution architecture drives quality by constraining development. These constraints may be critical to product quality as Canessane and Srinivasan (2013) indicated that architectural and design decisions made early on in the development cycle based on functional and nonfunctional requirements have a significant impact on the quality attributes of the resulting system. For example, Bosch, Capilla, and Hilliard (2015) and Galster et al. (2014) noted that variability—or an architectural consideration that affects the ability of a software solution to be adapted for application in different contexts via enhancements or configuration–has an impact on all of the phases in a development process. These researchers suggested what TQM states, that building quality in from the beginning improves product quality (Feigenbaum, 1985; Hill, 2008; Juran & Godfrey, 1999).

Requirements Engineering in the Process of Bespoke Software Development

There are differing schools of thought on the best time for requirement gathering. Atkinson and Benefield (2013) noted that a customer's perception of the ideal solution is at its weakest at the beginning of a project, making defining requirements before the project begins a precipice, though Huckabee (2015) argued that knowing what to build before development starts reduces rework later. Similarly, Göransson et al. (2003) indicated that requirements are developed continuously over the life of the design process, not determined all at once at the beginning of the project. In either case, Atkinson and Benefield (2013) mentioned that managing requirements throughout the development phase is critical as the user's perception of what is possible evolves during the development process. Chen and Huang (2009) added that selecting appropriate requirement engineering approaches and timelines are critical as low-quality requirements—defined by their completeness, accuracy, and clarity—are the source of most reported software issues. Having clear requirements is crucial as building quality in from the beginning is related to improved product quality (Feigenbaum, 1985; Hill, 2008; Juran & Godfrey, 1999).

Addressing business needs through customer involvement is another driver of quality. Singh and Jatain (2013) believed that requirements engineering and prioritization are critical to software value and the elicitation and prioritization process should include users. Göransson et al. (2003) added that different organizations have varying levels of process and business objective maturity and business objectives must be clear before proceeding with requirements engineering. Gathering poor quality requirements, which could be due to misunderstood or unclear business objectives, can have a detrimental effect on quality (Ahmad, 2016) as the design of both the process and the product influences product quality (Hill, 2008).

Nonfunctional requirements (NFRs) might be just as significant to quality as functional requirements. Thakurta (2013) stated that the requirements phase must include quality-related requirements as these NFRs define the quality attributes of bespoke software. Ameller, Ayala, Cabot, and Franch (2013) and Cleland-Huang (2014) indicated that the primary attributes of bespoke software solutions that are affected by NFRs and solution architecture are usability, reliability, performance, efficiency, and maintainability. Canessane and Srinivasan (2013) alternately stated that NFRs affect performance, interface, operational, resource, verification, acceptance, documentation, security, portability, quality, reliability, maintainability, and safety requirements. Göransson et al. (2003) added that usability must be included as a quality criterion when defining requirements and acceptance criteria for bespoke software. NFRs are a means of involving the customer in product quality discussions (Ameller, Ayala, et al., 2013; Cleland-Huang, 2014).

With the variety of NFRs to consider, it is not always clear who is driving their definition. Ameller, Ayala, et al. (2013) indicated that architects are the primary sources of NFRs and they tend to specify NFRs iteratively over the life of the project. Cleland-Huang (2014) stated that solution architects use NFRs to make decisions that influence bespoke software quality, including choosing from myriad architectural patterns and styles that affect extensibility, maintainability, availability, and security. Solution architects focus on maximizing the quality of bespoke software solutions (Suryanarayana et al., 2015).

Usability and Bespoke Software Development

Usability is frequently studied and complex NFR that is variable enough to warrant inclusion in the development process. Wallace, Reid, Clinciu, and Kang (2013) indicated that usability is a component of user satisfaction that is made up of effectiveness and efficiency, though also found that the importance of usability varies by culture. Singh and Jatain (2013) stated that usability is directly related to the perception of product quality. Göransson et al. (2003) concluded that the software development process must consider usability in requirements and must integrate usability considerations in the process. While usability drives the perception of quality, it is not clear who is responsible for including it in the design.

Despite the importance of usability, software engineers do not always address it in their process. Kassab, Neill, and LaPlante (2014) indicated that the software industry, in general, believes that achieving customer satisfaction is a greater indicator of quality bespoke solutions than defining well-built, defect free solutions. Ebert, Hoefner, and V.S. (2015) supported that idea, noting that critical failures correlate with low usability in software products. However, Göransson et al. (2003) noted that software engineers typically regard usability as someone else's responsibility, including those considerations in the product only if there is available time, and believe this occurs as usability requires creativity while software engineering follows a structured processes. The lack of addressing usability in the engineering process is one of many determinants of low quality (Kassab et al., 2014).

Bespoke Software Issues

Poor quality is prevalent in bespoke software development, resulting in higher than necessary development costs. Atkinson and Benefield (2013) noted that approximately 66% of all bespoke software projects are either delivered late or not at all and that 16.7% of software development projects exceed costs by 200% and timeline by 70%. Additionally, Ahmad (2016) cited a study indicating that 50% of software development projects between 2011 and 2015 were challenged in their ability to deliver requirements on-time and under budget. These failure rates are significant drivers of cost as Jones (2015) believed that the large number of software defects in bespoke software represent up to 60% of the development effort on large projects through extended test intervals (two to three times longer due to the high number of defects). The organization's sector is not a delimiter for this problem as Atkinson and Benefield (2013) indicated that organizations in both public and private sectors experience these cost and project overruns. The prevalence of bespoke software quality issues suggests a systematic cause.

Poor functional and nonfunctional requirements are one determining factor of low quality. Ahmad (2016) stated that software requirements elicitation and specification issues are two root causes of low bespoke software quality. This idea is supported by Kassab et al. (2014) who stated that poor requirements gathering techniques are the most frequently cited source of bespoke software failure. Gürses, Seguran, and Zannone (2013) believed that stakeholders' inability to distinguish requirements from design and functional requirements from nonfunctional requirements drives low-quality requirements.

Quality can suffer as Cleland-Huang (2014) noted that focusing on functional requirements and ignoring NFRs and other quality aspects lead to failed bespoke software solutions. Gürses et al. (2013) supported this by stating that teams often consider NFRs such as security after the architecture and design are complete, an approach that is likely to be ineffective. Architects should be involved throughout the lifecycle to build quality in from the start, though there are aspects of quality that architects do not influence.

Architecture requires more than just technical considerations to influence quality. Spinellis (2014a) noted that, as an architect on one quality-challenged project, his position afforded him the power to influence quality outcomes yet he was not authorized to direct efforts toward quality. Gürses et al. (2013) indicated that architecting a solution independent of requirements and having multiple people with independent perspectives architect the solution results in inconsistencies in the architecture. Empowering a lead architect helps to maximize quality (Bon & Mustafa, 2013).

Time, budget, and other process limitations do not always support achieving bespoke software quality. Ameller, Ayala, et al. (2013) and Patwardhan (2016) reported that time and budget limitations that constrain quality efforts are aspects of bespoke software project failure. Similarly, Atkinson and Benefield (2013) believed that the following three aspects of traditional development agreements are the basis of bespoke software failures: delivery of a system that satisfies all requirements by a defined date, assumption of a waterfall-based development process, and a change management process that affects the entire product development process. Carver, Yamashita, Minku, Habayeb, and Kocak (2015) noted that organically grown processes, budget protection, and undue pressure that causes teams to bypass policies and procedures are all causes of quality issues. Quality does require involving everyone from the beginning of the project (Kassab et al., 2014).

Inattention to implementation aspects of the solution can also cause quality issues. Patwardhan (2016) believed that variances in architecture and lack of enforcement of coding standards result in low overall product quality. Spinellis (2014b) added that bespoke solutions create supportability and maintenance issues, usually with significant ramp-up times for new developers, due to the use of lexicon and implementation details that are distinct to the organization. Singh and Jatain (2013) extended the idea that developers are a bottleneck as, in bespoke software development, the number of available developers is usually not sufficient to fulfill the number of defined requirements. Building quality in from the start and having a good process improves product quality.

Improving Bespoke Software

The primary means of improving quality mentioned in the literature revolve around process improvement, requirements management, and good architecture. Researchers have shown that process improvement that involves the customer has a positive influence on quality. Mitchell (2012) noted that software process improvement (SPI) is implemented to improve efficiency, effectiveness, and quality of the product. Chen and Huang (2009) added that improvements in the development process have a direct, positive impact on the quality of the product as well as the efficiency and effectiveness of the team, adding that teams can implement them at any number of levels. Mitchell (2012) stated that transitioning from a sequential process to an iterative process is a high-level change while improving a software inspection process is a lower-level change. Misra et al. (2014) added that code reviews and design reviews, collectively referred to as inspection, are shown to have a positive influence on bespoke software quality and should be included in the development process. Process improvement does not have to be a drastic change, however. Atkinson and Benefield (2013) noted that solving the issue of low bespoke software quality and project failures does not require implementing Waterfall, rather it does require implementing a methodology that includes end-users in the entire lifecycle of the development project. Quality leaders believe that the development process has a significant impact on product quality (Crosby, 1992; Deming, 1982; Juran & Godfrey, 1999). There are a few contemporary means of improving organizational processes.

Not all process improvement frameworks are available to all organizations, and many contemporary versions may not yield desired results. Yoon, Lee, Lee, and Yoon (2015) indicated that one SPI model—the Capability Maturity Model (CMM)—was defined to aid in the implementation of effective development processes. Lyu and Liang (2014) continued that teams can use other models like ISO-9000, Six Sigma, and Capability Maturity Model Integration (CMMI) to evaluate and improve processes, though they can be unachievable for small- to mid-size organizations. Kassab et al. (2014) found that only 32% of bespoke development projects used contemporary software quality management methods (e.g. ISO 9001, Six Sigma, CMM, or TQM). Yoon et al. (2015) added that, whether or not it is achievable, critics believe CMMI does not address all phases of the development cycle. Contemporary organizations are not implementing quality management frameworks as part of quality efforts.

Proper requirements engineering and architecture has a positive influence on bespoke software quality and may be an alternative to a formal process. Patwardhan (2016) indicated that consistent solution architecture in bespoke products results in lower development and maintenance costs. Similarly, Kassab et al. (2014) cited a study that indicated investing significant effort in requirements definition and architectural definition at the beginning of the project had an average of 92% cost savings over those that invested minimal effort at the beginning. To achieve these benefits, Singh and Jatain (2013) proposed a two-phase strategy for prioritizing requirements with the developers performing the first round of prioritization and the entire stakeholder team performing the second round. Requirements alone may not be sufficient. Patwardhan (2016) noted that a consistent solution architecture is an architecture that has a single inheritance strategy, solution organization, class design paradigm, library use paradigm, and good separation of concerns in tiers. Perhaps smaller, more informal process improvements of injecting architecture are better than formal process improvement frameworks.

Addressing NFRs through architecture early in the process improves software quality. Ahmad (2016) noted that including solution architecture early in the development process is required as it provides structure to the development team and drives product quality by meeting NFRs. Cleland-Huang et al. (2013) agreed that quality concerns should be addressed from the beginning and throughout the development of bespoke software to improve quality. Gürses et al. (2013) countered that addressing NFRs such as security should occur not only early on, but throughout the design and development cycle. A process that focuses on quality throughout the cycle may be best and proper architecture may be one means.

Architecting for reuse in a component-based pattern is one architectural approach to quality. Yi, Chanle, Lei, and Gang (2013) reported that component-based software development (CBSD) is a method of software development that decomposes complex software into multiple components to minimize complexity and maximize development efficiency. Jones (2015) believed that, while developing software components for reuse adds approximately 20% to the cost and 30% to the timeline, using reusable components reduces development costs by up to 80% and the timeline by 60%. Canessane and Srinivasan (2013) supported the idea, stating that reusability and maintainability are critical to achieving cost effectiveness. Yi et al. (2013) stated that solution architecture is essential to successful CBSD as it provides guidance and constraints for component definition and communications. Similary, Martinez-Fernandez (2013) noted that software reference architectures (SRAs) are blueprints that can be applied to multiple systems, reducing costs and time to market by minimizing diversity and complexity. Component based architecture is one way that architects help improve quality through re-use (Jones, 2015).

Research Design and Bespoke Software Development

Many studies in this literature review followed a case study design. Richardson, Casey, McCaffery, Burton, and Beecham (2012) conducted case study research of three global software engineering (GSE) cases over an 8-year span to explore factors of GSE that determine its viability. Martinez-Fernandez (2013) performed two studies, using a case study design in one study, to determine the efficacy of investing in the use of SRAs in software development and a survey design in the second study to identify the parameters used to measure the return on investment of using SRAs. Koch, Bener, Aytac, Misirli, ad Bernroider (2014) reviewed several case studies to understand how emerging economies and cultural backgrounds influences culture and management issues in software development. Patwardhan (2016) followed a multiple case study design to explore architectural, development, and deployment issues associated with bespoke software development. Ameller et al. (2013) followed a case study design when exploring how solution architects use nonfunctional requirements in their decision making on bespoke software projects. Gürses et al. (2013) presented a case report design in their study that explores the requirements engineering process in a large-scale security project.

Ramasubba et al. (2015) followed a mixed methods design, employing case study for their qualitative portion and correlation analysis for their quantitative portion, in their study on the effect of variation of process on project performance. Wallace et al. (2013) used a survey design in their study on the correlation between nationality and usability attributes of bespoke solutions. Whether case study is the sole methodology, researchers used it in many studies on bespoke software.

Critical Evaluation of Themes

The reviewed literature provided a consistent view of the need for bespoke software and the causes of issues with its development. Poor quality bespoke software is prevalent, extending project costs and timelines (Ahmad, 2016; Atkinson & Benefield, 2013; Jones, 2015). As a result, the industry has switched to focusing on customer satisfaction as a measure of quality (Göransson et al., 2003; Kassab et al., 2014; Wallace et al., 2013). The literature described the following sources of bespoke software quality issues:

- Poor requirements engineering processes,
- not including nonfunctional requirements from the start,
- unqualified expectations of project costs and delivery timeline, and

- not having a strong architecture from the outset of the development cycle (Ahmad, 2016; Ameller, Ayala, et al., 2013; Atkinson & Benefield, 2013; Cleland-Huang, 2014; Cleland-Huang et al., 2013; Gürses et al., 2013; Kassab et al., 2014; Patwardhan, 2016).

Common software development life cycle (SDLC) processes have not solved the issues with requirements or solution architecture in larger organizations as they could not always scale to their needs (Atkinson & Benefield, 2013; Despa, 2015; Ramasubbu et al., 2015; Shah, 2014). This lack of ability to address issues is troubling as SDLC processes are critical to product quality and should involve everyone (Göransson et al., 2003; Ramasubbu et al., 2015; Shah, 2014; Singh & Jatain, 2013; Verner et al., 2014). Improving the SDLC process will have a positive effect on product quality (Atkinson & Benefield, 2013; Chen & Huang, 2009; Misra et al., 2014; Mitchell, 2012).

Requirements engineering should include everyone in both functional and nonfunctional requirements (NFRs) definition (Ebert et al., 2015; Göransson et al., 2003; Ramasubbu et al., 2015; Shah, 2014; Singh & Jatain, 2013; Verner et al., 2014). Requirements should be managed throughout the development cycle as they are not fully known at the start of the project (Atkinson & Benefield, 2013; Göransson et al., 2003). Teams must balance this with the need to address quality early through NFR elicitation (Ameller, Ayala, et al., 2013; Canessane & Srinivasan, 2013; Cleland-Huang, 2014; Göransson et al., 2003; Thakurta, 2013). Architects use these NFRs to build quality into the product from the beginning (Ameller, Ayala, et al., 2013; Cleland-Huang, 2014) and architecting solutions for reuse has a positive impact on product quality (Canessane & Srinivasan, 2013; Jones, 2015; Yi et al., 2013). In summary, customer satisfaction (i.e. quality) can be positively influenced by improving the SDLC process (i.e.

process improvement), including quality from the start (i.e. product design), and including everyone in quality conversations. Several of the studies in the reviewed literature relied on case study design.

Total Quality Management

Coming from many years of work in the quality space, Deming was known as the father of TQM (Hill, 2008). Working with others such as Joseph Juran, Deming defined TQM with three primary focuses: customer satisfaction, continuous improvement, and involving everyone (Houston & Dockstader, 1997). Deming (1982) found that, as depicted in Figure 1, customers defined quality based on five aspects of a product and that there are five procedures grouped into two policies for achieving the customer-defined aspects of quality (Despa, 2015; Hallissy et al., 2016; Houston & Dockstader, 1997). No matter how it is defined, product quality—or customer satisfaction, in TQM terms—is a significant source of competitive advantage which translates into customer loyalty (Chou & Chiang, 2013; Guimaraes & Paranjape, 2014). The TQM focal points of customer satisfaction, continuous improvement, and involving everyone were congruent with the means of improving bespoke software quality.

Evolution of TQM

During the 1980s, the U.S. economy was facing significant competition from the Japanese (Deming, 1985; Joyce, 2015). Japan had been increasing the quality of products since the 1950s while the U.S. continued the status quo (Hill, 2008). To meet the competition and reestablish market share, the U.S. began to focus on the quality of its products (Joyce, 2015). U.S. leaders turned to W. Edwards Deming and his teachings of quality control, which came to be known as total quality management (TQM) after the United States Department of the Navy

(DoN) branded their quality improvement efforts by that name in 1985 (Houston & Dockstader, 1997).

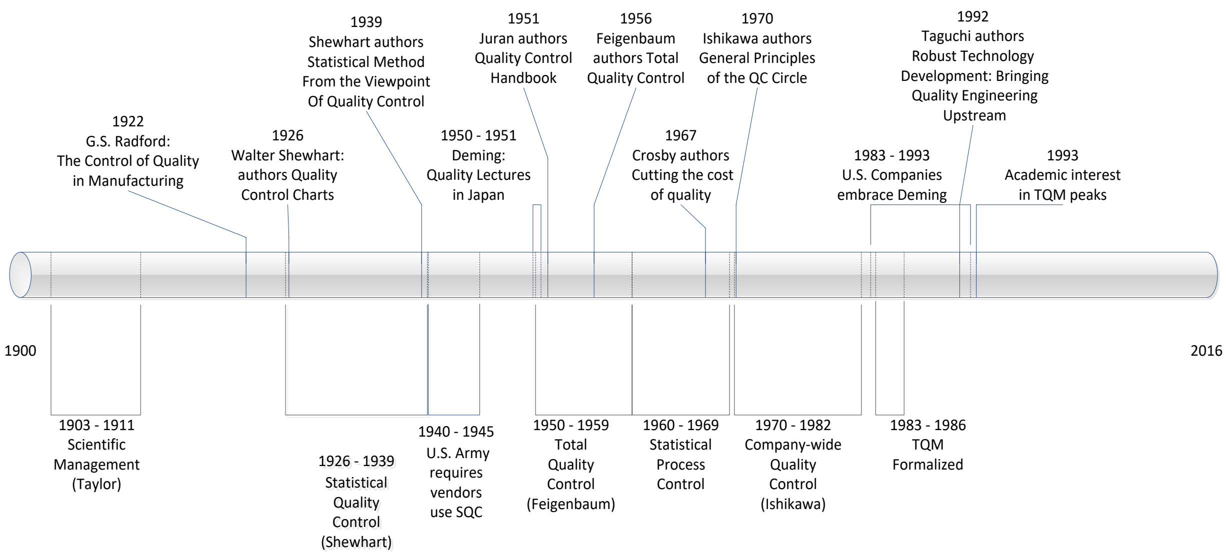

Figure 3. *Major milestones and timeframes in the evolution of TQM.*

Craftsmanship. The first evolution of quality was a shift in responsibility for quality from the individual worker to a supervisor. Joyce (2015) noted that, prior to the 1760s and the Industrial Revolution, skilled craftsmen were responsible for all stages of production and product quality. Mitchell (2012) added that individuals choosing a particular career worked with a mentor, who transferred knowledge to the apprentice over time. Joyce (2015) continued, stating that, during the Industrial Revolution, the division of labor shifted the quality focus from the craftsman to a foreman, and each craftsman was only responsible for a small portion of the finished product. The quality focus then became worker efficiency, as craftsmanship was not suited for mass production.

Scientific management. The next step in the evolution focused on the efficiency of individual work tasks. Khalil, Stockton, Alkaabi, and Manyonge (2015) noted that Henry Ford and Frederick Winslow Taylor influenced quality by applying a scientific methodology to work tasks. Khalil et al. (2015) stated that operations management started focusing on managing human, technical, and system resources. deWinter, Kocurek, and Nichols (2014) added that Taylor's visions of maximizing worker efficiency by decomposing work into distinct activities and optimizing the effectiveness of each activity became the foundation of scientific management. deWinter et al. (2014) mentioned that opponents criticized scientific management for different reasons such as being antihumanitarian; an idea supported by Taylor when he indicated that the system must replace the man as the most important consideration. At this point, the forefathers of TQM revolutionized the perception of quality as these overly mechanistic approaches had their limits.

Evolving contemporary perspectives of quality. Following the scientific management movement, the forefathers of quality began to transform the definition of quality, focusing on the customer. Deming (1982), Feigenbaum (1985), Ishikawa (1984), Juran(1999), and Shewhart (1939) redefined quality as achieving customer satisfaction. Deming (1982) and Ishikawa (1984) focused on customers' current and future needs while Juran (1999) focused on fitness for use and Shewhart (1939) focused on value for price paid. Feigenbaum (1985) and Ishikawa (1984) expanded the view of quality to include all stakeholders associated with the product (i.e. involving everyone), not just the customer. This focus on customer satisfaction was the central tenet of TQM (Houston & Dockstader, 1997).

Some of the forefathers believed that customers were tangential to the definition of quality, not central. Crosby (1992) thought that quality was conformance to requirements with zero defects. Taguchi defined quality as a measure of loss to the customer after the purchase (Shanmugasundaram & Vikram, 2015). While Shewhart (1939) believed perceived value determines quality, he also believed that quality was primarily a function of process management. These differences of opinion led to the evolution of TQM.

Customer satisfaction. Customer satisfaction is complex and includes aspects of Crosby's, Taguchi's, and Shewhart's views. Deming (1982), Feigenbaum (1985), and Ishikawa (1984) believed that product price was not enough to determine quality with Deming (1982) adding that customers were more valuable than vendors as it is easier to replace a vendor than to replace a customer. Similarly, Ishikawa (1984) argued that meeting specifications is not enough to determine quality. Shewhart (1939) defined quality as perceived value for the price paid and Juran (1999) defined quality as fitness for use (which is a complex concept itself) and customer dissatisfaction (which comprises psychological impressions and supportability). The common aspects shared by these experts agreed with Bon and Mustafa's (2013) view that quality was a combination of excellence, value, conformance to specifications, and exceeding customer expectations. Liu, Chang, and Tsai (2015) believed that product designers will not achieve customer satisfaction by addressing one or two considerations. All of the forefathers agreed that customer satisfaction is the focus of TQM and that organizations cannot achieve it all at once.

Continuous improvement. While organizations must continuously improve to achieve customer satisfaction, the forefathers expressed disagreement regarding who should be involved in continually improving the organization. Juran (1999) believed that continuous improvement (CI) should focus on managerial leadership of quality and little more. Ishikawa (1984) believed CI should extend to employees through training programs and definition of standards. Similarly, Deming (1985) felt that CI should not be limited to the internal organization, indicating it should extend to vendors as well. Involving everyone in improvement is thought to have a positive effect on quality (Feigenbaum, 1985; Ishikawa, 1984).

Improvement is a cycle that is essential to quality. Shewhart (1939) developed the plan-do-study-act (PDSA) cycle that Deming (1985) later re-branded as the plan-do-check-act (PDCA) cycle, which is the foundation of CI. Shewhart (1939) believed that constant evaluation and testing ideas are essential to quality and Crosby (1992) believed in quality measurement (baselining), quality awareness, corrective action, and removing sources of errors. Teams can base the PDCA cycle on statistical analysis, and Deming (1985) believed that statistical techniques maximize efficiency improvements in process and product design while also driving toward an ever-changing definition of customer satisfaction. Baselining and statistical analysis can help organizations achieve customer satisfaction through continuous improvement.

Not all of the forefathers agreed that statistics are critical, however. Deming (1982) believed more in statistical approaches to quality with a long-term focus based on Shewhart's work than Juran (1999), who added more human-centric aspects to quality. Crosby (1992) believed that organizations could transform through policies (zero defects), training (culture), requirements (customer needs), and enforcement of integrity, though Feigenbaum (1985) focused on cost control through total quality control. While Shewhart (1939) introduced Statistical Process Control Charts (SPCC) to monitor and manage variability, Ishikawa (1984) introduced the concepts of quality circles, the Ishikawa fishbone diagram for cause-and-effect, and the quality chain to address variability. Pure statistical process control evolved into a more complex combination of qualitative and quantitative approaches to quality.

The forefathers of quality did agree on CI as a means of addressing quality. To achieve quality, Deming (1982) believed that continuously improving the organization was the best approach while Juran (1999) and Feigenbaum (1985) believed that improving management and leadership was best. Shewhart (1939) felt that focusing on the reduction of variability in testing and experimentation was best. Continuous improvement is the first of two policies of TQM.

Everyone is involved. Continuous improvement relies on people of all levels of the organization. Deming (1985), Juran (1999), Crosby (1992), Feigenbaum (1985), and Ishikawa (1984) believed that successful quality management requires senior management commitment. Juran (1999) believed quality committees should comprise only senior leaders and Crosby (1992) felt that it is a failure to focus quality efforts on the lowest levels of organizational hierarchy as defects will always exist. Feigenbaum (1985), conversely, introduced the idea that quality is everyone's responsibility, from management to workers and through marketing, research and development, finance, purchasing, and all other departments. Ishikawa (1984) agreed with Feigenbaum, stating that commitment to quality must exist throughout the organization and quality is everyone's job with management leadership. Deming (1985) believed that quality is a culture composed of constant drive to quality, training, leadership, teamwork, and involving everyone. Involving everyone is the second policy of TQM.

Process effect on quality. Along with CI and involving everyone, the development process itself is a significant driver of quality. Deming (1982) believed that two drivers of quality are innovation of process and process improvement. Crosby (1992) thought that treating quality as a process, as opposed to a program, instills the idea that quality is a long-term consideration. To help drive this process view of quality, Ishikawa (1984) introduced the ideas of quality circles, continuous employee training, and the quality chain for aligning quality with processes. The second phase of Juran's (1999) trilogy focused on effectiveness and efficiency of processes, minimizing waste. Designing effective processes is one of the procedures for achieving customer satisfaction.

The forefathers had differences of opinion on how to maximize the effectiveness of processes. In the 1920s, Shewhart's (1939) work on SPCCs focused on the idea that process must support delivery of products that satisfy human wants reliably, only varying due to chance. While Crosby (1992) believed that quality is achieved by prevention and achieving zero defects, Deming (1985) thought that achieving zero defects is unachievable due to a focus on the overall system as opposed to individual work teams. Shewhart (1939), Deming (1982), and Taguchi all based process improvement on statistical analysis (Richardson et al., 2012) with Deming (1982) and Taguchi including the effects of variation in their analysis. Juran (1999) argued that statistical analysis is not needed, instead focusing on being able to speak to both business leaders and engineers. The variance between qualitative and quantitative views of quality extends beyond customer satisfaction to process control.

Product design effect on quality. Effective processes support proactive quality instead of reactive. Hill (2008) found that all of the forefathers agree that quality should be built-in to the product from the start. Deming (1982) believed that the third driver (after two process-oriented drivers) of quality is product innovation and design and Juran (1999) included infusing quality in design in the planning (first) phase of his trilogy. Shewhart (1939) noted that satisfying human wants through design is the first step toward quality while Feigenbaum (1985) added that quality through product design must consider the needs of marketing, engineering, manufacturing, and maintenance teams in addition to the customer. Building quality into the product from the start is another procedure for achieving customer satisfaction.

Contemporary TQM

Many of the contemporary frameworks and processes for continuous improvement of quality initiatives are based on TQM principles. Hallissy et al. (2016) noted that ISO 9000 and

CMMI are two commonly implemented quality management frameworks in contemporary organizations and Joyce (2015) added that Six Sigma is also a common implementation method. Chen and Huang (2009), Cronemyr and Danielsson (2013), Lyu and Liang (2014), Mitchell (2012), S.-H. Li, Yen, Lu, and Chen (2014), and Yoon et al. (2015) all agreed that most contemporary frameworks focus on one aspect of quality management like SPI (CMMI and Six Sigma) or documentation (ISO 9000). These atomistic approaches do not have the holistic view that TQM has.

Capability maturity model integration. Organizational leaders use CMMI to improve processes for multiple project types. Cronemyr and Danielsson (2013) stated that, while the Capability Maturity Model (CMM) focused on software development processes based on the original works of Philip Crosby, Capability Maturity Model Integration (CMMI) was an extension that supports any process. In either case, Marchewka (2013) specified that CMM and CMMI (hereafter referred to as CMM/I) only take development and maintenance process improvement into consideration, relying on an evolutionary model for organizations to improve their processes. Lyu and Liang (2014) believed that these improvements can be applied to a project, department, or enterprise to align activities with business strategy. Organizations should evaluate this framework to verify that it aligns with their culture and desires.

Even when it does align with organizational culture and projects, CMM/I may not be a cost-effective means of improving quality. Richardson et al. (2012) believed that CMM/I is not suitable for all development teams. Tunkelo et al. (2013) stated that, while researchers report higher product quality, researchers also associate it with higher development effort. Luftman et al. (2013) believed that achieving CMM/I certification is only required for vendors seeking projects to provide proof that they follow proper processes. These themes indicate that the global industry may not accept CMM/I as a quality framework. Digalwar, Haridas, and Joseph (2014) indicated the notion that ISO 9000 is the most popular approach in Indian companies supports the idea that CMM/I is not globally accepted. Whether or not it is globally accepted, CMM/I has qualities that align with the process improvement aspect of TQM.

Six sigma. Six Sigma may not be entirely new or effective as a quality improvement framework. Hill (2008) summarized Six Sigma, sometimes referred to as a method of TQM, as a business process for improving quality, reducing costs, and increasing customer satisfaction. Lyu and Liang (2014) stated that it is essentially a blueprint for one possible implementation of TQM that teams have adapted to software quality. Juran (1999) believed that implementing Six Sigma was not the best means of maximizing quality, arguing that it is more hype and just a new label placed on an old principle. Supporting Juran's perspective, Cronemyr and Danielsson (2013), Lyu and Liang (2014), and Pérez-Aróstegui, Bustinza-Sánchez, and Barrales-Molina (2015) stated that Shewhart's Statistical Process Control principles are the basis of Six Sigma. Additionally, L. R. Brown (2014) stated that Six Sigma has many of the same factors of success as TQM. Cronemyr and Danielsson (2013) added that it requires well-defined processes as a prerequisite to being successful. Regardless of whether it is a new framework, Six Sigma does have qualities that align with the statistical analysis of TQM.

ISO 9000. Successfully improving the quality of products may not be sufficient. Organizations may need to document their efforts as part of their framework. Topalović (2015) argued that achieving product quality in the eyes of the consumer is not enough, stating that quality must be proven to be of international standards and ISO 9000 certification is one model for establishing that proof. Barata and Cunha (2015) noted that ISO 9000 is a set of international standards for quality management and assurance. Joyce (2015) mentioned that it is also a blueprint for implementing a quality system across the entire product life cycle. Barata and Cunha (2015) believed that ISO 9000 by itself may not result in consistent process improvement, though also believe that TQM and ISO 9000 can be combined in implementation, creating an effective quality management framework. Digalwar et al. (2014) stated that teams should implement ISO 9000 before the implementation of a quality management solution. ISO 9000 is an integral part of quality management frameworks, though researchers think it cannot effectively stand alone.

Supporting Conceptual Models

Systems theory is one supporting model of TQM. Deming (1982, 1985) included an appreciation for systems in his 14-point management method, relating systems theory as a supporting model of TQM. Katina (2015) stated that systems theory seeks to explain behaviors of systems through propositions and axioms. Whitney, Bradley, Baugh, and Chesterman (2015) explained that some of the central axioms of systems theory include behaving in ways that drive desired outcomes, the use and manipulation of information, and that the meaning of the system is partly defined by the context in which it exists. Katina (2015) summarized systems theory as an approach to understanding current situations, which is a dependency of the CI aspect of TQM. Systems theory may simply be the qualitative variant of TQM.

With Shewhart's PDSA cycle—later revamped into Deming's PDCA cycle—being the basis of continuous improvement, PDSA and PDCA are also supporting frameworks of TQM. Yoon et al. (2015) stated that, based on the principles of the scientific method, planning is setting up a test focused on improvement, doing is executing the test, checking is examining the results, and acting is taking action based on the analysis. Houston and Dockstader (1997) stated that the PDCA cycle is also known as the learning cycle, which can lead to Deming's (1982) system of profound knowledge. Poth (2014) developed a new framework for improving product quality known as effective quality management (EQM), basing it on Deming's PDCA cycle. Continuous improvement alone may be an effective means of achieving customer satisfaction.

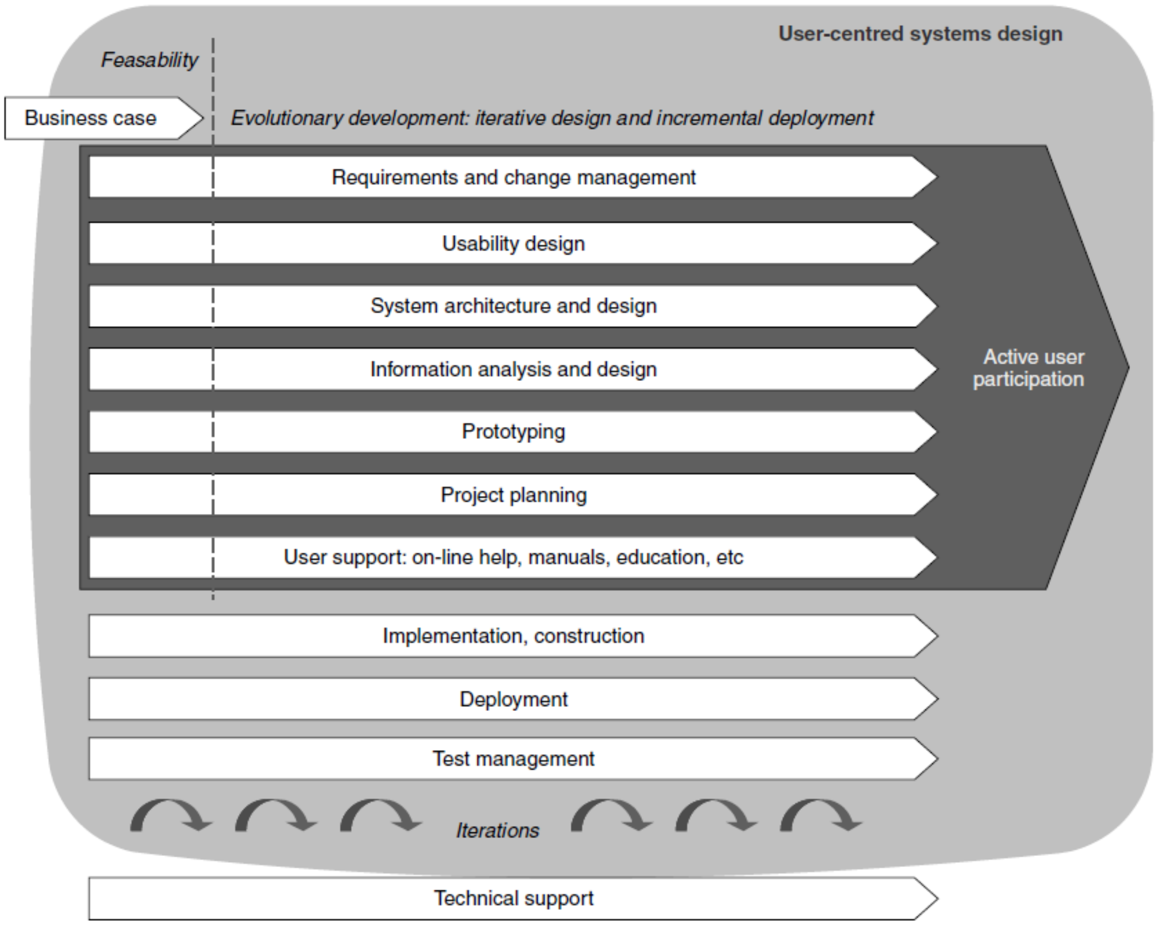

Figure 4. Draft framework for a user-centered system design (Göransson et al., 2003).

Any framework or initiative that focuses on customer involvement is a candidate supporting framework. Göransson et al. (2003) indicated that user-centered system design (UCSD) focuses on usability throughout the design and development activity (see Figure 4), making it a supporting model of TQM. Shah (2014) noted that UCSD can be culturally driven, working well in cultures that value individuals over organizations. Hill (2008) stated that TQM focuses on customer satisfaction through the design of both processes and products and

Göransson et al. (2003) added that UCSD does as well. UCSD supports TQM as it focuses on customer satisfaction, process design, and product design.

Contrasting Conceptual Models

Several of the contemporary models of quality management solutions are also contrasting models due to variances from TQM. Lyu and Liang (2014) explained that Six Sigma focuses on quality as a function of meeting specifications instead of customer satisfaction. Similarly, Misra et al. (2014) indicated that CMM/I is based on the tenets of TQM but also focuses on process improvement and control of quality through inspections as opposed to building quality into the product. Liberatore and Pollack-Johnson (2013) explained that ISO 9000, like Six Sigma, defines quality as conformance to specifications as opposed to customer satisfaction.

Another contrasting model is management by objectives (MBO). Deming (1985) explained that MBO seeks to improve organizational performance by aligning subordinate goals and objectives throughout the org, typically implemented as a supervisory rating system. Houston and Dockstader (1997) noted that this management theory is so contrary to the epitome of TQM that they recommend eliminating the practice altogether. Deming (1985) agreed, basing his perspective on the notion that the effects of MBO are a focus on short-term performance, reduction in teamwork, and lower individual morale. Spinellis (2014a) added that attempts to implement MBO are not successful at achieving higher-level objectives. The differences are so significant that MBO and TQM may be incompatible.

Critical Evaluation of Themes

The forefathers of TQM believe that quality is a complex idea and could not agree on a single view (Crosby, 1992; Deming, 1982; Feigenbaum, 1985; Hill, 2008; Ishikawa, 1984; Shanmugasundaram & Vikram, 2015; Shewhart, 1939). Many of them agreed that customers should be involved in the definition of quality and in the process of achieving quality (Deming, 1982; Feigenbaum, 1985; Hill, 2008; Ishikawa, 1984; Shewhart, 1939). All of the forefathers agreed that quality should be built-in from the start (Hill, 2008) and most also agreed that the process of developing a product is as important to quality as the product definition itself (Crosby, 1992; Deming, 1985; Hill, 2008; Richardson et al., 2012; Shewhart, 1939). Most of the forefathers believed that leadership commitment to quality is important and that quality efforts should extend to all departments and vendors (Crosby, 1992; Deming, 1985; Feigenbaum, 1985; Ishikawa, 1984; Juran & Godfrey, 1999), leading to the idea that quality should involve everyone (Deming, 1985; Feigenbaum, 1985; Ishikawa, 1984; Shanmugasundaram & Vikram, 2015). Most of the forefathers also agreed that continuous improvement is critical to quality and should be extended throughout the product development value chain (Deming, 1985; Ishikawa, 1984; Juran & Godfrey, 1999; Shewhart, 1939; Tunkelo et al., 2013).

TQM and Bespoke Software Development

As with many conceptual frameworks, Shanmugasundaram and Vikram (2015) and Topalović (2015) found that there is no widely-accepted definition of TQM. Hill (2008) defined TQM as an organization's culture, attitude, and structure towards providing customers with products that satisfy their needs. Barata and Cunha (2015) defined TQM as a philosophy that imposes a systematic view of managing an organization for the purpose of continuous process improvement. Gimenez-Espin, Jiménez-Jiménez, and Martínez-Costa (2013) defined TQM as a management tool for competitive advantage and long-term success. Houston and Dockstader (1997) and Sharma and Modgil (2015) defined TQM as a quantitative analysis to evaluate and improve everyone's ability to exceed the needs of consumers through improved purchasing, supply chain management, and process improvement. The U.S. Department of the Navy adopted

Houston and Dockstader's (1997) definition when they coined the term total quality management to describe the process. With this much variation in the definition, there cannot possibly be consistent contemporary implementation.

Even the primary components or tenets of TQM were in dispute. Liberatore and Pollack-Johnson (2013) noted that customer focus, leadership, delivery, and employee empowerment are the four components of TQM. Bon and Mustafa (2013) and Gimenez-Espin et al. (2013) believed that continuous improvement, customer satisfaction, employee empowerment, and leadership commitment are the four tenets. Talib, Rahman, and Akhtar (2013) believed that four tenets are not sufficient to describe TQM and provide alternate views that include strategic planning, information and analysis, process management, integrity, and training. Customer satisfaction, continuous improvement, employee empowerment, and leadership are the commonly perceived tenets of TQM.

Customer Satisfaction in Bespoke Software Quality

Defining a customer is critical to quality initiative leaders. Pérez-Aróstegui et al. (2015) indicated that customer focus is part of the process design where customers are continually involved in the process to attain customer satisfaction. Organizations must understand who their customer is to achieve satisfaction. Houston and Dockstader (1997) indicated that, in the context of TQM, a customer is a person who buys or uses the offered products or services. Cronemyr and Danielsson (2013) indicated Houston and Dockstader's definition refers to both the people that buy products developed and sold by an enterprise as well as its employees that consume the products and services delivered to them. Bespoke software quality initiatives focus on enterprise employees as the customer (Göransson et al., 2003).

Researchers confirm what the forefathers of quality believed, that customer satisfaction with bespoke software, or quality, is subjective. Goode, Lin, Tsai, and Jiang (2015) indicated that customers have perceptions about the bespoke products they buy and the services with which they interact. Liu et al. (2015) believed it is a combination of these perceptions that determine their satisfaction with, or perceived quality of, the bespoke product or service. Hill (2008) noted that the five identified considerations that affect perceptions of quality are psychological impressions, fitness for use, supportability, meeting specifications, and value. I used the details of the five components of customer satisfaction to highlight the differences in roles associated with bespoke software quality.

Psychological impressions and customer satisfaction. How customers feel about using bespoke products directly affects their perception of quality. Dai, Luo, Liao, and Cao (Dai, Luo, Liao, & Cao, 2015) believed that one of the most poignant aspects of quality is psychological impressions, which includes emotional and social factors. Zhou, Ji, and Jiao (2013) described psychological impressions as indicators of how the customer feels about the product and using it. Ding, Yang, Zhang, Liang, and Xia (2014) explained that ideas of reliability, credibility, security, trust, intimacy, social responsibility, and environmental management all feed into this one perception of quality. Dai et al. (Dai et al., 2015) and Liu et al. (2015) posited that satisfaction is a combination of perceived value and trust while Singh and Jatain (2013) stated that usability is directly related to the perception of bespoke product quality. Ameller, Galster, Avgeriou, and Franch (2013) found that usability is one of the most important aspects related to the perception of quality. No matter how well a product conforms to specifications or how mature the process used to develop the product is, customers will perceive low quality if they become frustrated while using the product or service.

Fitness for use and customer satisfaction. Fitness for use, like psychological impressions, is itself a complex consideration. Atkinson and Benefield (2013) and Belk, Papatheocharous, Germanakos, and Samaras (2013) found that customers tend to evaluate a bespoke product's or service's fitness for use, or how well it enables them to achieve desired outcomes. Ding et al. (2014), Lyu and Liang (2014), Shah (2014), and Shanmugasundaram and Vikram (2015) found that fitness for use is comprised of availability, features that meet needs (a.k.a. utility), safety, and overall design. Tunkelo et al. (2013) added aspects of integrity, flexibility, and reusability to the concept of fitness for use. While overall fitness for use is a component of quality, availability and reusability are two aspects of fitness for use that are common considerations in architectural approaches to quality (Cleland-Huang, 2014).

Supportability and customer satisfaction. Customers may perceive low bespoke software quality if it requires significant effort to correct issues when they occur. Mihalcin, Mazzuchi, Sarkani, and Dever (2014) indicated that how well team members support a product or service when something goes wrong is another aspect that affects a customer's perception of quality. Tunkelo et al. (2013) stated that customers have ideas of service-level agreements (SLA) that define resolution times for support issues. Any variance from those SLA agreements affects the customer's perception of quality. Mihalcin et al. (2014) added that the net effect of resolution time on quality also depends on the customer's perceived level of significance of the issue. Supportability and maintainability are two aspects of bespoke software that teams do not always address in bespoke software development (Chen & Huang, 2009) yet architects cover them in NFRs.

Meeting specifications and customer satisfaction. Bespoke software specifications must reflect proper requirements to be valuable. Huckabee (2015) explained that technical and functional specifications for a product or service define what the product or service should do from the customer's perspective. Bon and Mustafa (2013) and Shanmugasundaram and Vikram (2015) found that some perspectives define quality purely as conformance to these specifications and Shah (2014) added that teams must significantly consider maintaining the currency and accuracy of specifications. Shanmugasundaram and Vikram (2015) concluded that, as a neutralizing effect, customers view conformance to specifications as one of many aspects of a product that drive their satisfaction. The quality of the requirements in the specifications is only one aspect that influences the perception of quality (Ahmad, 2016).

Value and customer satisfaction. Cost and importance are central to perceived value of bespoke software. Tyagi, Choudhary, Cai, and Yang (2015) believed that only customers can define value and all of the previously mentioned aspects of customer satisfaction help customers form that sense of value when attempting to perform a task. Thakurta (2013) added that the perception of importance informs their idea of value. It appears that the greater importance a product or service has when completing a task, the more valuable it is.

Hill (2008) and Liu et al. (2015) believed that product price, or cost to the customer, is an integral part of the value proposition, as are overall quality and availability of the product when it is needed or desired. Goode et al. (2015) believed that perceived security is also an important aspect of value, though security is a relatively unstudied in the context of customer value. Price paid influences perceived value in combined terms of fitness for use, psychological impressions, supportability, and meeting specifications.

Quality is complex. Achieving bespoke software quality takes practice and cannot be attained by addressing any single aspect of satisfaction. Lyu and Liang (2014) believed that all of these elements of customer satisfaction interact with one another to form a perspective of quality and achieving customer satisfaction requires good product design as well as good process design. Chou and Chiang (2013) believed that customer satisfaction is reduced to a matter of trust in a provider's ability to meet needs. Deming (1985) stated that regularly measuring and improving the perceived quality of the product is critical to long-term success. Implementing a set of policies and procedures that constitute a culture of quality could help organizations maximize customer satisfaction (Deming, 1982).

Policies of TQM

The policies of TQM help organizations achieve quality culture. Hill (2008) stated that achieving customer satisfaction is realized through continuous improvement and involving everyone. Alotaibi (2014) found that, of the practices associated with implementing TQM, customer focus was one of a few that directly influenced the culture of quality. Deming (1982) believed that the combination of continuous improvement and involving everyone drives a culture of quality in an organization. Establishing a quality culture may be the best means of achieving customer satisfaction.

Continuous improvement policy. Implementing a quality culture depends on a means of improvement. Hill (2008) and Huang, Wu, and Chen (2013) defined CI as a policy used to determine opportunities for improvement by evaluating existing processes. Bon and Mustafa (2013) and Digalwar et al. (2014) stated that CI is central to quality management solutions and Shanmugasundaram and Vikram (2015) believed that is true as perfection is an asymptote; organizations will always be improving as customer satisfaction is always changing. Bon and Mustafa (2013) and Talib et al. (2013) stated that properly implemented CI can create or improve overall quality and customer satisfaction. According to TQM, customer satisfaction is the goal of the bespoke software industry and CI should focus on achieving satisfaction.

Establishing a quality culture can be difficult. Hill (2008) found that impediments to effectively implementing continuous improvement include poor training initiatives, poor planning, poor leadership, changes in top management, and poor customer focus, potentially due to the risk associated with a new leader implementing significant change programs. Understanding the current state of an organization's processes may help as Bon and Mustafa (2013) mentioned that successful CI is dependent on benchmarking, which may include collecting data from both customers and internal sources for use in statistical process control. Organizations should work to overcome these barriers as Tunkelo et al. (2013) believed that failure to implement CI may be part of the cause of low software quality. Quality should be viewed as a culture (Deming, 1982) to improve the quality of bespoke software and should start with benchmarking.

Everyone is involved in quality policy. Establishing a quality culture takes a team. A. Brown (2014), Digalwar et al. (2014), Gimenez-Espin et al. (2013), and Hill (2008) found that CI is a means of involving everyone and it must be instilled in organizational culture to be effective. Joyce (2015) suggested that having one or more person(s) that champion CI and quality culture can have a significant impact on the success of a TQM implementation. Involving everyone from customers to product designers and process engineers helps to improve quality (Feigenbaum, 1985).

Empowering employees is part of implementing quality culture. Deming (1982) states that quality should be everyone's focus for a successful implementation. Bon and Mustafa (2013) believed that empowering employees to drive quality by stopping processes when quality is low is a trend in some industries and employee empowerment improves the effectiveness of TQM. Bon and Mustafa (2013) also found that including employees in the implementation of TQM affects commitment and inventiveness. Empowering leaders helps to instill a quality culture in the organization (Bon & Mustafa, 2013).

Procedures of TQM

There are a few key procedures in TQM that can help improve the quality of bespoke software. Despa (2015) and Hallisey et al. (2016) reported that designing effective processes, providing problem-solving tools and services, designing a product or service that attends to the components of quality, and benchmarking are procedures for implementing quality culture. Effective processes and embedding quality in the product from the start will improve customer satisfaction (Feigenbaum, 1985; Hill, 2008; Juran & Godfrey, 1999).

Process design procedure. There are a few means of improving existing processes. At the project team level, Mitchell (2012) found that improving project-level shared knowledge within the team has a positive influence on the process, specifically efficiency and product quality. Additionally, Tunkelo (2013) found that implementing continuous improvement and linking improvement to organizational quality culture results in improved processes. No matter what the means of improving processes, Huang et al. (2013) stated that clear goals aligned with business objectives should provide the basis for process improvement and A. Brown (2014) noted that the Pareto principle should be used in process improvement to ensure teams focus on the most critical processes. Improvements in the software development process directly improve the quality of the product (Chen & Huang, 2009).

Process improvement can directly improve bespoke software quality. Chen and Huang (2009) found that software projects that implement process improvement have higher levels of maintainability in their products, potentially as a result of well-defined and clearly documented processes. Marchewka (2013) agreed with Chen and Haung by concluding that performance, product quality, and team productivity are directly proportional to process maturity levels. These authors provide a contemporary validation of the forefathers' thoughts on the effect of processes on product quality.

Problem-solving procedures and tools. Having defined means for resolving problems when they happen influences a customer's perception of quality. Tunkelo et al. (2013) stated that, when a user experiences a difficulty or problem while using a software solution, the means of and total time to resolve those issues affects their perception of quality. Hallissy et al. (2016) noted that one potential means of minimizing problem resolution times is to provide self-help tools such as training, tutorials, and user guides. Tunkelo et al. (2013) added that when a user has none of these means of solving problems on their own and have to call for support, tiered support teams result in increased resolution time for each tier involved in the resolution. The nature of problem-solving could be the basis for extended resolution times as Houston and Dockstader (1997) found that resolution methods rely on the process of identifying and resolving problems using a structured approach, such as Deming's PDCA cycle. How quickly and easily problems are solved influences the psychological impressions a user has of bespoke software solutions, affecting their satisfaction.

Product or service design procedure. Including quality from the start influences customers' perceptions of the product as well. Deming (1982) believed that, instead of relying on inspections to identify problems, teams should build quality into product design. Huckabee (2015) and Joyce (2015) stated that customers should be involved in product or service design from the beginning to maximize their satisfaction. One means of involving customers early in product design is the use of prototypes. Despa (2015) and Göransson et al. (2003) stated that the use of prototypes to validate design with stakeholders improves the overall quality of the product. Prototypes can be a low-cost means of addressing quality as Cleland-Huang et al. (2013) noted that teams build prototypes with specialized tools for rapid development to minimize investment and maximize potential quality. Showing the customer that quality is built-in to bespoke software from the start influences their perception of quality.

Benchmarking procedure. Continuous improvement as a basis for improvement must have a means of comparison. Bon and Mustafa (2013) and Talib et al. (2013) stated that continuous improvement depends on benchmarking. Digalwar et al. (2014) and Aleti, Buhnova, Grunske, Koziolek, and Meedeniya (2013) noted that teams can implement CI by directly measuring the attributes of a product related to its quality, by measuring improvements in processes, or by measuring solution architecture optimizations. Gimenez-Espin et al. (2013) and Tunkelo et al. (2013) added that benchmarking can also be implemented to gauge quality and performance improvements in software solutions over time. Including customers in benchmarking surveys to influence the quality of the bespoke software over time will affect their satisfaction, and there are two critical roles responsible for involving the customer.

Architecture and Engineering Roles in Bespoke Software Development

Businesses are transforming the way they view IT and roles associated with bespoke software development are adapting to this transformation differently. Andriole (2015) found that contemporary businesses are transforming by replacing the business/IT divide with a strategic/operational divide. Andriole (2015) also indicated this presents an issue as the majority of IT workers have little to no expertise in what their business constituents do. Organizations include enterprise architecture in their structure to address this concern as Keeling (2015) noted that promoting business agility requires sound architecture. Simon, Fischbach, and Schoder (2013, 2014) added that organizations depend on enterprise architecture teams to align people, processes, and IT with business strategy. Enterprise architecture teams bring software engineers and customers together to achieve quality.

Software engineers and engineering. Software engineers are relatively unstudied aspects of bespoke software development. Mitchell (2012) stated that software implementation (i.e. writing the code) is the phase of software development that has the greatest amount of associated research. P. L. Li et al. (2015) agreed that most research has focused on aspects of software engineering other than the software engineers themselves. The literature has a gap that substantiates this research. Software engineers are one critical role involved in creating high-quality bespoke software (P. L. Li et al., 2015).

Software engineers. Opinions on components or aspects of a good software engineer—regarding their effect on product quality—vary nearly as much as definitions of TQM. P. L. Li et al. (2015) believed that software engineers must have design analysis skills, the ability to work in culturally diverse teams, exposure to large-scale development, and ability to write quality code. Alfaro and Chandrasekaran (2015) believe that software engineers must be able to communicate, coordinate, and resolve conflict well within a team, noting that the critical elements are being able to share and integrate information. Of 53 identified attributes, P. L. Li et al. (2015) noted that passion for work, continuous improvement, sound decision-making skills, ability to deliver elegant solutions that anticipate need, and evaluating the colloquial big picture are the most critical aspects of a good software engineer. Mitchell (2012) added that the ability to share knowledge through communication and adaptive mental models is important as well. A good software engineer has attributes relating to the implementation (i.e. writing code) and delivery aspects of bespoke software.

Despite some beliefs, there is a significant difference between software engineers and solution architects. P. L. Li et al. (2015) indicated that software engineers with over 25 years of experience could be considered architect-level software engineers. However, Spinellis (2015) noted that software engineers do not always have time to consider changes or requirements from an architectural perspective, adding that architecture can require significant investment in education, training, and demands on time while on the job. Software engineers are not afforded the time to address quality considerations of bespoke solutions.

Even when afforded the time, software engineers do not always focus on customer satisfaction. Tunkelo et al. (2013) noted that software engineers do not always include all elements of customer satisfaction or usability in their design as they focus on other aspects of the solution. Göransson et al. (2003) indicated that solution usability is one consideration of customer satisfaction usually viewed as secondary and someone else's responsibility by software engineers. Huckabee (2015), agreeing with the concepts of TQM, stated that software engineers should embrace customer involvement to avoid reducing trust between customer and project or program manager.

Additionally, organizations do not motivate software engineers towards customer satisfaction. Verner et al. (2014) found that software engineer motivation is one of the leading influencers of product quality and factors that influence the motivation of individual software engineers vary by geographic region. Lyu and Liang (2014) indicated software engineer motivation comprises some combination of effective project managers, effective scope and risk management, customer focus, and software engineer rewards. Generally speaking, Göransson et al. (2003) noted that customer focus is considered very low in importance for motivating software engineers. Customer satisfaction is dependent on a role other than software engineer in bespoke software development driving toward quality.

Software engineering. Following a well-defined software engineering process is critical for software quality. Shah (2014) found that, while critics indicate that traditional processes are overly mechanistic and the industry is moving toward more people-centric processes, the process of developing bespoke software in practice frequently varies from recommendations. Suryanarayana et al. (2015) stated that quality is adversely affected by this variance because the effectiveness of a process is critical to the quality of the product. The variance in process affects other aspects of product quality as well. Dehaghani and Hajrahimi (2013) reported that the maintenance phase of software engineering represents around 90% of the total cost of developing software, adding that the top five influencers of that cost are as follows:

- Project considerations (modern practices and tools),
- maintenance considerations (documentation quality, team capability, and experience with the software),
- personnel considerations (experience and capability level),
- product considerations (required reliability and complexity), and
- hardware considerations (processing time and storage limitations).

Software engineers are not always responsible for quality issues. Chen and Huang (2009) reported that poor documentation of requirements, poor design, and improper initial coding negatively affect product quality. Aleti et al. (2013) found that, in some cases, software engineers and the software engineering process rely on solution architects to deliver solution architecture artifacts to start the development process and handle ambiguity. Software engineers and solution architects working together toward customer satisfaction is an implementation of involving everyone.

Enterprise and solution architects and architecture. Architects of all types are critical to the quality of bespoke solutions. Woods (2014) stated that there are a wide variety of both architects and definitions of architect responsibilities in the industry. Aleti et al. (2013) found that architectural decisions have a significant impact on product quality by reducing complexity and proving a blueprint for engineering tasks. Rozanski (2015) agreed that architects are valuable, adding that it is important that they not demand perfection and be open to compromise. Architects work with customers and software engineers to improve product quality (Ahmad, 2016).

Enterprise and solution architects. Enterprise and solution architects have the perspective that software engineers cannot have. Woods (2014) stated that application and enterprise architects focus their efforts on design-centric activity, leadership, stakeholder focus, system-wide concerns, full life-cycle involvement, and balancing concerns. Suryanarayana et al. (2015) found that an architect's primary influence on quality is through product design and process design. Product design and process design are two procedures for achieving quality in TQM (Hill, 2008).

Balancing strategy with operations is one type of balancing concerns. Spinellis (2014a) believed that balancing short and long-term goals has a positive effect on bespoke software quality and Woods (2014) stated that the influence of architectural guidance and teams may not immediately see the influence as architects tend to focus on long-term, strategic views over short-term, tactical views. Klein (2016) noted that the means of architects adding value change over the life of the product and proposes different types of architects for various phases: initial designer, extender, and sustainer. Avgeriou, Kruchten, Nord, Ozkaya, and Seaman (2016) mentioned that the idea of technical debt, or variations from the desired architecture to meet timelines, is a means of balancing short- and long-term needs by tracking those variations so they can be dealt with in the future, albeit at a higher cost. Extensibility, maintainability, availability, and security are long-term considerations addressed by architecture (Canessane & Srinivasan, 2013; Cleland-Huang, 2014).

Requirements engineering including NFRs and knowledge of business processes are critical to quality. Researchers van Vliet and Tang (2016) indicated that decisions are made based on what people know at the time and the nature of the problem. Given that enterprise architects align people, processes, and IT with business strategy, they are required to be knowledgeable of business functions. Ameller, Ayala, et al. (2013) reminded us that architects are the primary sources of NFRs and Cleland-Huang (2014) reminded us that architects use NFRs to make decisions that influence bespoke software quality, including choosing from myriad architectural patterns and styles that affect extensibility, maintainability, availability, and security. Given this, empowering architects to make decisions may directly improve quality. Bailey, Godbole, Knutson, and Krein (2013) stated that architect empowerment is so valuable that they suggest that organizations should adapt to technical structural changes dictated by architects. Aligning products and services with business needs and building quality in from the beginning drive customer satisfaction.

Enterprise and solution architecture. Enterprise and solution architecture teams work together tactically to drive customer satisfaction. Simon et al. (2013) stated that enterprise architecture (EA) is a strategic concept implemented to help align people, processes, and IT with business strategy, IT cost management, and platform portfolio management. Yi et al. (2013)

stated that solution architecture is one of four components of EA with the others being business architecture, data architecture, and technical architecture. Spinellis (2015) believed that solution architecture is most beneficial to bespoke software quality regarding maintainability, though it can also support improvements in the development process and reusability. Yi et al. (2013) indicated that the combination of component-based software development and enterprise architecture improves maintainability of bespoke solutions through integration and sharing of resources, resulting in a long-term improvement in the quality of the development process. Enterprise and solution architects drive quality in ways that software engineers cannot.

Solution architects use NFRs to maximize quality attributes of bespoke software. Cleland-Huang (2014) noted that, from the perspective of product design, clear requirements—both functional and nonfunctional—are critical for establishing a valid solution architecture. Spinellis (2015) believed this is true as solution architecture is about the principal components of a solution and the interactions between them, continuing that inter-component communication is directly related to quality attributes of performance, reliability, and scalability. Horcas, Pinto, and Fuentes (2016) indicated that typical quality attributes of software systems include availability, extensibility, interoperability, performance, reliability, scalability, security, and safety. Ameller, Galster, et al. (2013) and Göransson et al. (2003) found that quality attributes—such as performance, usability, security, safety, reliability, and scalability—play an important role while creating solution architectures, actually driving architecture in some cases. NFR elicitation is one of the challenges in bespoke software development that results in low customer satisfaction.

Solution architects work to improve reuse through variability. Galster et al. (2014) stated that variability, or the ability to adapt a software system to varying contexts and requirements, is also driven by solution architecture. Horcas et al. (2016) stated that, conceptually similar to reusability in this context, teams can achieve variability by interweaving quality attributes with a software product line framework. Reuse of systems or system components can improve the perception of quality in bespoke solutions.

Research Design in TQM, Architecture, and Software Engineering

The qualitative case study design appears to be popular in the context of information technology and quality management research. Martini, Bosch, and Chaudron (2015) performed an exploratory multiple case study on the factors that result in architectural technical debt in software engineering. Richardson et al. (2012) performed case study research to identify factors for managing global software engineering teams. Mitchell (2012) followed an exploratory case study design to understand impediments to the creation and maintenance of shared knowledge in software development teams. Of the studies on quality management represented in the selected literature, more than 15 used a case study design. Similarly, of the studies involving solution architecture or software engineering, more than 13 used a case study design. Of the remainder of studies represented in the literature review, some relied on data retrieved in other studies that used a case study design.

Critical Evaluation of Themes

Organizations of all types that choose to develop software for internal use regularly experience low bespoke software quality (BSQ). Software that is developed in-house for professional users is considered bespoke software (Göransson et al., 2003). Low BSQ is prevalent in the industry (Ahmad, 2016; Atkinson & Benefield, 2013; Jones, 2015) and the literature describes the following sources of BSQ issues:

- inadequate processes (i.e. SDLCs) for addressing quality needs;

- poor requirements engineering processes;
- not including nonfunctional requirements from the start;
- unqualified expectations of project costs and delivery timeline;
- not having a strong architecture from the beginning of the development cycle;
- lack of enforcement of good programming standards; and
- not implementing process improvement frameworks such as CMMI, Six Sigma, or ISO 9000 (Ahmad, 2016; Ameller, Ayala, et al., 2013; Atkinson & Benefield, 2013; Cleland-Huang, 2014; Cleland-Huang et al., 2013; Gürses et al., 2013; Kassab et al., 2014; Patwardhan, 2016; Ramasubbu et al., 2015).

Neither of the commonly implemented contemporary software development life cycles (SDLCs)—Waterfall or Agile—adequately address quality needs in all bespoke development contexts (Ramasubbu et al., 2015) resulting in low quality (Chen & Huang, 2009). Involving architecture early in the bespoke software design cycle to manage and address nonfunctional requirements significantly improves quality (Canessane & Srinivasan, 2013; Galster et al., 2014). Inadequate requirement engineering processes are a significant source of problems in bespoke software development (Chen & Huang, 2009) and nonfunctional requirements are critical to BSQ (Thakurta, 2013) as architects use them to drive usability, reliability, performance, efficiency, security, and maintainability (Ameller, Ayala, et al., 2013; Canessane & Srinivasan, 2013; Cleland-Huang, 2014). Project management constraints such as timeline, budget, and resource allocation are also sources of low BSQ (Ameller, Ayala, et al., 2013). I summarized the issues with bespoke software quality as follows:

- Not having an effective process (poor requirements engineering, not including architecture from the start, and not implementing process improvement),

- poor product design (not having strong architecture and not enforcing good programming standards), and
- not including everyone in quality culture (unqualified expectations of project costs and timelines).

TQM as a Conceptual Framework

TQM aligns very well with both the study of BSQ issues and as a means of addressing low BSQ. The forefathers of TQM define quality as customer satisfaction (Deming, 1982; Feigenbaum, 1985; Ishikawa, 1984; Juran & Godfrey, 1999; Shewhart, 1939). They also defined value at price paid, meeting specifications, fitness for use, and psychological impressions as components of customer satisfaction (Deming, 1985; Feigenbaum, 1985; Ishikawa, 1984; Juran & Godfrey, 1999; Shewhart, 1939). The following policies of TQM address these aspects of customer satisfaction: continuous improvement (Deming, 1985; Feigenbaum, 1985; Juran & Godfrey, 1999; Shewhart, 1939) and involving everyone in quality (Deming, 1985; Feigenbaum, 1985; Ishikawa, 1984). The procedures for implementing TQM include designing effective processes (Crosby, 1992; Juran & Godfrey, 1999) and designing quality products (Deming, 1985; Feigenbaum, 1985; Juran & Godfrey, 1999; Shewhart, 1939). Achieving continuous improvement depends on a process of benchmarking (Bon & Mustafa, 2013) and designing high-quality products is dependent on including problem-solving tools (Despa, 2015; Hallissy et al., 2016) and planning effective purchasing (Houston & Dockstader, 1997; Sharma & Modgil, 2015).

TQM and Bespoke Software Quality

In bespoke software development, customers are employees that use the software developed by the organization (Cronemyr & Danielsson, 2013). These customers have a complex view of quality (Goode et al., 2015) that is made up of psychological impressions, fitness for use, supportability (a.k.a. maintainability), value, and meeting specifications (Hill, 2008). Psychological impressions are a measure of how customers feel about using a product (Zhou et al., 2013) including ideas of reliability, security, and usability (Ding et al., 2014; Galster et al., 2014; Singh & Jatain, 2013). Usability is a driver of satisfaction based on efficiency and effectiveness of use by the user (Wallace et al., 2013).

Fitness for use is a perception of how well bespoke software enables users to achieve their desired outcomes (Atkinson & Benefield, 2013; Belk et al., 2013) including the concepts of availability, usability, and flexibility (Shah, 2014; Tunkelo et al., 2013). Supportability is a perception of how easy it is to correct issues when they occur and total resolution time is an aspect of supportability that influences perceptions (Mihalcin et al., 2014). Specifications include the functional and nonfunctional requirements of a system (Huckabee, 2015) and conformance to specifications is one component of customer satisfaction (Shanmugasundaram & Vikram, 2015). Customers should be involved in writing and validating these specifications to ensure high levels of quality (Singh & Jatain, 2013). Value is influenced by the price paid, psychological impressions, fitness for use, supportability, and meeting specifications (Tyagi et al., 2015). Quality is complex and teams cannot achieve it by addressing any one of these aspects alone (Lyu & Liang, 2014).

Addressing Low Bespoke Software Quality with TQM

Implementing TQM is analogous to creating a culture of quality. CI is central to implementing quality management systems (Bon & Mustafa, 2013; Digalwar et al., 2014) and failure to implement CI is one means of lowering BSQ (Tunkelo et al., 2013). Benchmarking key quality metrics provides the basis for implementing CI (Bon & Mustafa, 2013) and establishing a culture of quality based on CI requires including everyone in quality (A. Brown, 2014; Digalwar et al., 2014; Gimenez-Espin et al., 2013).

Linking CI to both organizational quality culture and business objectives results in well-designed SDLCs (Huang et al., 2013; Tunkelo et al., 2013) and well-designed SDLCs lead to higher quality bespoke software (Chen & Huang, 2009). Designing good bespoke software includes making self-help tools (e.g. training, tutorials and user guides) available to users and building quality in from the start via proper requirements and good architecture. Prototypes are used to involve customers early to achieve quality (Despa, 2015; Göransson et al., 2003). These themes in TQM align very well with the contemporary view of identifying and addressing issues with bespoke software. For example:

- The industry is focusing on customer satisfaction as a measure of quality (Göransson et al., 2003; Kassab et al., 2014; Wallace et al., 2013).
- SDLC processes are critical to product quality and should involve everyone (Göransson et al., 2003; Ramasubbu et al., 2015; Shah, 2014; Singh & Jatain, 2013; Verner et al., 2014).
- Quality must be addressed early through NFR elicitation (Ameller, Ayala, et al., 2013; Canessane & Srinivasan, 2013; Cleland-Huang, 2014; Göransson et al., 2003; Thakurta, 2013).

Software Engineer and Solution Architect Influence on Bespoke Software Quality

Software engineers are critical to BSQ (P. L. Li et al., 2015) though do not always include all aspects of customer satisfaction or usability (Göransson et al., 2003; Tunkelo et al., 2013). They consider customer satisfaction and usability as not important or someone else's responsibility (Göransson et al., 2003). For various reasons, software engineers frequently vary from recommended SDLC processes (Shah, 2014), though poor documentation of requirements does negatively influence their ability to achieve quality (Chen & Huang, 2009).

Enterprise architecture is helping organizations bridge the strategic/tactical gap in contemporary businesses (Andriole, 2015; Simon et al., 2013). Architects of all types are equally as important to BSQ as software engineers (Aleti et al., 2013) and typically add the most value through the design of processes and products (Suryanarayana et al., 2015). Architects help to improve the issue of low-quality requirements by driving and using nonfunctional requirements to influence BSQ and align to business strategy (Simon et al., 2013). Architects also work to balance short and long-term needs to improve BSQ (Spinellis, 2014a), though teams may not immediately realize architectural value as architects tend to focus on long-term considerations (Woods, 2014).

Summary

Despite the inclusion of both software engineers and solution architects in the development of bespoke software solutions, low BSQ remains prevalent in the industry. When viewed through the lenses of TQM and industry-standard expectations of the roles, the two have sufficient differences of responsibility related to achieving high-quality bespoke solutions. In both research and practice, BSQ remains low despite role alignment to achieve quality.

I focused this literature review on articles that detailed the aspects of quality that accentuate the difference between software engineers and solution architects. Each of the five components of quality as perceived by customers—psychological impressions, fitness for use, meeting specifications, support, and perceived value—have aspects that highlight these differences. The following examples as stated in the literature emphasize the rationale for reviewing the selected literature:

- Psychological impressions include the concept of usability and software engineers believe that usability is not their responsibility.
- Fitness for use includes the concepts of availability and usability. Software engineers are not always able to consider these architectural perspectives.
- Customers consider meeting specifications as one aspect of quality and software engineers tend to treat conformance to specifications as quality.
- Nonfunctional requirements (NFRs) partially control bespoke software quality. Either architects are the source of NFRs or NFRs are not included in specifications.
- Bespoke software development processes should include everyone though software engineers do not always consider customer perspectives.

Transition and Summary

In this section, I introduced my research, including a brief background, problem and purpose statements, research questions, an introduction to the selected conceptual framework, and a literature review of the framework and topic of the study. Bespoke software has quality issues that I summarized as having a poor requirements engineering processes, not including nonfunctional requirements from the start, having unqualified expectations of project costs and delivery timeline, and not having a robust architecture from the outset of the development cycle. According to TQM as a conceptual framework, quality is a culture that involves everyone in the continuous improvement of both processes and products to maximize customer satisfaction. In this context, I summarized the issues associated with bespoke software quality as not having an effective process (poor requirements engineering, not including nonfunctional requirements or architecture from the start), poor product design (not having strong architecture from the start, not having good requirements), and not including everyone in quality culture (unqualified expectations of project costs and timelines). It is unclear, according to the literature, why two traditional roles involved in bespoke software development (software engineers and solution architects) are not able to achieve higher levels of customer satisfaction working together.

I defined the planned execution of the study in section two. I included the intended population and sampling method to select participants; details of the research methodology and design that I used; how I collected, organized, and analyzed data; and the ethical, reliability, and validity considerations that I implemented in section two. I presented the results of executing the research and analyzing the data, drawing conclusions, and reporting on impacts to society and the target population in section three.

Section 2: The Project

Purpose Statement

The purpose of this qualitative exploratory case study was to explore strategies used by software and enterprise architects for applying architectural best practices to improve bespoke software quality, lowering the total cost of ownership. The population for this study comprised application and enterprise architects associated with the delivery of bespoke software for the enterprise architecture team at a large enterprise in the Nashville, TN metropolitan area. A wide variety of organizations can use the findings from this study to help realize and understand the benefits of having strategies architects can apply to improve the quality of bespoke software solutions. The potential social impact of this study is improved work-life balance, morale, and productivity of software and enterprise architects through streamlining development and maintenance activities.

Role of the Researcher

The nature of qualitative research dictates that the researcher is a primary data collection instrument (Yin, 1981). In my study, as the sole researcher, I was that primary data collection instrument. I played the role of the interviewer and collector of all data from all sources that informed the study, analyzed all the data, and authored the final report. I was familiar with the topic of this research as I had served in various roles as a solution architect and developer since 1995. I had lived in the target geographic area (metropolitan Nashville, TN area) since 2011 and had performed work at three major organizations in the area. Additionally, the case organization that agreed to participate in my research was the enterprise architecture team (EAT) at my employer's organization. The EAT was a completely separate organizational structure from the structure I reported to and I had no supervisory or subordinate roles with anyone in the EAT. I did work with one member of the EAT on a regular basis.

Ethical research, as dictated by the Belmont Report and related protocols, requires a balance between beneficence, justice, and respect for persons in all studies, which are partly achieved through the use of informed consent (Faden et al., 2013; Lantos & Spertus, 2014; Ryan et al., 2014). The epitome of ethical research is ensuring the researcher attends to all of the following:

- The balance between risk and benefit,
- that the participants exposed to the risk are the ones who realize the benefit, and
- that potential participants control whether or not they participate (Faden et al., 2013; Lantos & Spertus, 2014; Ryan et al., 2014).

I achieved this balance by treating all participants equally, fairly, with respect, and by following the processes described in the ethical research section.

Bias is also a significant part of ethical research as it could influence the results of the study (De Massis & Kotlar, 2014; Hyett et al., 2014; Yin, 1981). In qualitative research, bias is mitigated through the use of multiple types of data sources, performing interviews with more than one interviewee and from varying teams within the organization, following an interview protocol, and implementing member checking (De Massis & Kotlar, 2014; Hyett et al., 2014; Yin, 1981). Acceptable sources of data used to minimize bias include interviews, documentation, historical records, and direct observation (De Massis & Kotlar, 2014; Hyett et al., 2014; Yin, 1981). I collected data from multiple sources, including interview data and organizational documents such as policy and procedure manuals. I detailed of the data collection procedures I used in the data collection section.

The case study research design requires documenting and following protocols to establish and maintain rigor (Cronin, 2014). As an example, researchers must conduct all interviews with the same introduction, the same general questions, and with the same tools (Cronin, 2014). However, exploratory studies also require improvisation in interviews, emphasizing data saturation and rich descriptions (Galster, Avgeriou, & Tofan, 2013). I followed a semistructured interview approach using the protocol (see Appendix A) detailed in the data collection technique section. I used the protocol in Appendix B to document field notes and observations during interviews with participating architects.

Participants

In qualitative research, researchers frequently seek participants for their ability to provide rich descriptions of the phenomenon (Draper, 2015; Petty, Thomson, & Stew, 2012; Wahyuni, 2012). The case for my exploratory case study was the EAT of my employer's enterprise. The criteria for participation included the following: (a) individuals that were fulltime employees of the EAT in roles of application, solution, or enterprise architect as defined above, (b) individuals that had been in architecture roles for a minimum of 5 years either in the EAT or at any organization, (c) individuals that worked on projects focused on the delivery of bespoke software, (d) individuals that worked or lived in the metropolitan area of Nashville, TN, and (e) individuals with whom I did not have a recurring working relationship. I designed these criteria to maximize the benefits of the research while minimizing risk to participants in the study.

Prior to any communication with potential participants in a study, researchers must gain Institutional Review Board (IRB) approval to ensure proper protections are in place for human participants, including the use of an informed consent process for potential participants (Faden et al., 2013; Lantos & Spertus, 2014; Ryan et al., 2014). I obtained IRB approval from Walden

University's Center for Research Quality (approval number 02-17-17-0539677) before contacting or recruiting individual study participants. After receiving IRB approval, I followed the desires of the vice president of technology – EAT at the case organization, which were to forego the use of a gatekeeper and manage participants myself. I used organizational information systems to identify potential participants, coordinated the informed consent process with potential participants, and coordinated interview scheduling with each participant. I sent potential participants a consent form and a copy of the signed letter of cooperation via email using the invitation to participate email template in Appendix C to gain access to those participants.

Establishing rapport with participants is critical for effective and efficient data collection in research based on a qualitative methodology (Shah, 2014). Developing the understanding that the researcher is a member of the participant's community, that is establishing a basis for empathy with the participant, is part of establishing rapport (Chou & Chiang, 2013; Shah, 2014) as is providing personal attention to the participant (Goode et al., 2015). I established rapport with participants in multiple ways. First, I learned about the culture of the case organization, adapting my attire and behavior to their standards. Second, I worked with each participant when selecting a site for interviews to ensure I met their privacy and comfort needs. Third, I explained to participants that I was familiar with the aspects of software development that we discussed though may ask for an explanation of any unfamiliar terms. Fourth, I ensured that participants knew that these sessions were about their experience and input, not mine.

Research Methodology and Design

While the terms method and methodology have been used interchangeably throughout the literature, there is a distinction between the two (Wahyuni, 2012). A method is a means of analyzing data, while a methodology provides a foundation on which researchers can choose an appropriate design based on their beliefs (Wahyuni, 2012). True methodologies include positivism, postpositivism, interpretivism, and pragmatism, and each has varying stances on ontology (nature of reality), epistemology (what constitutes acceptable knowledge), and axiology (the role of values; (Wahyuni, 2012).

Methodology

The actual nature of research methodologies influences their use in research. The more common descriptions of methodologies—quantitative and qualitative (McCusker & Gunaydin, 2015; Wahyuni, 2012; Wells, Kolek, Williams, & Saunders, 2015)—are related to true methodologies: positivism relates to quantitative, interpretivism relates to qualitative, postpositivism could relate to either, and pragmatism relates to both (i.e. mixed methodologies) (Wahyuni, 2012). Research designs tie the methodology to the methods, such that selecting one methodology precludes the use of designs and methods associated with another methodology (Wahyuni, 2012; Wells et al., 2015).

The more common understanding of methodologies also influences their use in research. The quantitative research methodology is about measurement, identifying and explaining relationships between aspects of a topic described in variables (De Massis & Kotlar, 2014; McCusker & Gunaydin, 2015; Wahyuni, 2012). Conversely, the qualitative research methodology is about exploration and creation of detailed understandings (De Massis & Kotlar, 2014; Wahyuni, 2012; Yin, 2014).

I explored the strategies used by architects in the pursuit of quality. The exploratory nature of this study warranted the use of a qualitative methodology. After collecting qualitative data using the techniques and procedures included in the data collection section, I interpreted the data in the context of the selected conceptual framework (TQM in this study) and case organization. Interpretive evaluation of data follows an interpretivist paradigm, which relates to qualitative research. The combination of the explorative nature of this research and the interpretivist nature of the data analysis dictated the use of a qualitative methodology.

Design

The common designs in qualitative applied research are phenomenology, ethnography, and case study (Petty et al., 2012; Vaismoradi, Turunen, & Bondas, 2013; Yilmaz, 2013). Phenomenological research has roots in psychology and philosophy with a goal of understanding the essence of a phenomenon (Kruth, 2014; Petty et al., 2012; Sloan & Bowe, 2014). Understanding the essence is achieved by exploring the experiences of individuals that have firsthand experience with the phenomenon (Kruth, 2014; Petty et al., 2012; Sloan & Bowe, 2014). The significance of phenomenology is that the researcher describes what the phenomenon under study is and how it is experienced (Sloan & Bowe, 2014).

Ethnographic research is focused on understanding behaviors, languages, and beliefs of people in a cultural group (Draper, 2015; Kruth, 2014; Petty et al., 2012). With roots in anthropology, ethnography requires a significant amount of time where the researcher is immersed in the daily activities of the people under study (Draper, 2015; Kruth, 2014; Petty et al., 2012). This firsthand exposure is the basis of the rich descriptions required of qualitative research (Vaismoradi et al., 2013; Wahyuni, 2012; Yilmaz, 2013).

Case study research is focused on describing the complexity of the phenomenon of interest (Hyett et al., 2014; Petty et al., 2012; Wahyuni, 2012; Yin, 1981). The phenomenon under study is a real-world bounded system, or a case, and researchers study it in its natural environment (Hyett et al., 2014; Petty et al., 2012; Wahyuni, 2012). A more comprehensive description of the phenomenon can be established through the inclusion of multiple sites—known as a collective case study—and multiple sources of data in the case study analysis (Hyett et al., 2014; Wahyuni, 2012). The primary differentiator for case study research from other types of research is that the bounded system, or a case that is easy to distinguish from other cases (Kruth, 2014), becomes the focus of the research instead of the participants (Stake, 1978).

I explored strategies used by architects in the EAT team of my employer for applying architectural best practices to maximize bespoke software quality. The focus was not the architects themselves or the behaviors of all architects, making ethnography a poor fit. Similarly, my research was not concerned with how architects experienced the strategies, just what strategies they used, making phenomenology a poor fit. With the focus of my research being the exploration of a phenomenon within a single case to develop an understanding of the case, I used an exploratory single-case study design.

One critical aspect of rigorous case study research is data saturation. Data saturation is achieved by spending sufficient time in the field collecting data to the point where the researcher identifies no new data (Houghton, Casey, Shaw, & Murphy, 2013) or where any new data does not continue to inform the research question (Gentles et al., 2015; Kruth, 2014). Effective sampling methods are a means of achieving saturation (Petty et al., 2012). I achieved data saturation in a few ways. First, I gathered data from semistructured interviews and organizational documents. Second, I used member checking to ensure accurate and complete interpretation of the data. Third, I employed census sampling with criteria that defined the population best suited to inform my research. Fourth, I employed methodological triangulation (discussed in the Data Analysis section).

Population and Sampling

The population for this research was application and enterprise architects that were fulltime employees of the case organization, were involved with maximizing the quality of bespoke software solutions, had worked in the industry as an architect for at least 5 years, lived or worked in the Nashville TN metropolitan area, and did not have a recurring working relationship with me. I applied these criteria as they provided the greatest ability to produce detailed and valuable information on the phenomenon while minimizing potential risk and bias. The case organization had 10 architects that met these criteria. Purposeful sampling depends on selecting sources of information that can provide rich, in-depth details of the phenomenon under study (Gentles et al., 2015; Petty et al., 2012; Wahyuni, 2012; Yilmaz, 2013). Based on the small population size of the study and the criteria used to define the population, I included the entire population as potential participants in my study.

Researchers achieve saturation by spending a significant amount of time in the field and sampling enough data that no new information emerges from further sources of data (Gentles et al., 2015; Houghton et al., 2013; Kruth, 2014). Methodological triangulation is the use of a multistep process when analyzing data (Wahyuni, 2012) and helps to achieve saturation (Houghton et al., 2013; Yilmaz, 2013). Member checking is also a means of achieving saturation, giving participants the ability to read the researcher's interpretations and provide any corrections or additional information (De Massis & Kotlar, 2014; Petty et al., 2012; Yilmaz, 2013). I gathered multiple sources of data including participant interviews and organizational documents focused on strategies to maximize the quality of associated products to achieve saturation. I employed methodological triangulation by using a research database, a transcription service to transcribe audio data into text data, and thematic analysis for coding the data. I also implemented member checking through followup sessions with interviewees to review my interpretations and ensure saturation.

While performing face-to-face interviews, the location of the interview can have a significant influence on the interview (Dempsey, Dowling, Larkin, & Murphy, 2016; Gagnon, Jacob, & McCabe, 2015; Rimando et al., 2015). The interview location should be convenient, comfortable, and provide a sense of safety/privacy for open conversations from the participant (Dempsey et al., 2016; Gagnon et al., 2015; Rimando et al., 2015). I worked with participants to identify sufficient meeting locations to suit their comfort, convenience, and privacy.

Ethical Research

There are three primary areas of ethical consideration in the Belmont Report: respect for persons, beneficence, and justice (Ryan et al., 2014). Beneficence is concerned with ensuring the benefits of the research are balanced with the risks (Faden et al., 2013; Ryan et al., 2014) and justice is ensuring that one group doesn't benefit from risks experienced by a different group (Faden et al., 2013; Ryan et al., 2014). Respect for persons is concerned with allowing the individual to control the parameters of their participation which researchers address through an informed consent process (Lantos & Spertus, 2014; Ryan et al., 2014; Yilmaz, 2013). Every research organization must have an IRB that validates all research meets or exceeds ethical standards before executing the planned study (Faden et al., 2013; Lantos & Spertus, 2014; Ryan et al., 2014). The risk associated with participation in this research was akin to the risk encountered in everyday life and the participants in this study were part of the group that would realize the benefits of this study, addressing beneficence and justice. I addressed respect for persons with an informed consent process and allowed individual participants to end their participation at any time without any consequences. I maximized respect for persons and minimized the potential for coercion since I had no supervisory or subordinate relationship with any of the potential participants. I also minimized the possibility of coercion by excluding those potential participants that I had a regular working relationship with through the criteria for participation.

Before any communication with potential participants in the study, I obtained IRB approval (approval number 02-17-17-0539677) from Walden University. Once I received IRB approval, I used information systems at the case organization to establish a list of potential participants that met the defined criteria. Once I had a list of potential participants, I sent a consent form and a copy of the signed letter of cooperation to those potential participants using the invitation to participate email template in Appendix C and made myself available to answer any questions any potential participants had.

The consent form included details of participant criteria, consent, withdrawal, incentives for participation, data retention and protection policies, and individual identity protection. Participants had the opportunity to withdraw at any time and for any reason during the study without adverse effect or consequence. I asked that potential participants contact me either by email address or mobile phone number to let me know of their decision. I ensured that potential participants understood that they could withdraw by communicating with the researcher or by contacting Walden University's research participant advocate. The consent form included a statement indicating that there was no compensation or any other incentive available for participation aside from the altruistic benefits that they may realize.

I will retain all data obtained for this research in a locked safe, either in print form or digital form based on the nature of the data, for five years from the date of final research approval, where it will only be accessible to me. As noted below in the data collection section, I recorded participant IDs on all collected data and the only place I associated individual names with their respective participant IDs was in a spreadsheet that was stored electronically with the study data. I password protected and encrypted this spreadsheet in addition to the encryption applied to the remainder of the research data.

Data Collection

Collecting data for this study comprised the instruments that I used, the technique used to collect the data, and the technique used to keep the data organized.

Instruments

The case study design and other qualitative research methods rely on the researcher as the primary data collection instrument and the use of well-defined protocols to guide research activities (De Massis & Kotlar, 2014; Petty et al., 2012; Wahyuni, 2012). In addition to actual data collected from participants, it is valuable for researchers to document their observations. Researchers use field notes to capture contextual information about the interview (De Massis & Kotlar, 2014; Petty et al., 2012; Wahyuni, 2012) and reflexive journals to explain the rationale for decisions, thoughts, and challenges the researcher experiences during the study (Houghton et al., 2013; Sloan & Bowe, 2014; Vaismoradi et al., 2013). As noted in the Role of the Researcher section, I was the primary data collection instrument.

I collected data via semistructured interviews using the protocol presented in Appendix A to guide the interview. I asked participants to bring any available documents, multimedia sources, or historical documents that supported the idea of maximizing quality with them to the interview. I also worked with non-participant members of the case organization to collect policy and procedure manuals and other organizational documents that explained quality procedures. During interviews, I noted any observations I had about the participant, the environment, or the general context as field notes using the protocol in Appendix B. I kept a reflexive journal throughout my research, taking note of the basis of decisions and thoughts and perceptions of the study in general.

Researchers use member checking, transcript review, and triangulation to maximize research reliability and validity (De Massis & Kotlar, 2014; Houghton et al., 2013; Petty et al., 2012; Yilmaz, 2013). Member checking is a means of achieving saturation by giving participants the ability to read the researcher's interpretations and provide corrections or additional information (De Massis & Kotlar, 2014; Petty et al., 2012; Yilmaz, 2013). Triangulation is the use of several sources of data to study a phenomenon (Houghton et al., 2013; Hyett et al., 2014; Wahyuni, 2012). I iteratively used member checking (interviews and followup sessions with interviewees to review my interpretations) until participant responses provided no new data, data triangulation (use of interviews and organizational documents), and methodological triangulation (study database, interview transcription, and coding) to maximize reliability and validity in my study.

Data Collection Technique

After IRB approval and organizational agreement to participate, I used organizational information systems to establish a list of potential participant names and contact information. I met with each potential participant to determine interest in participation. When a potential participant indicated an interest in participating, I confirmed consent, answered any questions, and established times and locations for data collection. For document analysis, I collaborated with senior leaders of the EAT organization to collect organizational documents that defined the culture of the EAT organization related to quality.

For participant interviews, I sent a consent form and a copy of the signed letter of cooperation to potential participants. Three individual potential participants decided not to participate; I noted that in the Excel spreadsheet and moved on to the next potential participant. When individual potential participants agreed to participate by signing the consent form, I scheduled a time and a location with them and conducted the semistructured interviews. I conducted on-site interviews at a location agreed to by the participant and followed the protocol defined in Appendix A. I recorded the sessions using a portable audio recording device for subsequent transcription and evaluation.

Member checking sessions provide participants with the ability to read the researcher's interpretations and comment with any corrections or additional information (De Massis & Kotlar, 2014; Petty et al., 2012; Yilmaz, 2013). At the conclusion of interviews, I scheduled a followup session with each participant for member checking. I took copies of the transcription and evaluation report to the followup interview and reviewed the evaluation report with the participant. I gave each participant the opportunity to confirm, elaborate, or correct the information they provided as they felt necessary. I recorded the audio of those followup sessions and kept a copy of the updated transcriptions with the research data following the data retention policy described in the consent form.

Data Organization Techniques

Using a research database improves the reliability of the research (Aleti et al., 2013; De Massis & Kotlar, 2014; Petty et al., 2012) by providing a chain of evidence, logging where and when I collected data (Cronin, 2014). I cataloged all data collected as part of the case study in a computer-aided qualitative data analysis software tool (CAQDAS) named NVIVO 11 and—in the case of physical artifacts—lockable file storage for review and retrieval; I will retain this data in that lockable storage for five years. I indicated the date and participant ID that provided each data source in the database and on physical instances of collected data, establishing a chain of evidence. I cataloged all forms of data in this database including interview audio and transcriptions, collected documents (all in digital form), field notes (digitized copies of physical forms, which were destroyed after digitizing), and reflexive journals.

Data Analysis Techniques

Of the two primary modalities used in research, inductive and deductive, inductive approaches rely on the collected data to provide themes and deductive approaches compare categories in one study with those in a separate study (Kruth, 2014; Vaismoradi et al., 2013; Yilmaz, 2013). Qualitative studies typically use inductive approaches (De Massis & Kotlar, 2014; Kruth, 2014; Yilmaz, 2013). Triangulation is the use of multiple sources of data, adding credibility to inductive approaches and case study research (Cronin, 2014; De Massis & Kotlar, 2014; Houghton et al., 2013; Wahyuni, 2012).

There are four primary types of triangulation in qualitative research: data, investigator, theory, and methodological (Fusch & Ness, 2015; Houghton et al., 2013; Wilson, 2014). Investigator triangulation is applicable when more than one researcher participates in the study, and theory triangulation applies when multiple theoretical strategies are employed (Fusch & Ness, 2015; Wahyuni, 2012; Wilson, 2014). Researchers achieve data triangulation by gathering multiple sources of data from more than one person or time (Houghton et al., 2013; Wahyuni, 2012; Wilson, 2014) and methodological triangulation is the use of multiple methods to analyze and correlate data collected from multiple sources (Fusch & Ness, 2015; Houghton et al., 2013; Wahyuni, 2012). The methods used in methodological triangulation include data storage methods, transcribing audio sources and verifying the transcriptions, and removing personal or organizational identifiers from the data (Wahyuni, 2012).

Content and thematic analysis are two common approaches for inductive analysis of qualitative data (Petty et al., 2012; Vaismoradi et al., 2013; Wahyuni, 2012). Content analysis is the identification of categories of information within the data and the categories are used to quantify the phenomenon (McCusker & Gunaydin, 2015; Vaismoradi et al., 2013; Wahyuni, 2012). Thematic analysis is the inductive process of identifying codes within the qualitative data which researchers group into larger themes that describe the phenomenon (Petty et al., 2012; Vaismoradi et al., 2013; Wahyuni, 2012). The primary distinction between content analysis and thematic analysis is quantitative frequency measuring of categories in content analysis is used as a possible measure of significance (Vaismoradi et al., 2013).

I used both data and methodological triangulation in my study. Investigator triangulation was not applicable as I was the only investigator in my research and theory triangulation did not apply as I used one conceptual framework in my research. I achieved data triangulation by collecting interview and organizational document data. Regarding methodological triangulation, the method I used to analyze all gathered data began with collecting and organizing the data as noted in the preceding sections.

I used a third-party transcription service (Rev.com) to transcribe the audio-based recordings of interviews and member-checking sessions and validated the transcriptions using transcription software (InqScribe). I had Rev.com complete the enclosed confidentiality agreement (see Appendix D) to ensure privacy and confidentiality of participant information. I loaded all data—transcriptions, documents, field notes, reflective journals—into NVivo for coding and thematic analysis (I did not use content analysis). After reading through the collected data a few times for familiarity, I began the coding process. I based coding on words and phrases related to management of quality in software solutions (e.g. usability, satisfaction, requirements, design, and performance) and recorded the words and phrases that I grouped into those codes for future reference and verification. Once I defined the codes, I analyzed and evaluated those codes and identified themes.

Throughout my study, I monitored the literature, results of member checking, and new interviews for new information on my topic. No new academic research on the topic of strategies used by enterprise and application architects in the pursuit of quality became available, and I identified four new peer-reviewed articles on the topic. While no new information contradicted data collected in interviews, I did request further member checking sessions to validate whether the identified themes varied by organization or some other factor. All participants indicated that the noted strategies were consistent across all organizations.

Reliability and Validity

Validity is a measure of how well the research studies what was intended (Kruth, 2014) and reliability is a measure of how well the study could be repeated with relatively the same outcome (Petty et al., 2012). Validity is classified as construct validity, external validity, internal validity, and reliability for most research designs, though internal validity is not applicable to exploratory studies (Galster et al., 2013; Holweg & Helo, 2014; Martini et al., 2015). Instead of the four types of validity, qualitative research measures rigor based on credibility, dependability, confirmability, and transferability (Houghton et al., 2013; Petty et al., 2012; Wahyuni, 2012). Credibility is analogous to internal validity, dependability is analogous to reliability, confirmability is analogous to objectivity, and transferability is analogous to external validity (Yilmaz, 2013).

Credibility

Credibility is an indicator of how well the collected data represents the studied phenomenon (Wahyuni, 2012) and how accurate or believable the participants in the study find the results (Houghton et al., 2013; Yilmaz, 2013). Qualitative research that includes triangulation, member checking, and prolonged engagement to achieve saturation establishes credibility (Houghton et al., 2013; Petty et al., 2012; Yilmaz, 2013). As noted in the data collection section, I used member checking as one means of establishing credibility. I also used data triangulation through the collection of interview and organizational document data and methodological triangulation with a multi-step process for organizing and analyzing the data. I addressed credibility by requesting that participants not discuss any aspect of their participation with others until I concluded the research.

Data saturation is achieved by spending sufficient time in the field collecting data to the point where the researcher identifies no new information (Houghton et al., 2013) or where any new data does not continue to inform the research question (Gentles et al., 2015; Kruth, 2014). Effective sampling methods are a means of achieving saturation (Petty et al., 2012). I achieved data saturation in a few ways. First, I gathered data from semistructured interviews and organizational documents. Second, I used member checking to ensure accurate and complete interpretation of the data. Third, I employed census sampling. Fourth, I employed methodological triangulation (discussed in the Data Analysis section).

Dependability

Qualitative studies that employ audit trails, or chains of evidence, and reflexive journals achieve dependability (Houghton et al., 2013; Wahyuni, 2012; Yilmaz, 2013). As noted in the data collection and data organization sections above, I addressed dependability by including a reflexive journal with the remainder of my study data. I also addressed dependability by establishing the chain of evidence, recording when and from whom (using the Participant ID) I collected data in my study database.

Confirmability

Confirmability is an indicator of the accuracy and neutrality of data collected for the research (Houghton et al., 2013) and the objectivity of the researcher while working with it (Petty et al., 2012). In simpler terms, it is an indicator of how effectively other researchers can confirm the conclusions (Wahyuni, 2012). Studies that include audit trails, or chains of evidence, and reflexive journals meet the needs of confirmability (Houghton et al., 2013; Petty et al., 2012; Wahyuni, 2012; Yilmaz, 2013). As noted in the data collection and data organization sections above, I recorded reflexive journals and stored them with the remainder of my study data. I also recorded when and from whom I collected data, creating a chain of evidence.

Transferability

Transferability is an indicator of whether or not conclusions can be applied to other contexts or situations (Houghton et al., 2013; Wahyuni, 2012; Yilmaz, 2013). Qualitative research that establishes thick or rich descriptions of the setting, context, and participants establish transferability (Houghton et al., 2013; Petty et al., 2012; Yilmaz, 2013). To determine the transferability of results, I documented the details of the context and case in field notes as well as the case report and final study report.

Transition and Summary

In section two, I provided details of the project, indicating that the purpose was to explore how software and enterprise architects influence the quality of bespoke software solutions. I performed all data collection, acting as the primary data collection instrument. I followed all guidelines regarding the ethical treatment of participants outlined in the Belmont Report including informed consent, though I targeted no protected groups as participants; architects working on bespoke solutions were the intended participants. I used an exploratory qualitative case study with the EAT of my employer as the case and used census sampling to select participants to achieve data saturation. I collected data from interviews and organizational documents and used NVivo as my research database to organize and analyze all collected data and to keep a log of all decisions made during my research. I used both data and methodological triangulation across multiple data sources to ensure saturation and completeness. I addressed validity and reliability through member checking, using reflexive journals, keeping an audit trail, and including details of my experiences with the topic and study participants. I recorded the details of the study as executed in section three, including conclusions and recommendations for future research.

Section 3: Application to Professional Practice and Implications for Change

Overview of Study

The purpose of this qualitative exploratory case study was to explore strategies used by software and enterprise architects for applying architectural best practices to improve bespoke software quality, lowering the total cost of ownership. I collected data from the enterprise architecture team at a large enterprise in the Nashville Tennessee metropolitan area in the United States, interviewing and performing member-checking sessions with seven enterprise or solution architects and collecting 47 organizational documents. The architects I interviewed worked in three distinct capacities (e.g. application, cloud, business, data, security, and governance) and ranged in seniority (organizational chart, not tenure) from junior to senior on a 3-point scale. All participants had between 5 and 15 years of architecture experience and most had between 6 and 15 years of software engineering experience. Two participants had less than 5 years of experience as a software engineer.

The gender distribution of the participants was nonnormal, obviating the potential for gender-based analysis and the seniority apportionment of the participants was roughly normal. I categorized participants into two groups by years of experience as an architect, with five participants having between 5 and 10 years of experience and two participants having between 11 and 15 years of experience. I also categorized participants into three groups based on the number of years of experience as a software engineer. Two participants had between 1 and 5 years of experience, three participants had between 6 and 10 years of experience, and two participants had 11 or more years of experience. My analysis of the data resulted in five strategies for achieving the quality of bespoke solutions and indicated some variance in these strategies by context.

Presentation of the Findings

The research question I sought to address was the following: What strategies do application and enterprise architects use to apply architectural best practices to improve bespoke software quality?

It is significant to note that the participants had widely varying thoughts on architectural best practices. Three participants indicated that appropriate architectural best practices vary by solution and team maturity while four participants indicated that these best practices serve only as guidelines rather than blueprints for success. The perception of best practices as guidelines influenced the results as the gathered data indicated the strategies application and enterprise architects use to improve bespoke software quality, not the strategies they used to apply architectural best practices.

Four of seven participants stated that the strategies they employ vary by context and the defining context itself varied between participants. Four participants indicated that the strategies vary based on the makeup of the development team (regarding team maturity and cultural makeup), three of the participants stated that organizational culture influences the strategies they employ, and two said that the newness of the technology or concept influences their strategies. While participants indicated that strategies vary by context, all seven indicated that these strategies apply to all organizations. I present the strategies in order from the most to the least prevalent.

Focus on Customer Satisfaction

Focusing on customer satisfaction was one of the prominent themes. The concept was that architects should focus their activities in the process of developing software on delivering solutions to the customer's needs. Activities range from participating in business requirements definition and in buy-versus-build conversations to ensuring that solutions are available and usable and that they foster positive user experiences. These activities are critical as delivering bespoke software solutions is complex. Sometimes architects and software engineers create architectures and designs with quality in mind, yet issues arise during development that require a variance from the architecture or design. In those situations, decisions need to focus on delivering solutions to meet customer needs and minimizing solution lifecycle costs.

There are a few ways that architects can minimize lifecycle costs. Delivering solutions quickly, minimizing solution complexity, and simplifying issue resolution are a few cost-saving measures. Assigning responsibilities to individual solution components and aligning overall solution capabilities to those responsibilities is one means of maximizing maintainability. Doing so helps minimize costs associated with testing, maintaining, and extending overall solution capabilities. Deciding when to accept technical debt based on risk is one means of delivering solutions quickly, though accepting technical debt does require tracking all assumed technical debt so that teams can address it before it causes increased costs itself.

All seven participants indicated that focusing on achieving customer satisfaction is central to improving quality and 32 of 47 organizational documents supported the theme (see Table 2 for information source metrics). Six of the seven participants indicated that delivering solutions to customer's needs is part of focusing on customer satisfaction, mentioning that they participate in and sometimes drive buy-versus-build conversations, business requirement definition, and understanding the needs of all stakeholders. Twenty organizational documents supported those ideas and showed that architects should drive positive user experiences, the availability of solutions when users need them, and maximize the usability of solutions.

Table 2

Minor Themes of Focus on Customer Satisfaction with Supporting Metrics

	Participant		Document	
Major/Minor Theme	Count	References	Count	References
Focus on customer satisfaction	7	41	32	95
Deliver solutions to their needs	6	12	20	55
Drive toward minimizing solution lifecycle costs	6	19	24	43
Perform trade-off analysis	5	8	20	65

Similarly, six of seven participants indicated that working to minimize lifecycle costs is part of focusing on customer satisfaction. They stated that delivering and fixing issues quickly, including support options that COTS solutions include, and determining when to delay the implementation of specific architectural or design components based on risk analysis are all means of minimizing costs. Twenty-four organizational documents supported these ideas by recommending that architects follow best practices, drive data and solution component efficiency, track technical debt with business impact in product backlogs, and discuss the technical debt with product owners to prioritize its resolution.

Focusing on customer satisfaction aligns with the selected conceptual framework, as the forefathers of TQM define quality as customer satisfaction and agreed that it is a complex topic (Crosby, 1992; Deming, 1982; Feigenbaum, 1985; Ishikawa, 1984; Juran & Godfrey, 1999; Shewhart, 1939). The forefathers found that customer satisfaction comprises psychological impressions, fitness for use, meeting specifications, and the perception of value in the solution (Deming, 1985; Feigenbaum, 1985; Ishikawa, 1984; Juran & Godfrey, 1999; Shewhart, 1939). One of the participants summarized the feeling well, stating "And one other thing that we started to focus…on is mostly about the user experience, customer experience and customer journeys." Authors in the literature agreed that having a customer focus improves product quality. Shah

(2014) showed that having a customer focus is critical to testing software solutions while Cronemyr and Danielsson (2013) and Tunkelo, Hameri, and Pigneur (2013) indicated that customer focus is essential to process management and improvement. The data and literature both supported multiple aspects of focusing on customer satisfaction.

The collected data supported the TQM component of quality known as psychological impressions, or how users feel about using bespoke solutions. One participant mentioned that users consider reliability and consistent user experience when determining if they want to continue using solutions, adding that architects should maximize those attributes. In the literature, Goode, Lin, Tsai, and Jiang (2015) cited studies that linked customer focus to perceptions of security, aligning with both the user's psychological impressions and a responsibility that architects typically address (security). Another participant supported the idea of architects addressing psychological impressions by stating that architects focus, in part, on improving user experience, though a third participant noted that architects take on this role only in lieu of a dedicated person filling a user experience role on the team. Organizational documents indicated that user interface (UI) or user experience (UX) architects in the organization have the responsibility of ensuring solutions result in positive user experiences. Authors of these organizational documents indicated that developers could address usability through aspects of solutions including data validation, user feedback in the form of progress bars and other visual indicators, being as intuitive as is possible, providing self-help tools, and ensuring a consistent experience across solutions.

Additionally, the data supported the TQM concept of fitness for use, or the user's ability to achieve required outcomes with the solution. Two participants mentioned solutions must be usable, indicating that usability includes the concepts of whether the system performs the functions required by the user and that it is available when the user needs it. In the literature, Zhou, Ji, and Jiao (2013) noted that user-centered design was one of the classic design paradigms for maximizing the usability of solutions. A third participant supported alignment with fitness for use, stating, “Quality to me is that you are delivering software that meets the…product market fit may be the best way to say it. It is usable; it has value.” In the literature. Belk, Papatheocharous, Germanakos, and Samaras (2013) and Göransson et al. (2003) suggested that following a user-centric interface design improves solution usability. Organizational documents including applied architectural principles and architectural toolkits supported the ideas of availability and usability, denoting data availability standards, customer experience standards, and product evaluation matrices.

The collected data also supported the TQM concept of meeting specifications through product design. All seven of the participants indicated that they participate in solution requirements engineering in some way. In the literature, Kassab et al. (2014) cite a study that indicates teams that invested significant effort in requirements definition and architectural definition at the beginning of the project realized an average of 92% cost savings over those that invested minimal effort at the beginning. Two participants mentioned validating requirements, including addressing requirements conveyed as solutions as opposed to actual business needs. A third participant, who made the following statement, addressed the need for architects to validate requirements that express solutions instead of needs:

> Back in the 80s, technology was a black box and business people trust[ed] it as right and they would give us legitimate business requirements. Now, they have got[sic] some technology. Everybody has got[sic] a working understanding of technology, so they want

> to be technologists. They do not give business requirements anymore. They give technical requirements.

Another participant mentioned that architects are involved in ensuring everyone has a clear understanding of the requirements and a fifth participant indicated that architects must validate both functional and nonfunctional requirements. In the literature, Cleland-Huang (2014) supported this, noting that focusing on functional requirements and ignoring NFRs and other quality aspects lead to failed bespoke software solutions. The authors of organizational documents collected as part of this research, such as architect role profiles, training documents, and engagement documents, indicated that business and other architects are responsible for capturing business requirements and monitoring business activities to identify needs that IT solutions might support.

Participants and organizational documents also supported the TQM concepts of value and supportability. One participant summarized the importance of architects, stating that enterprise architects focus on how organizational information systems work together to deliver business value. Authors of organizational documents focused on defining and measuring architectural applied principles specifically noted how each principle aligns with related business value. Aside from improving the quality of solutions, customer-centricity has other benefits as well. Three participants noted that total cost of ownership, including operational and maintenance costs of solutions, increases as the need for manual processing and additional development increases, indicating that greater automation in solutions is one means of reducing operational costs and increasing value. Authors of organizational training materials for architects emphasized the need to maximize satisfaction by delivering solutions that add value beyond the immediate need. In the literature, Avgeriou, Kruchten, Nord, Ozkaya, and Seaman (2016) discussed the concepts that comprise total cost of ownership (TCO) while Aviles (2015) and Huckabee (2015) discussed the idea that low TCO represents business value, aligning with the collected data.

One participant summarized the change in value proposition when organizational leaders require that employees use bespoke solutions, indicating that the architect must work to ensure the solution delivers value as there is no other incentive to deliver truly valuable solutions. In the literature, Pass and Ronen (2014) agreed, mentioning that software solutions that are mandated by regulation are immune from cost considerations as they are assigned infinite value. Similar to other aspects of focusing on customer satisfaction, authors of organizational documents such as architectural applied principles, operations guides, and presentations regarding the purpose of the enterprise architecture team included several references to the financial performance of bespoke solutions through attention to return on investment and return on equity. Aviles (2015), in the literature, stated that customer focus is one of three enablers of collaborative relationships, aligning well with other findings in my research.

Collaborate and Communicate with All Stakeholders

Collaborating and communicating effectively with all stakeholders was another prominent theme. The essence of this theme was that architects participate in conversations with both technical and nontechnical stakeholders and must be able to do so effectively. To be effective at these communications, architects must exercise humility and actively listen to all interested parties, yet act as a mentor when necessary or beneficial. Architects should continue this communication throughout the life of the product and participate in defining a common lexicon to reduce confusion and maximize the effectiveness of communications. Additionally, architects should work with stakeholders to define KPIs and SLAs to facilitate conversations and communications.

All seven participants indicated that collaborating and efficiently communicating with stakeholders is central to improving quality (see Table 3 for information source metrics). Six of the participants stated that architects should work to keep stakeholders happy by not considering themselves experts, actively learning from both technical and nontechnical stakeholders at all levels of the organization. Five of the participants indicated that architects should continuously involve themselves in the solution, not only establishing credibility by interacting and mentoring in real time but also by periodically monitoring the development team's coding and progress to ensure teams meet defined standards. Four of the participants discussed effective communications, both verbal and written, by including appropriate diagrams and using a common vocabulary based on the type of stakeholder with whom they are communicating. Two participants indicated that defined KPIs or SLAs are a basis for establishing the importance or significance of tasks in meetings.

Table 3

Minor Themes of Collaborate with All Stakeholders with Supporting Metrics

	Participant		Document	
Major/Minor Theme	Count	References	Count	References
Collaborate and communicate with all stakeholders	7	37	24	79
Exercise humility	6	12	4	8
Be continuously involved in the solution	5	11	8	16
Effective communication with common vocabulary	4	6	21	43
Define and drive to KPIs[a] or SLAs[b]	2	2	3	3

Note: [a]KPIs are key performance indicators. [b]SLAs are service level agreements.

Additionally, authors of 24 organizational documents supported the theme (see Table 3 for information source metrics). Authors of training and policy manuals indicated that architects should seek guidance from domain and application experts (i.e. other architects) for advice and validation while also seeking the viewpoints of others and putting aside personal needs. In role profiles and policy manuals, authors mentioned that architects should participate in design reviews, create reference implementations, coach and mentor others when needed, and be one of the first technical resources to be involved in projects. Authors of twenty-one organizational documents of various types supported effective communication and common vocabulary either by prescribing it as a tactic or by defining terms to facilitate communication. In three of the organizational documents, authors provided a means of measuring KPIs or mentioned that architects should focus on meeting KPIs.

Collaborating with all stakeholders aligns with TQM, but not completely as defined by research participants at the case organization. Many of the forefathers of TQM believed that involving everyone in quality is critical to achieving quality (Deming, 1982; Feigenbaum, 1985; Ishikawa, 1984). One research participant explicitly linked this to architects, stating that they "…think an architect wants to work with everyone on the team and understand where their strong and their weak points are and encourage them to work together." Support for this idea also exists in the literature, as Simon, Fischbach, and Schoder (2013) found that enterprise architecture teams must collaborate with all stakeholders to be effective. There are means of improving collaboration and communication.

Having a clear vocabulary is one means of ensuring effective communications. In TQM terms, Juran (1999) believed that ongoing communications were critical to quality as operational needs constantly change, also believing that common vocabulary is crucial. From the literature, Gürses, Seguran, and Zannone (2013) noted that using a controlled vocabulary is critical to requirements engineering to limit ambiguity. One participant indicated the importance of having a common vocabulary, stating, "…even in English, there are something like 2,000 different English-speaking communities that speak it differently, so [having a shared understanding of vocabulary] is extremely important." Several other authors in the literature supported the idea.

Richardson et al. (2012) believed that team communications must use a clearly-defined, common vocabulary. Another participant expressed the complexity involved in integrating systems, each with their own lexicon, stating “It’s like speaking two different languages.” In some cases, teams might address this on their own, naturally. Alfaro and Chandrasekaran (2015) noted that teams tend to develop and communicate with a specific vocabulary. This natural creation of a common language leads towards involving everyone in quality,

All stakeholders must be involved to achieve customer satisfaction. In TQM, Feigenbaum (1985) introduced the idea that quality is everyone’s responsibility and Ishikawa (1984) added that commitment to quality must exist throughout the organization. Spinellis (2014a), from the literature, indicated that collaborating with all stakeholders was a significant part of one project’s success. One participant stated that knowledge could come from anyone, pointing out that junior people on the team may have new ideas that lead to innovation and senior people on the team have detailed knowledge of the business and existing solutions, supporting the importance of collaboration. Deming (1982) believed that collaboration and communication were so critical to quality that organizations should build cultures that involve everyone in quality objectives.

TQM implementations rely on continuous improvement, which requires the use of baselining, to drive decisions. Juran (1999) believed that establishing units of measure, much like KPIs and SLAs, improves communication as they are the basis of continuous improvement, adding that quality control councils are responsible for defining and monitoring the units of measure. One participant indicated that they included the impact of KPIs on their architectures in communication and collaboration. In the literature, Tunkelo, Hameri, and Pigneur (2013), Huang, Wu, and Chen (2013), and Spinellis (2014a) all discussed the importance of KPIs in facilitating communications. Deming (1982), conversely, believed that measuring through metrics like KPIs is valuable for management and reporting but had no direct influence on quality.

While there is alignment between the findings in this research and TQM, it may be appropriate to use an alternate or multiple frameworks to evaluate portions of this data. I found no mention of exercising humility from any of the forefathers of TQM, though it could be inherent in the paradigm of involving everyone. This is significant as six of the participants alluded to exercising humility in their interviews and, in the literature, Li, Ko, and Zhu (2015) mentioned that—in addition to technical aptitude—effective development staff members have personality traits that include being humble. French and Raven (1959) developed a theory titled the bases of social power in 1959. This theory proposes five bases of power: reward power (incentive-based power), coercive power (punishment-based power), legitimate power (prescription-based power), referent power (personal-association-based power), and expert power (expertise-based power). Appearances are that exercising humility would be akin to using referent power to have stakeholders desire to conform as opposed to other types of power in this theory, though a detailed understanding of bases of social power is required to reach that conclusion.

Define Boundaries and Empower People within Them

The define boundaries and empower people within them theme has a dual meaning. The first meaning is clearly defined team member roles and responsibilities support empowering people within their roles. While defining the roles and responsibilities is a management function, architects can empower other team members within those definitions and team members can empower architects within their defined responsibilities. How stringently architects follow management-defined boundaries depends on the technical maturity of the team, the newness of the technology or concept, and the organizational culture.

This theme also means that architects should clearly define the boundaries of components in a solution and empower software engineers within those boundaries. Architects define these boundaries within the context of the holistic solution which they have bounded by aligning the solution with corporate strategy. Within the defined boundaries, architects empower teammates by only escalating variances from the defined boundaries when the variance creates moderate to high risk of the solution not resulting in customer satisfaction.

All seven participants indicated that defining and working within boundaries is one means of maximizing product quality (see Table 4 for information source metrics). The predominant means of achieving quality, as mentioned by all seven participants, is to clearly define the boundaries and empower people to take ownership within those boundaries, adding that holding them accountable for their portion is part of empowerment. Three participants indicated that architects should not confuse empowerment with autonomy, yet the importance of followups and review sessions varies by the maturity of the team. One participant referred to clearly defined job responsibilities and how that allowed them to work towards quality.

Table 4

Minor Themes of Define Boundaries and Empower People within Them with Supporting Metrics

	Participant		Document	
Major/Minor Theme	Count	References	Count	References
Define boundaries and empower people within them	7	28	15	36
Clearly define the boundaries	7	15	7	9
Escalate only when risk dictates	5	10	1	1
Align solutions with corporate strategy	4	11	13	36

Four of the participants (one senior and three intermediate in the organizational hierarchy) indicated that aligning solutions to corporate strategy is a critical part of their role and significant to the perception of quality. Regarding escalation, one participant mentioned that it should only occur when there is a clear misunderstanding between business need and technical direction. Four of the remaining participants agreed that escalation should only happen when associated risks are high enough. The common preference amongst five of the participants was to use persuasion over enforcement, supporting the idea of future research based on French and Raven's (1959) theory titled the bases of social power.

Additionally, authors of 15 organizational documents supported the theme (see Table 4 for information source metrics). Authors in seven documents supported defining clear boundaries of which four describe different roles of architects along with the technical and expected colloquial soft skills. The remaining three documents either were standards used in the tools for evaluating architectures or were the tools for evaluating and presenting architectures. The author of one organizational document discussed when escalation should occur, primarily focusing on changes in architectural scope and the risk associated with that change.

In total, authors of 13 organizational documents discussed aligning solutions with corporate strategy with some overlap of the seven documents that define technical boundaries. Authors of these documents defined the standards and tools the architecture team uses to measure the alignment of solutions with strategy. Some examples are disaster recovery and business continuity standards, product roadmaps, and product evaluation matrices for measuring alignment with corporate goals.

Defining boundaries and empowering people aligns with TQM, but not completely as defined by research participants at the case organization. Juran (1999) noted that overall products and their components must be defined with clear functional boundaries and alignment to the organization's strategic goals, indicating that it is the team's responsibility to set these limits. Three research participants noted that architects should set technology, tool, middleware, and library boundaries and the engineering team should stay within those boundaries while delivering solutions. In the literature, Huckabee (2015) agreed, discussing how team members defined boundaries for solution components with use cases and user stories. Defining boundaries and aligning solutions to strategy are two parts of what make up an effective product or service design in TQM as defining these boundaries is what allows teams to build quality in from the beginning (Deming, 1982; Feigenbaum, 1985; Shewhart, 1939).

In the literature, Richardson et al. (2012) indicated that clearly defining and disseminating roles and responsibilities of all team members to all team members is a best practice for teams. Regarding team member roles and responsibilities in TQM, Juran (1999) noted that role boundaries would fluctuate over time, similar to what one research participant mentioned as the converging of application architect and software engineer. Another participant described the current distinction between the role of architect and the role of engineer by stating that, "if [the architect is doing it right], they are setting boundaries for [the developers], which they may or may not like, but [the developers] still have a lot of freedom to design." The sentiment supports what Brown (2014) mentioned in the literature, empowering teammates within specified boundaries. Hill (2008) and Bon and Mustafa (2013) noted that creating empowered employees is one of the primary objectives of TQM.

Authors in the literature and research participants also discuss escalations. Tunkelo (2013) mentioned that escalating issues to management was triggered based on an evaluation of three critical aspects of the phenomenon being monitored exceeding expected thresholds. Four research participants agreed with this concept, stating that escalations should only occur when risks are high enough. Shah (2014) cited several incidents in her case study where elevated risk based on specific KPIs resulted in the escalation of issues. The alignment between the findings in this research and TQM are akin to those of the theme involving collaboration and communication in that alternate or multiple frameworks may be appropriate. I found no mention of escalation from any of the forefathers of TQM, though French and Raven's (1959) theory titled the bases of social power may address the idea of escalation. My initial thoughts are that escalation would occur through a shift from legitimate power (where the influenced individual decides that they want to change) to a coercive power (where the threat of punishment forces a change in the person) or to an expert power (where one individual perceives the other as an expert). A detailed understanding of this theory is required to support evaluation.

Deliver Prototypes and Work Products

While it is important to exhibit soft skills and have a focus on quality, it is equally as important to deliver artifacts that help the development team meet those objectives. Artifacts include prototypes or proofs of concept, models to describe solutions, and a defined set of tools and technologies to implement the solution. Architects use models to support multiple views or perspectives of a single solution and as a basis for many solutions. Similarly, architects use prototypes to prove a set of technologies combine in a specific way to achieve a goal and to mentor team members on implementation techniques. The level of detail that architects include in prototypes varies. Detailed prototypes may not be necessary when the development team and the architect have a good working relationship and understanding of the components of the solution or when the team has a good knowledge of the included technology. Additionally, architects may not need to spend as much effort fostering a relationship of trust through delivering prototypes with the development team if the organization naturally empowers the architect.

Six of seven participants indicated that delivering prototypes and work products are critical to helping the team achieve quality (see Table 5 for information source metrics). Five of those participants believed that prototypes should focus on high-level interactions between components and prove that the technology concept works to support the solution. Three of the participants believed that architects should not use prototypes to prescribe the use of specific patterns unless they use the prototype to mentor the development team on implementation strategies or new technologies.

Table 5

Minor Themes of Deliver Prototypes and Work Products with Supporting Metrics

	Participant		Document	
Major/Minor Theme	Count	References	Count	References
Deliver prototypes and work products	6	19	12	24
Prove the concept, don't prescribe patterns	3	6	1	2
Focus on high-level interactions	2	5	2	2
Mentor team members on technical implementations	2	3	11	15
Recommend technologies	2	2	3	3

Authors of twelve organizational documents supported delivering prototypes and work products (see Table 5 for information source metrics). One document was a solution architecture document in which authors (i.e. architects) included a background of the solution, a diagram of current and proposed future states, data models, vocabularies, structural and deployment models, and how the solution adheres to previously defined architectural principles and statements of direction. Authors of other documents included the statements of direction and applied principles that architects defined. The author of one policy manual mentioned that architects should create reference implementations (i.e. prototypes) to aid the development team in the implementation.

Another organizational document is a technical quality assessment document, which architects used to verify that development teams implemented solutions in accordance with the defined architecture and that architects are tracking technical debt. Other document types included job role descriptions in which authors define deliverables for each type of architect in the organization, the work products and role of architectural governance teams, and competencies defined by organizational leaders for each type of architect.

The delivery of prototypes and models directly aligns with TQM. Deming (1982) and Juran (1999) noted that the use of prototypes is a common practice, with each component of the prototype being as similar to the production counterpart as is reasonably possible. Six of the seven research participants discussed using prototypes as blueprints for successful implementations. In the literature, Simon, Fischbach, and Schoder (2013) discussed how architecture-driven business models influenced their solution architectures and design processes. Prototypes have specific uses in product lifecycles and Aleti et al. (2013) proposed the use of architectural models when optimizing the quality of solution architectures.

When provided, architects used prototypes to indicate product direction, not fully represent the final delivered product. Deming (1982) noted that these prototypes will not always represent the outcome of the production runs due to required and unexpected variations during production. One research participant stated "…part of what the prototype does is give the person that I am handing this off to an assurance that it is gonna[sic] actually work when they put it together." Similarly, aligning with this from the literature, Göransson et al. (2003) noted the importance of using less formal models and prototypes to facilitate communication with stakeholders when following user-centric design.

Delivery of these prototypes must be timely to deliver the greatest value. In the literature, Cleland-Huang, Hanmer, Supakkul, and Mirakhorli (2013) mentioned that prototypes should be delivered early in the project to demonstrate the solution's ability to meet quality goals. From a TQM perspective, Juran (1999) pointed out that prototypes are used to test products in marketing and that any results from those tests should be passed on to production teams. One research participant indicated that they wrote prototypes before receiving full requirements specifications and before they defined other solution documents.

Aside from prototypes, architects must deliver other work products as well. From a TQM perspective, Juran (1999) promoted that the use of diagrams, schematics, and other specifications including the 4+1 model of architecture. In the literature, Despa (2015) discussed projects where teams improved quality by providing diagrams and quickly building prototypes to validate ideas with stakeholders, refine specifications, and validate assumptions. Four research participants discussed architect involvement in the definition and delivery of models (e.g. logical models and physical models) and diagrams, with models providing a foundation for dynamically creating different diagrams. On participant summarized the value of models over diagrams as models are interactive and based on "a repository of data that has been blessed, that there is control over…, that there is change management on."

Use Process as a Guideline

The last identified theme was the use of process as a guideline rather than a prescriptive model for daily tasks. One rationalization of this thought is that architecture tends to have no formal time allocation in agile methodologies which can lead to inconsistent architectures and lower product quality. The importance of process, however, varies by organizational culture. When time-to-market is not the dominant driver in the culture, teams may use a more methodical approach that includes detailed research before beginning development. Even in organizations that do favor quick delivery, having a process that includes periodic reviews is essential to quality, but those reviews are discouraged when they prevent quick delivery. For those reasons, formal process definitions exist, though teams do not precisely follow them when they challenge cultural drivers within the organization.

Four of seven participants indicated that process is more of a guideline than a prescriptive notion (see Table 6 for information source metrics). Two of those participants mentioned that benchmarking and measuring for continuous improvement is part of a prescribed process they follow. Three of the participants indicated that they tend to vary from prescribed processes when the processes are blocking progress or quality with one participant mentioning the challenges architecture teams face with agile methodologies, given agile methods do not formally allot time for architecture. Two participants noted that the importance of process varies by risk, stating that a team developing medical devices requires more stringent processes than a team developing games, as an example.

Table 6

Minor Themes of Use Process as a Guideline with Supporting Metrics

	Participant		Document	
Major/Minor Theme	Count	References	Count	References
Use process as a guideline	4	9	9	10
Benchmark, measure, and continuously improve	2	2	7	7
Process importance is driven by risk tolerance	3	5	1	1

Authors of nine organizational documents supported the theme of using processes as a guideline (see Table 6 for information source metrics). The author of one document highlighted the case organization’s use of architecture in their risk model, which prioritized delivery time over comprehensive solutions. This document’s author addressed the role of architects in agile development methodologies, noting that different types of architects have different deliverables and responsibilities in various phases of the product's lifecycle. For example, project architects develop solution architecture documents in Sprint 0 to aid in project approvals. Also in Sprint 0, project architects work with others to define business requirements, finalize solution architecture documents, and prepare for technical quality assessment reviews. In sprints 1-n, project architects prepare architectural scorecards to measure progress against approved architecture. The author of another document defined architectural governance as a meta-model, explaining how architectural governance interleaves with any development methodology teams prefer to use. The authors of seven documents supported benchmarking and continuous improvement, providing processes and tools to measure the solution's quality alignment with organizational objectives. Examples were the technical quality assessment and related architectural principles, architectural scorecards, and role profiles that explained the role of architects in these activities.

Using process as a guideline does not align well with TQM. Deming (1982), Crosby (1992), Ishikawa (1984), Juran (1999), and Shewhart (1939) all agreed that having a formal process is the basis of continuous improvement, making it more than a simple guideline. One research participant noted that delivering a product follows a process whether or not the team follows a well-defined, formal process. That approach complicates continuous improvement. Juran (1999) believed that determining continuous improvement relies on benchmarking of both process and product. In the literature, Richardson et al. (2012) believed in process improvement so much that they developed the global teaming model to extend the concepts introduced in CMMI to geographically dispersed development teams due to process peculiarities in those environments.The importance of formal process appears to vary by organizational culture.

Well-defined processes are not static and are not limited to specific activities. Crosby (1982) noted that processes are important parts of all types of work, including creative work. Deming (1982) and Juran (1999) agreed that customer satisfaction is not static; that it is constantly changing. For that reason, processes and products must undergo continuous improvement to maintain customer satisfaction (Shanmugasundaram & Vikram, 2015). One participant summarized the overarching perceived perspective on process, stating the following:

> Some people have…very regimented process[es], especially in terms of collaterals they deliver, describing a solution. I do not…disagree with any of those, good or bad, I just don't always use them if I do not think they are necessary.

Applications to Professional Practice

The specific IT problem that formed the basis of this research was the perceived lack of strategies used by enterprise and application architects in the industry for applying architectural best practices to improve bespoke software quality. Participants in this research provided strategies that other enterprise and application architects could apply to maximize bespoke software quality. There were different thoughts on architectural best practices, indicating that the myriad best practices in the industry applied to different types of projects in a variety of ways. The majority of participants stated that they relied on industry best practices as a guideline as opposed to prescriptive standards. After evaluating the collected data, I identified five primary themes: focus on customer satisfaction, effectively collaborate and communicate with all stakeholders, define boundaries and empower people within those boundaries, deliver prototypes and work products, and use development processes as a guideline. There are a few ways that organizations and individuals either in the role of architect or aspiring architects can use these results.

Organizations that include bespoke software development in their capability model can use these results to set or update organizational policies. The overarching policy is what W. Edwards Deming referred to as a culture of quality, or making customer satisfaction the focus of architectural activities. Organizational leaders can choose to implement this quality culture. In the case of bespoke software development, the customer is the organizational employee that consumes the solution. The policy should recommend that teams—and specifically architects—collaborate and communicate with consumers to understand their needs and foster an overall positive consumer experience.

Any organizational policy that establishes a culture of quality could follow the themes found in this research, amongst others. Leaders should define the policy in a way that clearly states roles and responsibilities such that individuals filling those roles know their accountabilities and the accountabilities of others. To enable the intrinsic empowerment of individuals filling the roles, organizational leaders should effectively communicate these roles and responsibilities throughout the organization. As with any policy, implementation of these roles and responsibilities should be governed and either incented or enforced. Organizations are not the only entities that can use these results.

Individuals aspiring to be architects can use these findings as well. Senior developers and people with no development background aspiring to become architects should learn to change their definition of quality from technical quality to customer satisfaction. Changing their definition of quality requires that they learn to understand their customer’s needs as fully as possible and communicate effectively with both technical and nontechnical stakeholders. Aspiring architects should also learn to deliver solution models and prototypes that prove the concept without being production-ready solutions. They should also learn to use processes as a guideline, knowing when to bypass the details of the process in the pursuit of customer satisfaction. One more group can benefit from these findings.

Existing architects can use these results to evaluate the strategies they currently use. Most of the participants in this research indicated that—while participating in the interviews—it was interesting to consider the use of the strategies that they used in a way that they had not considered before, influencing the way they now perceive their role. Focusing on customer satisfaction does not simply mean talking with consumers about their needs. It means truly understanding how they perceive quality, including minimized lifecycle costs, ensuring the solution is available when they need it, ensuring the solution is usable by the customer's definition, and that the customer's experience is positive throughout designing and implementing the solution.

Implications for Social Change

My initial expectations for social change included improved work-life balance, morale, and productivity of users and software and enterprise architects. I now believe those expectations to be narrow and shortsighted as they focused on the architect. While I expected that architects would benefit from the findings and recommendations of this research, customers, organizational leaders, software engineers, and aspiring architects involved in delivering bespoke software will also realize benefits.

Instilling the concept of quality culture has wide-ranging implications. Consumers of bespoke solutions will realize better work-life balance through a reduction in required effort to complete their work tasks. Organizations will realize a reduction in costs of both developing bespoke solutions through lower defect rates and less time spend testing and remediating as well as the cost savings associated with increased consumer efficiency. Software engineers will spend less time remediating low-quality code and more time maximizing the efficiency of consumers. All of these stakeholders will realize lower stress, improved work-life balance, and improved morale.

The potential for social and cultural change exists outside of the case organization and IT practitioners as well. When development teams improve the quality of bespoke solutions used to deliver products and services, organizational costs are minimized and organizations could pass those cost savings on to customers and other organizations. When organizations base the prices of their products and services on the cost-plus-expenses model, a reduction in the cost of any materials or aspects of delivering the product has the potential to affect the prices of all related products and services. Reducing the cost of consumer goods and services benefits the overall economy and society as consumers can choose how to manage the additional funds, either through additional spending or through savings.

Additionally, organizational customer satisfaction would increase. Consumers, either current or potential, become frustrated when communications with providers of products and services have inefficient or ineffective communications. Increasing the efficiency and effectiveness of organizational staff can result in better communications with customers. Better communications and lower frustration levels result in less stress and better overall health of consumers. I believe that many contemporary organizations have either forgotten about or disregarded customer service, resulting in frustrated, unhappy, or lost customers. This lowered sense of customer service can be, at least in part, addressed by improving the quality of bespoke solutions employees use when interacting with customers. With enough implementation, emphasis, and results, improved bespoke software quality could result in a societal shift away from its current everything-is-disposable perspective when working to repair a product or service becomes less costly—economically, emotionally, and otherwise—than replacing it.

Recommendations for Action

The architects from the case organization that participated in this research aligned with the concepts of TQM. They focused on customer satisfaction by working to deliver available and usable solutions at the lowest possible cost. I did not include a determination of whether or not there was a perception of low-quality bespoke solutions at the case organization in this research, making any recommendations for action within the case organization speculative at best. Additionally, actions already taken by architecture team leaders to align the EAT with the product development team were not included in this research, adding to the speculative nature of recommendations within the case organization. The generalized recommendation with the assumptions that the case organization does perceive the low quality of their bespoke solutions and that work is already underway to align the architecture team with development teams is to continue work activities to align the architecture team and product development teams. This recommendation is appropriate for architecture team leaders and development team leaders alike.

Outside of the case organization, I recommend that organizational leaders consider implementing or extending the implementation of a quality culture in their organization. Potential actions include training not only architects but also development team leaders in the concepts of TQM and the leadership skills found in French and Raven's (1959) bases of social power, focusing on the base that best aligns with their organizational culture. I also recommend additional training in human-computer interactions and the psychology behind perceptions of quality for appropriate leaders.

Recommendations for Further Study

I have several recommendations for further research, some deriving from the limitations noted in this research and others arising from the findings of this study. The limitations of this research included the potential influence of bias and preconceived notions on the results due to the subjective nature of qualitative research. The first recommendation is to continue this research as additional qualitative studies at other case organizations to compare and contrast the results with those from one or more other case organizations. An additional limitation of this research was the inability to generalize the results outside of the case organization due to the nature of qualitative case study research. I recommend the continuation of this research as a quantitative study to determine the generalizability of the findings.

Recommendations for further research based on the results of this research are varied. Due to this research focusing on the architect's involvement in quality, I recommend performing the same research with software engineers instead of architects to compare and contrast the results. The results would confirm or refute the perceived gap between architects and software engineers.

One of the serendipitous comments from a few of the participants in this research was that the roles of application architect and software engineer are converging. I recommend future multiple case studies of application architects and enterprise architects as independent cases to compare and contrast the results from each. This research would help identify the validity of the perception of role convergence and possibly set a precedent for quantitative research, such as a survey design, to determine the ubiquity of the perception.

While I considered including the concept of architect empowerment as a primary question in this research, I decided to focus on overall strategies. Empowerment did arise briefly in the reviewed literature and again in the collected data. I could not substantiate the concept of empowerment by itself in the collected data, only in the context of empowerment within boundaries. I recommend further qualitative research of the importance of architect empowerment, perhaps using French and Raven's (1959) Bases of Power as a framework.

Reflections

As a professional who has worked as both a software engineer and architect involved with delivering bespoke software solutions, I have always held customer satisfaction in high regard. I had academic exposure to the concepts of TQM and human-computer interactions several years ago and worked throughout my career to implement the influences in bespoke software development. In this research, I was as diligent as possible in my analysis to remain objective in the results, though it is possible that I unknowingly and unintentionally biased this research through the framing of interview questions and analysis of the collected data. While I remain steadfast in my recommendation to focus on the complex concept of customer satisfaction, I have realized that establishing a trust relationship with not only the consumers but the development team is required. I understand that involves exercising both humility and diplomacy when the organizational culture favors influential power over coercive, reward, or expert power.

Summary and Study Conclusion

The quality of bespoke software is both subjective and complex. It is a combination of how the consumer feels about using the software, how well they can accomplish what they need to accomplish with it, how well it meets their specifications, how easy it is to fix when something goes wrong, and their perception of how valuable the software is. Only the consumer can determine the quality of a software solution. In bespoke software development, there are many opportunities to understand your consumer, to understand what quality means to them. The role of the architect is more than technical mastery. It is more than the next colloquial rung in the software engineering career ladder. It is personal. It is qualitative. It is customer-centricity. The architect of bespoke solutions is responsible for bringing together two stakeholder groups with diverse knowledge and understanding, leading them to quality: business consumers and development teams. Accomplishing this requires deeply understanding and addressing both group's needs, doing so with diplomacy and humility.

References

Ahmad, R. A. (2016). Interface-driven software requirements analysis. *European Scientific Journal*, *12*(30), 40–54. doi:10.19044/esj.2016.v12n30p40

Aleti, A., Buhnova, B., Grunske, L., Koziolek, A., & Meedeniya, I. (2013). Software architecture optimization methods: A systematic literature review. *IEEE Transactions on Software Engineering*, *39*, 658–683. doi:10.1109/TSE.2012.64

Alfaro, I., & Chandrasekaran, R. (2015). Software quality and development speed in global software development teams: The role of previous working ties and national diversity. *Business & Information Systems Engineering*, *57*, 91–102. doi:10.1007/s12599-015-0372-6

Alotaibi, F. M. S. (2014). Impact on quality culture of total quality management practices factors. *International Journal of Business and Economic Development*, *2*(3), 35–48. Retrieved from http://www.ijbed.org/

Ameller, D., Ayala, C., Cabot, J., & Franch, X. (2013). Non-functional requirements in architectural decision making. *IEEE Software*, *30*(2), 61–67. doi:10.1109/MS.2012.176

Ameller, D., Galster, M., Avgeriou, P., & Franch, X. (2013). The role of quality attributes in service-based systems architecting: A survey. *Lecture Notes in Computer Science*, *7957*, 200–207. doi:10.1007/978-3-642-39031-9_18

Andriole, S. J. (2015). The need for new business-technology relationships. *IT Professional*, *17*(4), 4–6. doi:10.1109/MITP.2015.60

Atkinson, S., & Benefield, G. (2013). Software development: Why the traditional contract model is not fit for purpose. In *2013 46th Hawaii International Conference on System Sciences* (pp. 4842–4851). Maui, Hawaii: IEEE. doi:10.1109/HICSS.2013.492

Avgeriou, P., Kruchten, P., Nord, R. L., Ozkaya, I., & Seaman, C. (2016). Reducing friction in software development. *IEEE Software*, *33*(1), 66–73. doi:10.1109/MS.2016.13

Aviles, M. E. (2015). *The impact of cloud computing in supply chain collaborative relationships , collaborative advantage and relational outcomes* (Doctoral dissertation, Georgia Southern University). Retrieved from http://digitalcommons.georgiasouthern.edu/

Bailey, S. E., Godbole, S. S., Knutson, C. D., & Krein, J. L. (2013). A decade of Conway's law: A literature review from 2003-2012. In *2013 3rd International Workshop on Replication in Empirical Software Engineering Research* (pp. 1–14). Baltimore, MD, USA: IEEE. doi:10.1109/RESER.2013.14

Barata, J., & Cunha, P. R. (2015). Synergies between quality management and information systems: A literature review and map for further research. *Total Quality Management & Business Excellence*, Advance online publication. doi:10.1080/14783363.2015.1080117

Belk, M., Papatheocharous, E., Germanakos, P., & Samaras, G. (2013). Modeling users on the world wide web based on cognitive factors, navigation behavior and clustering techniques. *Journal of Systems and Software*, *86*, 2995–3012. doi:10.1016/j.jss.2013.04.029

Bon, A. T., & Mustafa, E. M. A. (2013). Impact of total quality management on innovation in service organizations: Literature review and new conceptual framework. *Procedia Engineering*, *53*, 516–529. doi:10.1016/j.proeng.2013.02.067

Booch, G. (2015). Then a miracle occurs. *IEEE Software*, *32*(4), 12–14. doi:10.1109/MS.2015.90

Bosch, J., Capilla, R., & Hilliard, R. (2015). Trends in systems and software variability [Guest editors' introduction]. *IEEE Software*, *32*(3), 44–51. doi:10.1109/MS.2015.74

Brown, A. (2014). Organisational paradigms and sustainability in excellence. *International Journal of Quality and Service Sciences*, *6*, 181–190. doi:10.1108/IJQSS-02-2014-0020

Brown, L. R. (2014). *Qualitative study of the end user's perception of software as a service (SaaS) in the cloud* (Doctoral dissertation, Colorado Technical University). Retrieved from http://search.proquest.com/docview/1656502761

Buenen, M., & Walgude, A. (2015). *World quality report: 2015-16*. Retrieved from http://www.worldqualityreport.com

Canessane, R. A., & Srinivasan, S. (2013). The software architecture towards performance evolution. *I-Manager's Journal on Software Engineering*, *8*(1), 24–34. Retrieved from http://www.imanagerpublications.com/JournalIntroduction.aspx?journal=imanagersJournalonSoftwareEngineering

Carver, J. C., Yamashita, A., Minku, L., Habayeb, M., & Kocak, S. A. (2015). Software quality, energy awareness, and more. *IEEE Software*, *32*(6), 98–100. doi:10.1109/MS.2015.127

Chen, J. C., & Huang, S.-J. (2009). An empirical analysis of the impact of software development problem factors on software maintainability. *Journal of Systems and Software*, *82*, 981–992. doi:10.1016/j.jss.2008.12.036

Chou, S.-W., & Chiang, C.-H. (2013). Understanding the formation of software-as-a-service (SaaS) satisfaction from the perspective of service quality. *Decision Support Systems*, *56*, 148–155. doi:10.1016/j.dss.2013.05.013

Cleland-Huang, J. (2014). Don't fire the architect! Where were the requirements? *IEEE Software*, *31*(2), 27–29. doi:10.1109/MS.2014.34

Cleland-Huang, J., Hanmer, R. S., Supakkul, S., & Mirakhorli, M. (2013). The twin peaks of requirements and architecture. *IEEE Software*, *30*(2), 24–29. doi:10.1109/MS.2013.39

Cronemyr, P., & Danielsson, M. (2013). Process management 1-2-3 – a maturity model and diagnostics tool. *Total Quality Management & Business Excellence*, *24*, 933–944.

doi:10.1080/14783363.2013.791114

Cronin, C. (2014). Using case study research as a rigorous form of inquiry. *Nurse Researcher*, *21*(5), 19–27. doi:10.7748/nr.21.5.19.e1240

Crosby, P. B. (1982). The management of quality. *Research Management*, *15*(4), 10–12. Retrieved from http://www.tandfonline.com/toc/urtm20/current

Crosby, P. B. (1992). *Completeness: Quality for the 21st century*. New York, New York, USA: Penguin Group.

Dai, H., Luo, X., Liao, Q., & Cao, M. (2015). Explaining consumer satisfaction of services: The role of innovativeness and emotion in an electronic mediated environment. *Decision Support Systems*, *70*, 97–106. doi:10.1016/j.dss.2014.12.003

De Massis, A., & Kotlar, J. (2014). The case study method in family business research: Guidelines for qualitative scholarship. *Journal of Family Business Strategy*, *5*(1), 15–29. doi:10.1016/j.jfbs.2014.01.007

Dehaghani, S., & Hajrahimi, N. (2013). Which factors affect software projects maintenance cost more. *Acta Informatica Medica*, *21*, 63–66. doi:10.5455/aim.2012.21.63-66

Deming, W. E. (1982). *Out of the crisis*. Cambridge, MA: Massachusetts Institute of Technology, Center for Advanced Engineering Study.

Deming, W. E. (1985). Transformation of Western Style of Management. *Interfaces*, *15*(3), 6–11. Retrieved from http://pubsonline.informs.org/journal/inte

Dempsey, L., Dowling, M., Larkin, P., & Murphy, K. (2016). Sensitive interviewing in qualitative research. *Research in Nursing & Health*, Advance online publication. doi:10.1002/nur.21743

Despa, M. L. (2015). Formalizing the ISDF software development methodology. *Informatica*

Economica, *19*(2/2015), 66–80. doi:10.12948/issn14531305/19.2.2015.07

deWinter, J., Kocurek, C. A., & Nichols, R. (2014). Taylorism 2.0: Gamification, scientific management and the capitalist appropriation of play. *Journal of Gaming & Virtual Worlds*, *6*, 109–127. doi:10.1386/jgvw.6.2.109_1

Digalwar, A. K., Haridas, P., & Joseph, I. N. (2014). Development and validation of quality management constructs for software industries: An empirical investigation from India. *International Journal of Services and Operations Management*, *17*, 453–478. doi:10.1504/IJSOM.2014.060003

Ding, S., Yang, S., Zhang, Y., Liang, C., & Xia, C. (2014). Combining QoS prediction and customer satisfaction estimation to solve cloud service trustworthiness evaluation problems. *Knowledge-Based Systems*, *56*, 216–225. doi:10.1016/j.knosys.2013.11.014

Draper, J. (2015). Ethnography: Principles, practice and potential. *Nursing Standard*, *29*(36), 36–41. doi:10.7748/ns.29.36.36.e8937

Ebert, C., Hoefner, G., & V.S., M. (2015). What next? Advances in software-driven industries. *IEEE Software*, *32*(1), 22–28. doi:10.1109/MS.2015.21

Faden, R. R., Kass, N. E., Goodman, S. N., Pronovost, P., Tunis, S., & Beauchamp, T. L. (2013). An ethics framework for a learning health care system: A departure from traditional research ethics and clinical ethics. *Hastings Center Report*, *43*(s1), S16–S27. doi:10.1002/hast.134

Feigenbaum, A. V. (1985). Quality: Managing the modern company. *Quality Progress*, *18*(3), 18–21. Retrieved from http://asq.org/qualityprogress/index.html

Felt, U., Igelsböck, J., Schikowitz, A., & Völker, T. (2013). Growing into what? The (un-)disciplined socialisation of early stage researchers in transdisciplinary research. *Higher*

Education, *65*, 511–524. doi:10.1007/s10734-012-9560-1

French, J. R. P., & Raven, B. (1959). The bases of social power. In *Studies in social power* (pp. 150–167). Ann Arbor, MI: Institute for Social Research, University of Michigan.

Fusch, P. I., & Ness, L. R. (2015). Are we there yet?: Data saturation in qualitative research. *The Qualitative Report*, *20*, 1408–1416. Retrieved from http://nsuworks.nova.edu/tqr

Gagnon, M., Jacob, J. D., & McCabe, J. (2015). Locating the qualitative interview: Reflecting on space and place in nursing research. *Journal of Research in Nursing*, *20*, 203–215. doi:10.1177/1744987114536571

Galster, M., Avgeriou, P., & Tofan, D. (2013). Constraints for the design of variability-intensive service-oriented reference architectures – An industrial case study. *Information and Software Technology*, *55*, 428–441. doi:10.1016/j.infsof.2012.09.011

Galster, M., Weyns, D., Tofan, D., Michalik, B., & Avgeriou, P. (2014). Variability in software systems: A systematic literature review. *IEEE Transactions on Software Engineering*, *40*, 282–306. doi:10.1109/TSE.2013.56

Gentles, S. J., Charles, C., Ploeg, J., & McKibbon, K. A. (2015). Sampling in qualitative research: Insights from an overview of the methods literature. *The Qualitative Report*, *20*, 1772–789. Retrieved from http://nsuworks.nova.edu/tqr

Gimenez-Espin, J. A., Jiménez-Jiménez, D., & Martínez-Costa, M. (2013). Organizational culture for total quality management. *Total Quality Management & Business Excellence*, *24*, 678–692. doi:10.1080/14783363.2012.707409

Goode, S., Lin, C., Tsai, J. C., & Jiang, J. J. (2015). Rethinking the role of security in client satisfaction with Software-as-a-Service (SaaS) providers. *Decision Support Systems*, *70*, 73–85. doi:10.1016/j.dss.2014.12.005

Göransson, B., Gulliksen, J., & Boivie, I. (2003). The usability design process - Integrating user-centered systems design in the software development process. *Software Process: Improvement and Practice*, *8*, 111–131. doi:10.1002/spip.174

Guimaraes, T., & Paranjape, K. (2014). Testing cloud computing for customer satisfaction and loyalty. *International Journal of Electronic Customer Relationship Management*, *8*(1/2/3), 72–89. doi:10.1504/IJECRM.2014.066885

Gürses, S., Seguran, M., & Zannone, N. (2013). Requirements engineering within a large-scale security-oriented research project: Lessons learned. *Requirements Engineering*, *18*(1), 43–66. doi:10.1007/s00766-011-0139-7

Hallissy, B. P., Laiosa, J. P., Shafer, T. C., Hine, D. H., Forsythe, J. R., Abras, J., … Dahl, C. (2016). HPCMP CREATE-AV quality assurance: Lessons learned by validating and supporting computation-based engineering software. *Computing in Science & Engineering*, *18*(1), 52–62. doi:10.1109/MCSE.2015.136

Hill, D. A. (2008). *What makes total quality management work: A study of obstacles and outcomes* (Doctoral dissertation, Capella University). Retrieved from http://search.proquest.com/docview/304832602

Holweg, M., & Helo, P. (2014). Defining value chain architectures: Linking strategic value creation to operational supply chain design. *International Journal of Production Economics*, *147*, 230–238. doi:10.1016/j.ijpe.2013.06.015

Horcas, J.-M., Pinto, M., & Fuentes, L. (2016). An automatic process for weaving functional quality attributes using a software product line approach. *Journal of Systems and Software*, *112*, 78–95. doi:10.1016/j.jss.2015.11.005

Houghton, C., Casey, D., Shaw, D., & Murphy, K. (2013). Rigour in qualitative case-study research. *Nurse Researcher*, *20*(4), 12–17. doi:10.7748/nr2013.03.20.4.12.e326

Houston, A., & Dockstader, S. L. (1997). *Total quality leadership: A primer*. Washington, D.C.: Department of the Navy, Total Quality Leadership Office. Retrieved from http://unpan1.un.org/intradoc/groups/public/documents/aspa/unpan002508.pdf

Huang, S.-J., Wu, M.-S., & Chen, L.-W. (2013). Critical success factors in aligning IT and business objectives: A Delphi study. *Total Quality Management & Business Excellence*, *24*, 1219–1240. doi:10.1080/14783363.2011.637785

Huckabee, W. A. (2015). Requirements engineering in an agile software development environment. *Defense Acquisition Research Journal*, *22*, 394–415. Retrieved from http://www.dau.mil/publications/DefenseARJ/default.aspx

Hyett, N., Kenny, A., & Dickson-Swift, V. (2014). Methodology or method? A critical review of qualitative case study reports. *International Journal of Qualitative Studies on Health and Well-Being*, *9*, 1–12. doi:10.3402/qhw.v9.23606

Ishikawa, K. (1984). Quality and standardization: Program for economic success. *Quality Progress*, *17*(1), 16–20. Retrieved from http://asq.org/qualityprogress/index.html

Johann, S. (2015). Software architecture for developers. *IEEE Software*, *32*(5), 93–96. doi:10.1109/MS.2015.125

Jones, C. (2015). Wastage: The impact of poor quality on software economics. *Software Quality Professional*, *18*(1), 23–32. Retrieved from http://asq.org/pub/sqp/

Joseph, J. (2013). The use of the classical twin method in the social and behavioral sciences: The fallacy continues. *The Journal of Mind and Behavior*, *34*(1), 1–40. Retrieved from http://umaine.edu/jmb/

Joyce, S. A. (2015). Outstanding quality using team management. *International Journal of*

Metalcasting, *9*(3), 7–13. Retrieved from http://www.afsinc.org/technical/IJM.cfm

Juran, J., & Godfrey, A. B. (1999). *Juran's Quality Handbook*. New York: McGraw-Hill.

Kassab, M., Neill, C., & Laplante, P. (2014). Software development practices and software quality: A survey. *Software Quality Professional*, *16*(4), 36–42. Retrieved from http://asq.org/pub/sqp/

Katina, P. F. (2015). Emerging systems theory-based pathologies for governance of complex systems. *International Journal of System of Systems Engineering*, *6*, 144–159. doi:10.1504/IJSSE.2015.068806

Keeling, M. (2015). Lightweight and flexible: Emerging trends in software architecture from the SATURN conferences. *IEEE Software*, *32*(3), 7–11.

Khalil, R., Stockton, D., Alkaabi, M. S., & Manyonge, L. M. (2015). Investigating the effect of variability in product development life cycle. *International Journal of Product Development*, *20*, 495–513. doi:10.1504/IJPD.2015.073521

Klein, J. (2016). What makes an architect successful? *IEEE Software*, *33*(1), 20–22. doi:10.1109/MS.2016.9

Koch, S., Bener, A. B., Aytac, T., Misirli, A. T., & Bernroider, E. W. N. (2014). Influence of context and culture on managerial perceptions and practices in the Turkish software industry. *Journal of Global Information Technology Management*, *17*(1), 26–44. doi:10.1080/1097198X.2014.910990

Kruth, J. G. (2014). Five qualitative research approaches and their applications in parapsychology. *Journal of Parapsychology*, *79*, 219–233. Retrieved from http://www.rhine.org/what-we-do/journal-of-parapsychology.html

Lantos, J. D., & Spertus, J. A. (2014). The concept of risk in comparative-effectiveness research.

New England Journal of Medicine, *371*, 2129–2130. doi:10.1056/NEJMhle1413301

Li, P. L., Ko, A. J., & Zhu, J. (2015). What makes a great software engineer? In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (pp. 700–710). Florence, Italy: IEEE. doi:10.1109/ICSE.2015.335

Li, S.-H., Yen, D. C., Lu, W.-H., & Chen, T.-Y. (2014). The characteristics of information system maintenance: An empirical analysis. *Total Quality Management & Business Excellence*, *25*, 280–295. doi:10.1080/14783363.2013.807679

Liberatore, M. J., & Pollack-Johnson, B. (2013). Improving project management decision making by modeling quality, time, and cost continuously. *IEEE Transactions on Engineering Management*, *60*, 518–528. doi:10.1109/TEM.2012.2219586

Liu, J. W., Chang, J. Y. T., & Tsai, J. C. A. (2015). Does perceived value mediate the relationship between service traits and client satisfaction in the Software-as-a-Service (SaaS)? *Open Journal of Social Sciences*, *3*(7), 159–165. doi:10.4236/jss.2015.37026

Luftman, J., Zadeh, H. S., Derksen, B., Santana, M., Rigoni, E. H., & Huang, Z. (2013). Key information technology and management issues 2012–2013: An international study. *Journal of Information Technology*, *28*(4), 354–366. doi:10.1057/jit.2013.22

Lyu, J., & Liang, C.-C. (2014). Effective approach to quality control for small–medium software companies. *Total Quality Management & Business Excellence*, *25*, 296–308. doi:10.1080/14783363.2012.669545

Marchewka, J. T. (2013). Applying the capability maturity model to assurance of learning. *Communications of the IIMA*, *13*(1), 1–16. Retrieved from http://scholarworks.lib.csusb.edu/ciima/

Martinez-Fernández, S. (2013). Towards supporting the adoption of software reference architectures: An empirically-grounded framework. In *Proceedings 11th International Doctoral Symposium on Empirical Software Engineering* (pp. 1–8). Baltimore, MD, USA. Retrieved from http://upcommons.upc.edu/

Martini, A., Bosch, J., & Chaudron, M. (2015). Investigating architectural technical debt accumulation and refactoring over time: A multiple-case study. *Information and Software Technology*, *67*, 237–253. doi:10.1016/j.infsof.2015.07.005

McCusker, K., & Gunaydin, S. (2015). Research using qualitative, quantitative or mixed methods and choice based on the research. *Perfusion*, *30*, 537–542. doi:10.1177/0267659114559116

Mihalcin, M. J., Mazzuchi, T. A., Sarkani, S., & Dever, J. R. (2014). Quality control: An approach applying multivariate control charts during the operation of systems involving human processes. *Systems Engineering*, *17*, 204–212. doi:10.1002/sys.21263

Misra, S., Fernández, L., & Colomo-Palacios, R. (2014). A simplified model for software inspection. *Journal of Software: Evolution and Process*, *26*, 1297–1315. doi:10.1002/smr.1691

Mitchell, S. M. (2012). *Software process improvement through the removal of project-level knowledge flow obstacles: The perceptions of software engineers* (Doctoral dissertation, University of Maryland). Retrieved from https://search.proquest.com/dissertations/docview/1029444348/2C09CE3DDEAA4A40PQ/38

Pass, S., & Ronen, B. (2014). Reducing the software value gap. *Communications of the ACM*, *57*(5), 80–87. doi:10.1145/2594413.2594422

Patwardhan, A. (2016). Analysis of software delivery process shortcomings and architectural pitfalls. *PeerJ Computer Science*, Advance online publication. Retrieved from http://arxiv.org

Pérez-Aróstegui, M. N., Bustinza-Sánchez, F., & Barrales-Molina, V. (2015). Exploring the relationship between information technology competence and quality management. *BRQ Business Research Quarterly*, *18*, 4–17. doi:10.1016/j.brq.2013.11.003

Petty, N. J., Thomson, O. P., & Stew, G. (2012). Ready for a paradigm shift? part 2: Introducing qualitative research methodologies and methods. *Manual Therapy*, *17*, 378–384. doi:10.1016/j.math.2012.03.004

Poort, E. R. (2014). Driving agile architecting with cost and risk. *IEEE Software*, *31*(5), 20–23. doi:10.1109/MS.2014.111

Poth, A., & Sunyaev, A. (2014). Effective quality management: Value- and risk-based software quality management. *IEEE Software*, *31*(6), 79–85. doi:10.1109/MS.2013.138

Ramasubbu, N., Bharadwaj, A., & Tayi, G. K. (2015). Software process diversity: Conceptualization, measurement, and analysis of impact on project performance. *MIS Quarterly*, *39*(4), 787–807. Retrieved from http://www.misq.org

Richardson, I., Casey, V., McCaffery, F., Burton, J., & Beecham, S. (2012). A process framework for global software engineering teams. *Information and Software Technology*, *54*, 1175–1191. doi:10.1016/j.infsof.2012.05.002

Rimando, M., Brace, A. M., Namageyo-Funa, A., Parr, T. L., Sealy, D.-A., Davis, T., … Christiana, R. (2015). Data collection challenges and recommendations for early career researchers. *The Qualitative Report*, *20*, 2025–2036. Retrieved from http://nsuworks.nova.edu/tqr/

Rozanski, N. (2015). The five properties of successful architectural oversight. *IEEE Software*, *32*(1), 110–112. doi:10.1109/MS.2015.18

Ryan, K. J., Brady, J., Cooke, R. E., Height, D. I., Jonsen, A. R., King, P., … Turtle, R. (2014). The Belmont report: Ethical principles and guidelines for the protection of human subjects of research. *The Journal of the American College of Dentists*, *81*(3), 4–13. Retrieved from http://acd.org/publications.htm

Shah, H. (2014). *An ethnographically-informed analysis of the influence of culture on global software-testing practice* (Doctoral dissertation, Georgia Institute of Technology). Retrieved from https://smartech.gatech.edu/handle/1853/53983

Shanmugasundaram, P., & Vikram, P. (2015). Total quality management, process analytical technology, five basic principles and pharmaceutical industry: An overview. *International Journal of PharmTech Research*, *8*(6), 178–185. Retrieved from http://sphinxsai.com/pharmtech.php

Sharma, S., & Modgil, S. (2015). Supply chain and total quality management framework design for business performance-case study evidence. *Journal of Enterprise Information Management*, *28*, 905–930. doi:10.1108/JEIM-10-2014-0104

Shewhart, W. A. (1939). *Statistical method from the viewpoint of quality control*. Washington, D.C.: The Graduate School, the Department of Agriculture.

Simon, D., Fischbach, K., & Schoder, D. (2013). An exploration of enterprise architecture research. *Communications of the Association for Information Systems*, *32*, 1–72. Retrieved from http://aisel.aisnet.org/cais/

Simon, D., Fischbach, K., & Schoder, D. (2014). Enterprise architecture management and its role in corporate strategic management. *Information Systems and E-Business Management*, *12*, 5–42. doi:10.1007/s10257-013-0213-4

Singh, D., & Jatain, A. (2013). An interactive approach to requirements prioritization using quality factors. *International Journal in Foundations of Computer Science & Technology*, *3*(6), 25–40. doi:10.5121/ijfcst.2013.3603

Sloan, A., & Bowe, B. (2014). Phenomenology and hermeneutic phenomenology: The philosophy, the methodologies, and using hermeneutic phenomenology to investigate lecturers' experiences of curriculum design. *Quality & Quantity*, *48*, 1291–1303. doi:10.1007/s11135-013-9835-3

Spinellis, D. (2014a). Against the odds: Managing the unmanagable in a time of crisis. In *26th International Conference, CAiSE 2014* (Vol. 8484, pp. 24–41). Thessaloniki, Greece: Springer International Publishing. doi:10.1007/978-3-319-07881-6_3

Spinellis, D. (2014b). Bespoke infrastructures. *IEEE Software*, *31*(1), 23–25. doi:10.1109/MS.2014.2

Spinellis, D. (2015). Architecture from a developer's perspective. *IEEE Software*, *32*(5), 4–7. doi:10.1109/MS.2015.110

Stake, R. E. (1978). *Case studies in science education, volume II: Design, overview and general findings*. Retrieved from http://eric.ed.gov/

Suryanarayana, G., Sharma, T., & Samarthyam, G. (2015). Software process versus design quality: Tug of war? *IEEE Software*, *32*(4), 7–11. doi:10.1109/MS.2015.87

Talib, F., Rahman, Z., & Akhtar, A. (2013). An instrument for measuring the key practices of total quality management in ICT industry: An empirical study in India. *Service Business*, *7*, 275–306. doi:10.1007/s11628-012-0161-y

Teichler, U. (2014). Opportunities and problems of comparative higher education research: The daily life of research. *Higher Education*, *67*, 393–408. doi:10.1007/s10734-013-9682-0

Thakurta, R. (2013). A framework for prioritization of quality requirements for inclusion in a software project. *Software Quality Journal*, *21*, 573–597. doi:10.1007/s11219-012-9188-5

Topalović, S. (2015). The implementation of total quality management in order to improve production performance and enhancing the level of customer satisfaction. *Procedia Technology*, *19*, 1016–1022. doi:10.1016/j.protcy.2015.02.145

Trochim, W. M. K., & Donnelly, J. P. (2008). *The research methods knowledge base* (3rd ed.). Mason, OH: Cengage Learning.

Tunkelo, T., Hameri, A.-P., & Pigneur, Y. (2013). Improving globally distributed software development and support processes - A workflow view. *Journal of Software: Evolution and Process*, *25*, 1305–1324. doi:10.1002/smr.1604

Tyagi, S., Choudhary, A., Cai, X., & Yang, K. (2015). Value stream mapping to reduce the lead-time of a product development process. *International Journal of Production Economics*, *160*, 202–212. doi:10.1016/j.ijpe.2014.11.002

Vaismoradi, M., Turunen, H., & Bondas, T. (2013). Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. *Nursing & Health Sciences*, *15*(3), 398–405. doi:10.1111/nhs.12048

van Vliet, H., & Tang, A. (2016). Decision making in software architecture. *Journal of Systems and Software*. doi:10.1016/j.jss.2016.01.017

Verner, J. M., Babar, M. A., Cerpa, N., Hall, T., & Beecham, S. (2014). Factors that motivate software engineering teams: A four country empirical study. *Journal of Systems and Software*, *92*, 115–127. doi:10.1016/j.jss.2014.01.008

Wahyuni, D. (2012). The research design maze: Understanding paradigms, cases, methods and methodologies. *Journal of Applied Management Accounting Research*, *10*(1), 69–80.

doi:10.1675/1524-4695(2008)31

Wallace, S., Reid, A., Clinciu, D., & Kang, J. (2013). Culture and the importance of usability attributes. *Information Technology & People*, *26*(1), 77–93. doi:10.1108/09593841311307150

Wells, R. S., Kolek, E. A., Williams, E. A., & Saunders, D. B. (2015). How we know what we know: A systematic comparison of research methods employed in higher education journals, 1996-2000 v. 2006-2010. *Journal of Higher Education*, *86*(2), 171–195. doi:10.1353/jhe.2015.0006

Whitney, K., Bradley, J. M., Baugh, D. E., & Chesterman, C. W. (2015). Systems theory as a foundation for governance of complex systems. *International Journal of System of Systems Engineering*, *6*, 15–32. doi:10.1504/IJSSE.2015.068805

Wilson, V. (2014). Research methods: Triangulation. *Evidence Based Library and Information Practice*, *9*(1), 74–75. Retrieved from https://ejournals.library.ualberta.ca/index.php/EBLIP

Woods, E. (Ed.). (2014). Return of the Pragmatic Architect. *IEEE Software*, *31*(3), 10–13. doi:10.1109/MS.2014.69

Yi, L., Chanle, W., Lei, H., & Gang, Y. (2013). A component-based software development method combined with enterprise architecture. In *Proceedings of the 2013 International Conference on Advanced Computer Science and Electronics Information* (pp. 87–91). Paris, France: Atlantis Press. doi:10.2991/icacsei.2013.22

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

Yin, R. K. (1981). The case study as a serious research strategy. *Science Communication*, *3*(1), 97–114. doi:10.1177/107554708100300106

Yin, R. K. (2014). *Case study research: Design and methods* (5th ed.). Thousand Oaks, CA: SAGE Publications.

Yoon, B., Lee, K., Lee, S., & Yoon, J. (2015). Development of an R&D process model for enhancing the quality of R&D: Comparison with CMMI, ISO and EIRMA. *Total Quality Management & Business Excellence*, *26*, 746–761. doi:10.1080/14783363.2014.882040

Zhou, F., Ji, Y., & Jiao, R. J. (2013). Affective and cognitive design for mass personalization: Status and prospect. *Journal of Intelligent Manufacturing*, *24*(5), 1047–1069. doi:10.1007/s10845-012-0673-2

Appendix A: Case Study / Interview Protocol

Topic: strategies used by software and enterprise architects to apply architectural best practices to maximize the quality of bespoke software products.

Sources of data collected:

___ Interviews (face-to-face or phone) ___ Documents

___ Agency records ___ Multimedia data ___ Observations

Interview Protocol

Date and Time		
Location		
Participant ID		
Step 1	Introduction	Thank you for your time and for participating in this interview. My name is Dan Wagner and I am a doctor of information technology candidate at Walden University. I have both worked in and studied the software industry since 1995.
Step 2	Purpose	The purpose of this study is to explore strategies used by enterprise and application architects to apply architectural best practices to maximize the quality of custom software.
Step 3	Describe reason for participation	The information you provide today, both in interview responses and in any documentation or other sources you may have, will support my study in partial fulfillment of the degree of Doctor of Information Technology from Walden University.
Step 4	Describe benefit of participation	This information could add to academic and professional bodies of knowledge on quality strategies and is geared towards application and enterprise architects and

		anyone else interested in maximizing the quality of custom software solutions. There is no compensation of any sort associated with your participation.
Step 5a	Discuss ethics	To maintain ethical standards and respect your right to privacy, I am requesting your permission to record the audio of this conversation and keep notes on this entire session starting now. Once audio recording starts, I will introduce this session using your participant ID <Participant ID> and ask you to reconfirm your permission to record and take notes on this session. Is it ok to start recording now?
Step 5b	Start recording	My name is Dan Wagner, and I am here with Participant <X>; today's date is <Y>. Would you please confirm that I have provided you with background information on this study including the purpose, the reason for your participation, benefits of participation, and that you approve of my recording and taking notes during this session?
Step 6	Discuss confidentiality	Please feel free to decline to answer any question or stop participating at any time; this is a completely voluntary session. You are free to decline to answer any individual questions or decline to provide any information if you are not comfortable providing the information. All information you provide will be treated as strictly confidential and will not be disclosed to anyone, including your employer. I request that you avoid using organizational or individual names or any indicators that could be used to identify

		your organization or individuals in your responses. Any names or comments that are mentioned in the interview will be removed from the transcripts and will not be included in the final report. I also request that you do not discuss your participation with anyone until the study concludes. Any information provided in any form in this session will only be used for the purpose of this study, which will be presented in composite form with data from other participants in a doctoral study that may be published. None of your responses will be presented in individual form. I will keep research records in an encrypted and password-protected format, locked in a safe for five years, after which time they will be destroyed. Only I will have access to this data during that five-year period.
Step 7	Ask if there are any questions and if they want to proceed	Do you have any questions for me before we start? If no, are you ready to proceed?
Step 8	Transition to the interview	This is a semistructured interview that is about understanding your thoughts on the topic and questions. I have a few questions outlined for which your open and honest thoughts are appreciated. I am interested in your thoughts about these questions and ask that you not consider any prior relationship I may have with you or the topic in your responses. I may ask for more thoughts or explanations on portions of your responses. As much information as you can provide on your

		thoughts and perspective is greatly appreciated.
Step 9a	Interview	- What is your current role and how long have you been in similar roles? - Have you worked in any other roles over your career in the delivery of bespoke software solutions? - How would you describe the distinction between a software architect and a software engineer? Please explain - What is your understanding of architectural best practices? Please explain. - What does quality mean to you? Please explain. - What is your perception of the relationship between architects and software engineers regarding the pursuit of product quality? Please explain. - What, if any, challenges do you face regarding the application of architectural best practices in delivered software products? Please elaborate. - What strategies do you have for ensuring the highest possible quality of software products? Please elaborate.
Step 9b	Possible follow up questions	- How long did you work as an <previous role as mentioned>? - What dictates or determines quality? - What do you believe constitutes or makes up customer satisfaction? - What if your initial strategies fail? What alternate strategies might you employ? - What are your thoughts on empowerment or authority in the

		context of achieving quality? Please explain.
Step 10	Gather any secondary data or artifacts from the participant	That concludes the interview portion of the meeting. Do you have any documents, multimedia presentations, or other information with you that I can collect at this time?
Step 11	Conclusion	Thank you for your time today. To ensure I have interpreted your responses correctly, I would like to schedule a follow-up interview with you in a few days. Would that be acceptable? Is there a preferred method of communication for rescheduling? Thank you again.

Appendix B: Naturalistic Observation Protocol

Topic: strategies used by software and enterprise architects to maximize the quality of bespoke software products.

Date and Time	
Location	
Participant ID	

Notes or ABCs:

Antecedent	Behavior	Consequence

Appendix C: Invitation to Participate Email Template

Dear <First name>,

My name is Daniel Wagner, and I am a Doctor of Information Technology candidate at Walden University. As part of my doctoral program, I am researching strategies used by application and enterprise architects in the pursuit of high-quality software products developed for internal organizational use. I believe that you support the principles I am researching and would like to include you in my research.

I have attached a copy of the organizational approval to conduct my research and a consent form with details of my study for your consideration. If you read through the consent form and would like to participate, please forward a signed copy of the consent form to me at <email address redacted>. If you do not wish to participate for any reason, no communication is necessary. Participation in this study is entirely voluntary; you can choose to not participate or withdraw from the study with no personal or professional consequences. Interviews and other data collection activities are anticipated to occur in mid- to late-February, possibly extending into early March 2017. I will work with you to schedule participation times that do not adversely affect your work schedule or work tasks.

I thank you for your consideration and look forward to working with you.

Daniel Wagner

Doctor of Information Technology candidate

Walden University

<email address redacted>

Appendix D: Confidentiality Agreement

Name of Signer:

During the course of my/our activity in collecting or processing data for this research: "Architect Influence on Quality", I /we will have access to information that is confidential and should not be disclosed. I acknowledge that the information must remain confidential and that improper disclosure of confidential information can be damaging to the participant.

By signing this Confidentiality Agreement, I acknowledge and agree that:

1. I will not disclose or discuss any confidential information with others, including friends or family.
2. I will not in any way divulge, copy, release, sell, loan, alter or destroy any confidential information except as properly authorized.
3. I will not discuss confidential information where others can overhear the conversation. I understand that it is not acceptable to discuss confidential information even if the participant's name is not used.
4. I will not make any unauthorized transmissions, inquiries, modification or purging of confidential information.
5. I agree that my obligations under this agreement will continue after termination of the job that I will perform.
6. I understand that violation of this agreement will have legal implications.
7. I will only access or use systems or devices I am officially authorized to access and I will not demonstrate the operation or function of systems or devices to unauthorized individuals.

Signing this document, I acknowledge that I have read the agreement and I agree to comply with all the terms and conditions stated above.

Signature: **Date:**