THE NEW ERA OF SOFTWARE QUALITY ASSURANCE

Organizational Design for Measurement

By

Damico S. Nicome

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Management and Systems
At the
School of Continuing and Professional Studies
New York University

Spring, 2010

Table of Contents

Table of Tables

Table of Figures

Dedication

I dedicate this thesis to my father, Roosevelt L. Nicome and my Mother, Shirley D. Nicome for their many sacrifices and undying love. Their support of my academic and professional career has been invaluable to me. My father, the pioneer of our family, has spent his entire career as an educator dedicated to enriching the lives of children and adults all over the world. It has been my blessing to be the son of a man whose relentless pursuit of education is my most cherished inheritance. I aspire to pass his blessing on to my progeny. He has set the bar extremely high for those to come and it is my hope that this thesis gives him some satisfaction that his sacrifices have not been in vain.

My mother's compassion and lessons of love have helped me to temper my pursuit of success with kindness and honesty. She always knew the right things to say and more importantly, knew when to acquiesce. I have witnessed her own personal struggles in pursuit of her education and have been able to draw on her strength of character, integrity and perseverance to help me overcome the many challenges I have faced during the course of my life. She has been and continues to be my confidant, my friend and my biggest fan.

Thank you mom and dad for your love and support, may you both enjoy your retirement surrounded by love and peace knowing that the fruit of your labor has helped to make the world I live in a better place.

Acknowledgements

I thank my loving wife, Michelle Nicome, for her support of my continuing education. Her moral support, constant reviews and challenging queries have undeniably helped to improve this thesis. She is the driving force behind all that I do and deserves much of the credit for all of the successes we enjoy. I love you dearly Michelle and I am excited about what lies ahead for us.

I thank my loving son, Shamah Devonish, for his assistance with my research. His help in creating the distribution lists to collect survey responses was very much appreciated. Shamah's spirit was always kind and willing when asked for assistance and truly made the research process easier. Shamah, thank you for your help, I love you and wish you much success in all of your future endeavors. Do not hesitate to call on me whenever you are in need.

I want to express my deep appreciation to Joanne Peterson, President and CEO of Abator Information Services, Inc. Joanne you have been my colleague, my business partner, my mentor and most importantly, my friend. Thank you for your patience in responding to all of my ridiculous requests. Your professionalism, and integrity are rivaled only by your amazing sense of humor and I am so very grateful that you are on my side.

Declaration

I grant powers to the Department, SCPS, and NYU to allow this thesis to be copied in part or in whole without further reference to me. This permission covers only copies made for study purposes or for inclusion in Department, SCPS, and NYU research publications, subject to normal conditions of acknowledgement.

Abstract

An effective Quality Assurance (QA) organization must have ARMS, an acronym developed by the author to describe the organization's ability to perform certain measurement **Activities**, understand the activity's **Relevance** to the software quality model, know the type of **Metrics** to collect, and create **Synergy** with adjacent organizations. This thesis focused on the difficulty in obtaining measurement and explored whether or not QA organizational design can make it easier to measure certain characteristics of software quality. The thesis relied on primary and secondary sources to test the null hypothesis that there is no difference in difficulty to measure any ISO 9126 software quality sub-characteristic between a centralized and decentralized QA organization. An online survey was used as the mode of data collection. Survey respondents included seasoned IT professionals from distribution lists accumulated by the author, distribution lists obtained through business partners, and publicly available distribution lists. A total of 5,216 surveys were distributed and 106 were completed for an overall response rate of 2.03%. The findings from this research did not reveal that either type of QA organization better facilitated measurement with the exception of the suitability characteristic, which was the only one to show statistical significance. The study was exploratory and intended to examine if the twenty-one characteristics of software quality as defined in the International Standards Organization (ISO) 9126 standard were easier to measure in a central QA organization or decentralized QA organization. A large delta in the average difficulty rating may provide us with an area that should be further explored to determine the organizational trait or behavior that was responsible for the difference. A key aspect of originality in this research was the attempt to link the design of the QA organization to the measurement activities of a specific aspect of software quality. The thesis is further unique in that QA organizations are relatively new and there is not a lot of documented information or research on their function within an enterprise. There have been works published on the advantages and disadvantages of each organizational type. However, they do not address if the organizational design makes it easier to measure software quality. The available literature did suggest that there are some definite considerations that should not be ignored when building a QA organization. This study can be used by QA professionals, Managers and Graduate students conducting research in the relatively new field of QA. The notion of designing a QA organization that is customized to measure the metrics that are most important to your business is an enticing proposition to anyone who wants to minimize their operational costs, protect their revenue streams and deliver consistent quality. Increased knowledge in this area is essential to understanding the impact of an emerging area of IT on firms, industries and economies around the world. This research presents a current perspective on QA organizations that can be used by today's corporate managers who strive to institutionalize quality as a means of differentiating their business.

Abbreviations and Symbols

Abbreviations	Definitions
ARMS	Activities, Relationships, Metrics, Synergy
AUT	Application Under Test
BSI	British Standards Institute
CIS	Computerized Information Systems
CoE	Center of Excellence
DBA	Database Administrator
GUI	Graphical User Interface
IEEE	The Institute of Electrical and Electronics Engineers
ISO	International Standards Organization
IT	Information Technology
IV&V	Independent Verification and Validation
JAD	Joint Application Design
KTLO	Keep The Lights On
LOB	Line of Business
NASA	National Aeronautics and Space Association
QA	Quality Assurance
SDLC	Software Development Life Cycle
SEI	Software Engineering Institute
url	Universal resource locater

Glossary

Terms	Definitions
Black Box Testing	Testing based on an analysis of the specification of a piece of software without reference to its internal workings. The goal is to test how well the component conforms to the published requirements for the component.
Grey Box Testing	Testing software while already having some knowledge of its underlying code or logic. It implies more understanding of the internals of the program than black box testing, but less than white box testing.
ISO 9126	**A**n international standard for the evaluation of software quality. The fundamental objective of this standard is to address some of the well known human biases that can adversely affect the delivery and perception of a software development project. These biases include changing priorities after the start of a project or not having any clear definitions of "success." By clarifying, then agreeing on the project priorities and subsequently converting abstract priorities (compliance) to measurable values (output data can be validated against schema X with zero intervention), ISO 9126 tries to develop a common understanding of the project's objectives and goals.
IEEE Standard 12207	The quality assurance process is a process for providing adequate assurance that the software products and processes in the product life cycle conform to their specific requirements and adhere to their

Terms	Definitions
	established plans.
Regression Testing	Testing an application to ensure that existing functionality has not been broken as a result of making a change to the application.
Survey Collector	A SurveyMonkey term that describes the means by which survey responses are collected. A survey can have one or more collectors.
Test Case	A set of input values, execution preconditions, expected results and execution post conditions, developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.
Test Plan	A document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning.
Residual Defects	A software anomaly that goes undetected until it reaches the production environment. Residual defects are extremely costly to fix and in some cases can cause a production outage that prevents a company from conducting business.
Software Bug	A software bug is the common term used to describe an error, flaw, mistake, failure, or fault in a computer program that prevents it from behaving as intended (e.g., producing an incorrect or unexpected result). Most bugs arise from mistakes and errors made by people in either a program's source code or its design, and a few are caused by compilers producing incorrect code. A program that contains a large number of bugs, and/or bugs that seriously interfere with its functionality, is said to be buggy. Reports detailing bugs in a program are commonly known as bug reports, fault reports, problem reports, trouble reports, change requests, and so forth.
White Box Testing	Testing based on an analysis of the internal workings and structure of a piece of software. White box techniques include Branch Testing and Path Testing, also known as Structural Testing and Glass Box Testing.

Chapter 1

The New Era of Software Quality Assurance

The history of Quality Assurance (QA) sometimes referred to as Independent Verification and Validation (IV&V) can be traced back to the Atlas Missile Program in the late 1950s. The program officially launched what is now known as the U.S. Space Program and consisted of several large software development vendors, who had a tendency to give overly optimistic estimates of the software's development status. The implication was that the development contractors were not objective when measuring the true health of the project. This prompted the Program Manager to hire an independent software tester in hopes of getting a more accurate assessment (Nelson, 1979). Since then many studies have been done to support the assertion that projects with QA perform much better that projects without (Arthur & Nance, 2000; Wallace & Fuji, 1989).

QA has since evolved into a distinct discipline, but the best model for the organization must be determined to ensure that key characteristics of software quality can be measured in order to support and promote the mission of the business (Hayes, 2003). Although there have been works published on the pros and cons of various QA organizational models, research on which model makes measuring quality characteristics easier should be further explored if quality conscious organizations are to reap the benefits of their investment (Bowers, 2003; Hayes, 2003; Siggelkow & Levinthal A., 2003). Some Chief Information Officers (CIOs) believe that all of their problems can be solved by out-sourcing QA activities to domestic or overseas companies (Fajardo, 2008). However, in this model, the vendor is only an extension of the in-house QA model, so though this introduces new complexity, the underlying question as to what model makes measuring quality easier still remains.

Introduction to Research Topic

An effective QA organization must be able to measure the software quality characteristics that are most relevant to the success of their business in order to control the quality of software that is released into production. This is the first step in designing a QA organization that supports the overall mission of the business. This research strived to deepen the understanding of the relationship between the QA organizational model and the difficulty involved in measuring twenty-one, ISO 9126 characteristics of software quality in a centralized and decentralized QA organization. A large delta between the two models may provide us with an area that could be further explored to determine the role that organizational design and measurement played in contributing to the difficulty. This type of understanding takes us one step closer to the ultimate goal of trying to determine which type of model has better ARMS (Activities, Relevance, Metrics, and Synergies) to measure a certain characteristic of quality.

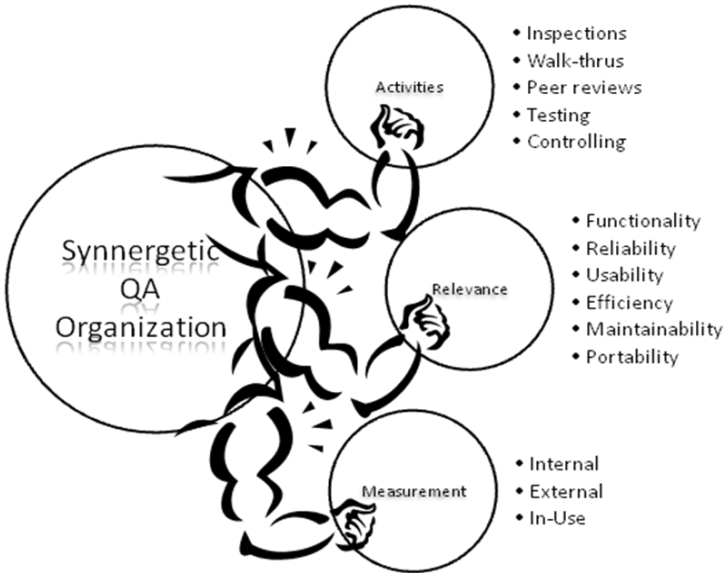


Figure 1-1 ARMS of the QA Organization

The acronym ARMS, describes the characteristics of a mature QA organization. That is, a mature QA organization must perform certain measurement activities, understand the activity's

relevance to a specific software quality characteristic, know the type of metrics to collect in order to support their business, and create synergy with adjacent organizations. The ARMS of an effective QA organization are shown in Figure 1-1.

Background and Purpose of Research

According to the US Department of Commerce, software bugs cost the US Economy $59.5 billion (Kasabe, 2005). Consider eBay who experienced an outage in 1999 that resulted in a loss of millions in revenue (Kasabe, 2005). Events like the eBay outage help to explain why software QA has emerged as a distinct discipline separate than that of software development. An inadequate QA process can result in the release of sub-standard software into production and, in some cases, cost companies millions of dollars in lost revenues. Less time is now spent convincing project managers that testing, a critical QA activity is necessary and more time is spent discussing whether or not the testing being performed is sufficient (Elfield, 2004). A defect in a mission critical application can potentially result in the loss of millions of dollars (Clark, 1999).

The magnitude of potential losses underscores the need to ensure that quality is an integral part of the software development life cycle (SDLC). The result is the creation of formal teams or departments dedicated to QA and charged with measuring and controlling software quality. In some cases, these teams are centralized and provide QA services to multiple lines of businesses (LOBs), while some organizations choose a more decentralized approach (Hayes, 2003). There are pros and cons to both approaches and both are capable of producing quality software. However, further research is needed to determine if a centralized or decentralized model has an effect that can be traced to measuring a specific software quality characteristic.

A robust QA process is designed to mitigate financial losses by detecting and removing defects as early in the SDLC as possible. This is accomplished using a variety of metrics to gauge the health of the software application. Gathering these metrics is rarely done in isolation and may require interacting with groups that are external to the QA organization. Furthermore, the types of activities involved in gathering these metrics will differ from company to company. The QA organization should enable these activities so that accurate measurements can be achieved for the quality characteristics that are most relevant to the mission of their business.

The purpose of this research was to explore the impact, if any, that the model of the QA organization has on measuring a specific software quality characteristic. It was an exploratory study that strove to deepen the understanding of whether or not the benefits inherent in centralized and decentralized QA organizations make one model better suited for measuring a specific characteristic of software quality. This knowledge may prove to be very valuable in helping managers to determine which model to choose so that the quality characteristics that are most relevant to their business can be easily measured. In other words, an organization that conducts commerce on its website may be primarily concerned with measuring the speed of their software to differentiate themselves in the market place. However, an organization that develops expert systems to determine the optimal level of inventory a company should carry may be less concerned with measuring software speed and more concerned with measuring software accuracy. In both cases, the QA organization should have ARMS that support the mission of their business.

Reasons for Carrying Out Research

The reason for carrying out this research was to add to the body of knowledge available to QA professionals, corporate managers, and students of QA. It sought to expand on the

literature that has documented the advantages offered by centralized and decentralized QA organizations to determine if they could potentially play a role in measuring a software quality characteristic. This study aimed to develop more data that could help companies build QA organizations that are consistent with their overall mission and vision.

Goals and Objectives of Research

The goal of this research was to design a study that involved practicing IT professionals in order to obtain a current perspective on how easy it is to measure the same software quality characteristics in centralized and decentralized QA organizations. The insight gained from the study will help to address the question of whether or not there is effect on measuring a quality characteristic by the QA organizational design, which is a question that requires additional data and research to answer with statistical significance. The study discusses the advantages of both organizations in Chapter 2 to provide some context and lay the groundwork for the findings discussion presented in Chapter 5. The overall objective of this research was to identify software quality characteristics that could potentially be affected by the design of the QA organization so that subsequent studies could be undertaken to determine the role that measurement played, if any, in contributing to software quality. This question is central to developing a QA organization with ARMS.

Propositions to be Examined or Tested

The key proposition explored by this research is whether or not measuring software quality can be affected by the design of the organization that is responsible for ensuring software quality – the QA organization. If it is true that one type of organizational design produces better quality across some characteristic of software quality, it may be possible to examine the behaviors exhibited and other aspects of that organization's design to determine if

measurement is the cause. The establishment of cause can then be used to design QA organizations that facilitate the measurement of quality metrics and are customized to their respective businesses.

Problem Definition

The problem statement addressed by this thesis was: The role, if any, that organizational structure plays on measuring software quality. Measuring software quality often involves activities that require collaborating with multiple groups within the organization. In some cases, the groups may be internal to the enterprise, while others may be external, such as a customer. Today's companies must design QA organizations that strengthen their presence in the marketplace (Siggelkow, 2003). This can only be accomplished by ensuring that each department, including the QA organization has access to the information it needs and is able to perform the activities that are required to measure the characteristics of software quality that are most relevant to their business (Westfall, 2005). In some cases, this may require decentralization, other cases may require centralization, and yet others may be able to achieve the same results in either model. The QA organization is not immune to this problem and is the focus of this research.

The Importance of This Research

QA has gained wide acceptance as an integral part of any, good SDLC. An efficient QA organization must be designed so that it facilitates the activities that are necessary to measure the characteristics of software quality that are pertinent to the success of the business. Such activities may include code inspections, requirements gathering and system monitoring (Florac, 1992). Many of today's institutions that rely on computerized information systems (CIS) now have formal QA organizations but are unaware if a centralized or decentralized

model is better suited to measure quality characteristics that are relevant to their business. It is challenging to manage and control software quality if it cannot be measured (Lubinsky, 2009). This inability to measure quality can cause a business to lose valuable customers and struggle to compete in the market place. In today's ever changing market landscape, firms must react to external forces to ensure that their activity configurations are internally consistent and appropriate for the current environment of the firm. This challenge is more acute when the competitive landscape changes, such that a new set of high-performing activities are required.

It has been noted that adaptive firms must maintain a balance of exploitation and exploration to stay ahead of the curve in the market place. To accomplish this, some firms may rely on centralized organizations while others rely on decentralized. Using the internet as an event that changed the competitive landscape for many companies we can illustrate the new activity choices that became available to companies. The Gap and Vanguard for example, viewed the internet as just another store and distribution channel respectively and chose to leverage its existing infrastructure. Bank One and Disney, on the other hand, chose to create web-based subsidiaries that operated under their own umbrella (Siggelkow & Levinthal A., 2003).

Similar challenges will be faced by the QA organization. Again, using the same event as above, we can illustrate the new activity choices that would become available to the QA organization. One option could be to form a QA organization that is dedicated to any business done via the internet. This option may require developing new processes, acquiring new skill sets, relocating and retraining existing personnel or creating a new QA methodology that supports the more sophisticated development processes that are often associated with newer technologies. Though this option may be very costly in the short term, it may eventually pay dividends as the new processes mature and QA practitioners become more and more intimate

with the Applications Under Test (AUTs). Furthermore, it may be more conducive to measuring specific quality metrics that are more relevant to that particular business such as efficiency and reliability, which are both included in the ISO 9126 quality model.

Another option that may become available to the QA organization can be seen when a firm merges with another firm due to an acquisition. This is a very common occurrence as many firms rely on acquisitions to grow their business and in most cases will require the integration of multiple systems. A company that follows this type of growth strategy must place keen focus on measuring different ISO 9126 quality characteristics like functionality and portability. It must ensure that all in-house applications are designed to be interoperable and easily integrated to adjacent applications and platforms. This may be best accomplished by maintaining a centralized QA organization that applies a rigid and consistent set of testing standards in order to properly measure these characteristics. A centralized model is also conducive to knowledge transfer from the acquired company and allows for the creation of a consistent set of documents that will aid the learning process. This enables the measurement of yet another ISO 9126 quality characteristic, that of learnability.

Now consider a decentralized group in this same scenario. Multiple QA teams are now charged with testing various integration points of numerous applications. This model poses many measurement challenges for the acquired firm including (Bowers, 2003):

- Review of documentation and the approval process may vary from team to team. This may make it challenging to measure software learnability.
- Testing artifacts are created with a wide variety of formats and templates, which may make it difficult to measure software functionality.
- Multiple teams must be briefed on conformance and compliance details, which leaves measurement open to interpretation.

- Multiple dependencies are created among test teams and development teams, which may cause measurements to be incomplete and fragmented.
- Testing can not start until all prerequisites are in place, which may extend the time that it takes to gather key quality metrics.
- The risk of duplicating some functions across testing teams is created, which may cause erroneous or inconsistent quality measurements to be taken.

This research was therefore important because it gives a deeper insight of QA organizational design as it relates to measuring relevant characteristics of software quality. This insight can be used to help firms design QA organizations that can measure software quality characteristics that support both their business and their growth strategies.

The Contemporary QA professional can refer to this study as their starting point to understand why certain applications in their portfolio may be suffering from certain software deficiencies. The understanding gained from this research can also be used by QA professionals and corporate managers alike to suggest areas where subtle changes to existing processes may make measuring the software quality characteristics that are critical to their business easier without disrupting normal operating procedures. In addition to benefiting the QA community this study may also be beneficial to researchers, who desire to learn more about human behavior in the QA organization. More specifically, the findings from this study can be analyzed to determine if decentralized and centralized QA organizations elicit certain human behaviors that are essential for measuring a certain characteristic of software quality. This type of research would be beneficial to a very broad community including QA professionals, corporate managers, sociologists, and anthropologists. Information from this research adds to the existing body of knowledge and can be leveraged by educators and future QA professionals

to improve software quality in their application portfolios. Finally, this research provides a platform on which future studies can be designed.

Chapter Summary

The realization that software defects found in production can cost a company millions of dollars has placed a greater emphasis on QA. The net effect is the creation of formal QA organizations by many of today's companies that rely on Computerized Information Systems (CIS) to support their mission critical business processes. However, a QA organization that is not designed with the overall mission of the company in mind may find it difficult to measure the characteristics of quality that are most important to their business. The challenge then becomes determining the best model for the organization to ensure that specific quality characteristics can be measured in order to support the overall mission of the organization (Hayes, 2003).

Although there have been works published on the pros and cons of various QA models (Hayes, 2003; Topping, 2009), research on which QA model is best suited to measure the quality characteristics that are most relevant to a type of business should be further explored if quality conscious organizations are to reap the benefits of their investment. An organization that conducts commerce on its website should have a QA model that facilitates the measurement of application speed, whereas an organization that produces expert systems that may include very sophisticated mathematical operations should have a QA model that facilitates the measurement of accuracy. This is the first step in designing a QA organization with ARMS.

The overall objective of this research was to contribute to the body of knowledge that currently exists on QA organizations and their affect, if any, on measuring software quality. Chapter 2 presents the literature currently available on the advantages and disadvantages of

centralized and decentralized QA organizations. The literature stops short of addressing whether or not the advantages of either organization make it easier to measure software quality, which is the area that this thesis strove to expand upon. Chapters 3 through 5 will address key aspects of the research process and design. The thesis culminates with the conclusions of the research which are presented in Chapter 6. Finally, this study is important to QA professionals, researchers, educators, and students of QA who seek to improve the quality of software and gain valuable insights into the QA organization and the software development process.

Chapter 2

Literature Review

An extensive review for available literature on the design of the QA organization and its effect on measuring software quality was performed in support of this research. The review did not uncover any previous research or publishing that specifically addressed the problem statement described in this thesis. However, the review did uncover literature that supports various components of the problem statement, such as the importance of measurement in the field of QA. The literature also lays the groundwork for examining the impact that the advantages inherent in centralized and decentralized QA organizations have on the organization's ability to measure aspects of software quality that are consistent with the overall mission of their business.

The Institute of Electrical and Electronics Engineers (IEEE) Standard 12207 defines QA in this way: "The quality assurance process is a process for providing adequate assurance that the software products and processes in the product life cycle conform to their specific requirements and adhere to their established plans" (Feldman, 2005). Webopedia, the online encyclopedia dedicated to computer technology, defines software quality as: "Abbreviated as SQA, and also called software assurance, it is a level of confidence that software is free from vulnerabilities, either intentionally designed into the software or inserted at anytime during its lifecycle, and that the software functions in the intended manner" (Webopedia, 2009). Yet another source, the National Aeronautics and Space Administration (NASA), defines QA as: "The function of software quality that assures that the standards, processes, and procedures are appropriate for the project and are correctly implemented" (NASA, 2009).

Feldman (2005) points out several key aspects of the IEEE definition that represent a common theme expressed by all three aforementioned sources in his paper Quality Assurance:

Much More Than Testing. Firstly, the multiple use of the word process suggests that QA is also very much about a method and approach. Secondly, Feldman (2005) states that QA is about providing assurance to various stakeholders that the software will in fact, perform as expected. Finally and most importantly, Feldman (2005) points out that QA is not just a philosophy, but is a very measurable science. This opinion is supported in NASA's (2009) definition which states that standards are "correctly implemented" suggesting a measurable characteristic to software QA. The concept of measurement can also be seen in Webopedia's definition of QA which states that the software is "free from vulnerabilities" and "functions" as intended. All three definitions provide a good working definition of the measurement aspect of software QA. However, none of the aforementioned definitions of QA provide any insight into the type of QA organization that is required to properly measure software quality, nor could any such definition be found. It is the area of software quality measurement as it relates to QA organizational design that this research explored.

The Centralized QA Organization

Literature was also available on the advantages and disadvantages of centralized and decentralized QA organizations. One of the earliest papers written on the advantages and disadvantages of centralized and decentralized QA organizations states that there are several advantages to a centralized QA organization including standardization of processes, better demand management and reduction in redundant activities (Hayes, 2003; Podlogar, 2002). However, the paper suggests that the centralized model is not without its challenges. Unlike the decentralized model, centralization reduces the need for subject matter expertise which Hayes states is due in large part to the pooling of personnel resources (2003). Hayes goes on to imply that while pooling of resources is more suitable for demand management, different personnel may be assigned to support a different application, which does not allow the practitioner to

develop deep knowledge about the application under test (AUT) or cultivate relationships (2003). Topping later supports Hayes' opinion by stating that this model may create a "just another client" mentality where the client has no face (2009).

The literature also suggests that the biggest benefit of a centralized QA model is specialization (Hayes, 2003; Podlogar, 2002). QA professionals in a centralized QA organization tend to have more defined career paths and develop deeper skill sets. Centralized QA groups may also be better able to establish and follow QA processes (Topping, 2009). The implication here is that formalized processes may make access to application information and other external groups easier for a centralized organization. Centralized groups also create certain operating efficiencies by reducing redundancies that are likely present in a decentralized model. The available literature on centralized QA organizations does not, however, make any assertions as to how or whether the benefits gained by this model, facilitate the measurement of key software quality characteristics.

The documented advantages of a centralized organization, however, all fall shy of mapping an advantage to measuring a specific characteristic of software quality. For example, the literature does not allow the researcher to make the assertion that since QA practitioners in a centralized QA organization tend to have more specialized training and may better understand formal QA techniques that they would therefore, be more adept at measuring some characteristics of software quality (Hayes, 2003). Furthermore, the literature does not address how or whether the specialized training facilitates better measurement of any quality sub-characteristic. As such, there are gaps in the literature as it pertains to centralized QA organizational design and measuring certain characteristics of quality.

The Decentralized QA Organization

Early literature on decentralized QA organizations suggests that one of the key benefits available in this model is that it is more conducive to partnering with the development team as interactions tend to be more frequent (Hayes, 2003). This creates a safe environment where the developers are less likely to be threatened when asked questions about defects found in their code and are more willing to participate in the defect resolution process. Historically, this relationship has been contentious and can lead to defects that are intentionally overlooked to avoid conflicts with the development organization. The reasons for the contention can generally be attributed to one or more of the following reasons (Ogale, 2005):

- Programmer and Tester have differing opinions on functionality that is required
- Programmer and Tester are not properly trained on the use of the software
- Poor estimation of time needed to properly develop and test the software

A decentralized QA model helps to address these challenges by cultivating relationships between the development and testing organizations.

In her article Organizing Localization in Large Companies, Suzanne Topping (2009) states about a decentralized team that "The close connection to project teams tends to result in development of esprit de corps. Communication of critical issues, product changes and risks is faster and more complete. Since the localization staff is accountable to the people managing product development, their careers have a more direct link to results. The clients they serve are more than "just another customer. Though the context here deals with localization projects, the benefits identified in Topping's (2009) article are very much the same as those stated by Hayes (2003). Another key advantage of this model is that subject matter expertise is increased which leads to better testing coverage and effectiveness (Hayes, 2003).

Much like the literature on centralized QA organizations, the literature on decentralized QA organizations focused on how each QA organization functions internally as well as with other organizations. The literature also emphasized certain advantages present in decentralized QA organizations. However, the literature does not address if the decentralized QA model is more conducive to measuring certain aspects of software quality. Therefore, it remains to be seen if a decentralized QA model, by addressing the aforementioned challenges, facilitates the measurement of software quality characteristics and if so, which ones. Neither Hayes nor Topping addresses this aspect of organizational structure in their writings. However, they both lay the groundwork for examining the impact that the advantages inherent in each model have on an organization's ability to measure those characteristics of software quality that can support the overall mission of their business.

Comparing the Two Models

Table 2-1 presents a summarized view of how the literature on the advantages offered by either type of QA organization could be used to interpret the findings from this research. For example, the table indicates that a quality characteristic that can be better measured by a centralized QA organization can be interpreted to require a higher degree of QA specialization, involve a higher degree of process, require a lower degree of application expertise or does not require a high degree of interaction with external groups. The opposite interpretation would be true for a characteristic that can be better measured by a decentralized QA organization. These interpretations will be used as the basis for the findings discussion in Chapter 5.

Table 2-1 Summary of Organizational Advantages

Advantage	Degree	Description	C	D
Specialization	**High**	Sub-characteristics that require a high degree of QA specialization to measure.	+	-
	Low	Sub-characteristics that require only a general knowledge of the subject matter, which is	-	+

Advantage	Degree	Description	C	D
		generally available to all project team members.		
Process	**High**	Sub-characteristics that rely on processes to receive information about software applications.	+	-
	Low	Sub-characteristics that receive information using more informal processes.	-	+
Expertise	**High**	Sub-characteristics that require in-depth knowledge of the application.	-	+
	Low	Sub-characteristics that require general knowledge of the entire application portfolio or knowledge of the activities that occur across all LOBs.	+	-
Relationship	**High**	Sub-characteristics that require the constant interaction or direct interaction with individuals or groups outside of the QA organization.	-	+
	Low	Sub-characteristics that can be measured without the direct interaction or direct interaction with individuals or groups outside of the QA organization.	+	-

Chapter Summary

An extensive literature review on centralized and decentralized QA organizations and their ability to facilitate the measurement of software quality did not uncover any relevant works that could be used in support of this thesis. However, there have been works published on the benefits of centralized and decentralized QA organizations that provide some context that can be used to help explain the difference in difficulty measuring certain aspects of software quality. Benefits may include better working relationships, specialized skills, more formal processes, and in-depth application expertise.

The literature on the advantages and disadvantages of centralized and decentralized QA organizations all fall short of addressing whether or not either type of QA organization is better for measuring certain software quality characteristics. The organizational design and its impact on measuring software quality is the area that this research attempted to explore. Chapter 3

discusses the key elements of the research methodology and design, including the research variables, tools, and hypothesis that was tested.

Chapter 3

Research Methodology and Design

This chapter examines the research variables and the quantitative methods used to determine the change, if any, to the dependent variable and whether or not there was statistical significance. It also discusses the steps taken to ensure that the data collected was reliable. Figure 3-1 shows the process that this study used to collect and analyze the research data.

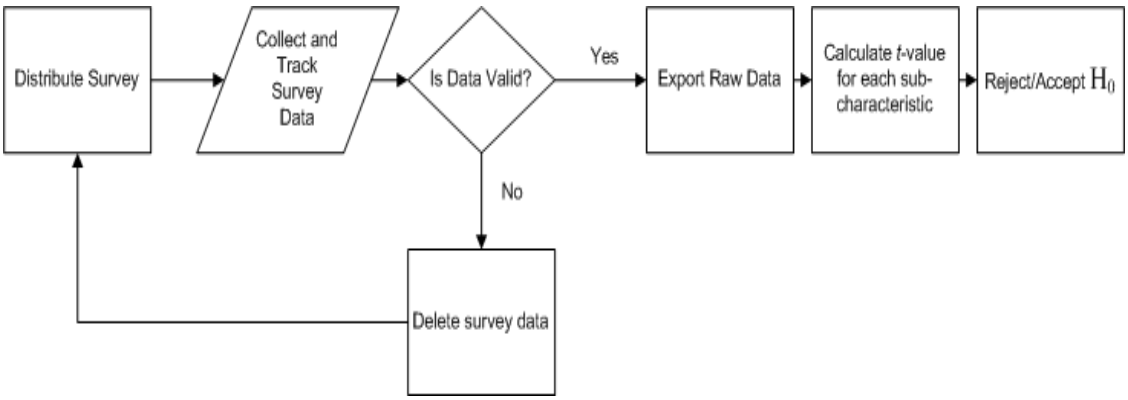


Figure 3-1 - Research Design Flow

Theoretical Framework

Figure 3-2 depicts the theoretical framework of this research.

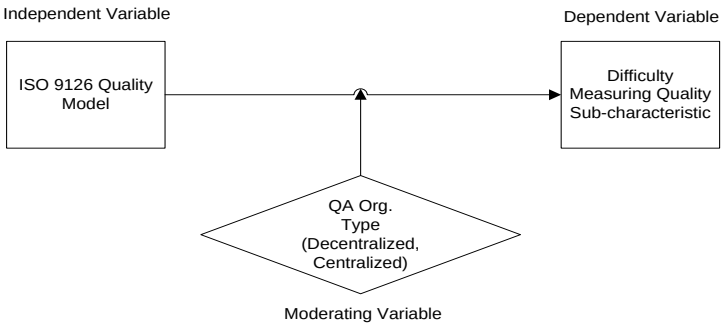


Figure 3-2 Theoretical Framework

The independent variable of this study was the ISO 9126 software quality model which has six characteristics and twenty-one sub-characteristics, the moderating variable was the QA model, which could only have two values, centralized and decentralized, and the dependent variable under study was the difficulty measuring a software quality sub-characteristic. Note that changes to the dependent variable do not imply improvement to a quality sub-characteristic. The only implication is that one organization may better facilitate some of the activities that are required to measure that specific aspect of quality.

Problem Statement

The problem statement addressed by this thesis was: The role, if any, that organizational structure plays on measuring software quality. Measuring software quality often involves activities that require collaborating with multiple groups within the organization. In some cases, the groups may be internal to the enterprise, while others may be external, such as a customer. Today's companies must design QA organizations that strengthen their presence in the marketplace (Siggelkow, 2003). This can only be accomplished by ensuring that each department, including the QA organization has access to the information it needs and is able to perform the activities that are required to measure the characteristics of software quality that are most relevant to their business (Westfall, 2005). In some cases, this may require decentralization, other cases may require centralization, and yet others may be able to achieve the same results in either model. The QA organization is not immune to this problem and is the focus of this research.

Hypothesis Tested

The intent of this research was to understand the effect, if any, that a decentralized and centralized QA organization had on measuring a specific ISO 9126 software quality

characteristic. Key to this understanding was ascertaining the degree of difficulty obtaining measurement on each of the twenty-one software quality sub-characteristics identified in the ISO 9126 software quality model. Difficulty measuring a certain sub-characteristic may indicate that the QA organization is not designed to measure a sub-characteristic that may be relevant to its business. A characteristic that cannot be measured or is difficult to measure will be difficult to control, and therefore, difficult to optimize. Optimized, within the context of this research, describes the degree to which a sub-characteristic supports the mission of the business. For example, an organization that conducts commerce on its website may be primarily concerned with the speed of their software to differentiate themselves in the market place. However, an organization that produces expert systems to determine the optimal level of inventory a company should carry may be less concerned with application speed and more concerned with accuracy. The study tested the following hypothesis:

> H_0: There is no difference in difficulty to measure any ISO 9126 software quality sub-characteristic between a centralized and decentralized QA organization.

The ISO 9126 Software Quality Model

The independent variable under study was the ISO 9126 standard for software quality, which is part of the ISO 9000 family of standards that addresses software quality. The ISO Model was selected primarily for two reasons. Firstly, the ISO model is more concerned with the way an organization performs it works and not the outcomes (Patton, 2006). The quality model leaves it up to each individual organization to determine the quality levels that are appropriate for their respective organizations. Secondly, the ISO quality model only dictates the process requirements and not the implementation. It does not matter how large or small your organization is, it is up to it to implement processes that are suited to their individual businesses (Patton, 2006).

The ISO 9000 standards can be dated back to 1959 when Mil-Q-9858a, the first quality standard for military procurement was established (Emerson, 2006). However, it was not until the late 1970s when the British Standards Institute (BSI) published BS 5750, a series of standards for use by manufacturing companies, that the ISO eventually adopted BS 5750 and published it globally under the name ISO 9000 (Emerson, 2006). Initially, ISO 9000 was used only by the manufacturing industry, but was soon adopted by many other types of businesses (Emerson, 2006). More than 100 countries have now adopted ISO 9000 as their national quality assurance standard (Emerson, 2006).

The ISO 9126 quality model defines software quality using six characteristics and twenty-one sub-characteristics that are intended to be exhaustive. The model measures quality by gathering internal, external and quality in-use metrics across all quality characteristics. The QA organization must enable the activities involved in collecting all three types of metrics, such as the interaction between the development and QA organizations during the defect resolution process or the review of software requirements before they are finalized. Figure 3-3 portrays the relation among the different metrics in the ISO 9126 quality standard (Basu, 2005).

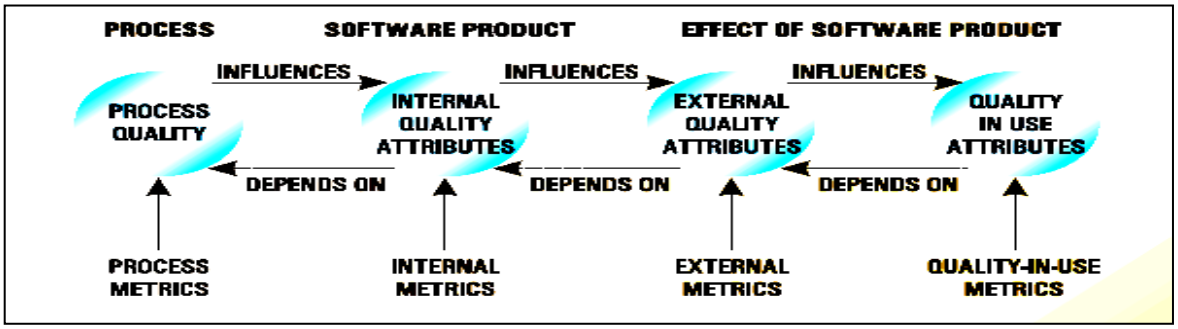


Figure 3-3 ISO 9126 Metric Relationships

Internal metrics, sometime referred to as static metrics, are those metrics that do not require the execution of the actual source code (Basu, 2005). There are primarily three project activities necessary to gather these metrics, inspections, walk-throughs, and peer reviews. All three activities involve the review of various project artifacts, such as a project plan, testing strategy, or requirements document. They are typically performed by a team consisting of individuals who play various roles (Ganssle, 2001). The output from this review may result in multiple modifications to the project plan, in which it is important for the QA organization to have a voice, since it may affect the time allotted to properly test the application. Failure to do so may result in the QA organization's inability to measure certain quality sub-characteristics.

External metrics, sometimes referred to as dynamic metrics, are those metrics that require execution of the source code (Basu, 2005). The key activities involved in gathering external metrics are commonly referred to as product testing or test execution (Florac, 1992). In this phase of the SDLC, test cases that are typically derived from a set of system requirements created earlier in the SDLC are executed. Highly skilled QA practitioners may employ several white box, black box, and grey box techniques to determine the appropriate number and types of test cases that should be executed to increase the likelihood that defects will be found, while avoiding the point of diminishing returns (Bach, 1998). Knowledge of these techniques become extremely important as the project draws to an end and deadlines approach.

Test execution is intended to simulate application functionality in a test environment, and compare the expected results to the actual results achieved by executing the test case. When an anomaly is found, another key activity, the defect management process that resembles Figure 3-4 is followed (Nindel-Edwards & Steinke, 2006).

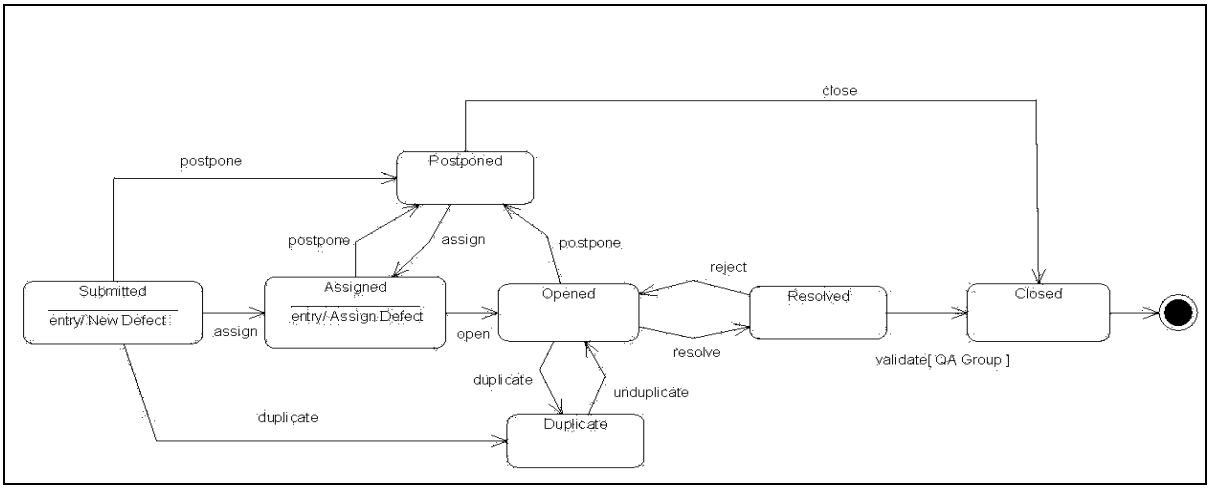


Figure 3-4 Typical Defect Management Process

During the defect management process, a defect or bug report is created that typically includes the steps taken to recreate the bug, a severity rating, a priority rating and any other information that may be useful to the bug resolution process (Florac, 1992). The bug report is then sent to the development organization, and the defect resolution process begins. Resolving defects is the iterative process and requires excellent communication skills on the part of the QA practitioner, who must describe the bug in detail to the developer, who is then charged with making any changes to the source code.

Once the changes have been made to the source code, QA must retest the source code to validate the bug's resolution. If the bug is successfully retested, the corrected source code is then promoted into the production environment. Otherwise, the bug report is updated with any additional information and is returned to the developer for further resolution and the bug continues to live until it has been successfully retested and promoted into production. Relationships can become very important in this process, as the developers are more willing to work with members of the QA team with whom they have developed a rapport. A contentious relationship between the development and QA organizations may lead to bugs that are

intentionally overlooked by the QA team due to their reluctance to partner with their development counterparts (Ogale, 2005). Conversely, the development team may take more liberties when interpreting software requirements and writing source code, essentially bullying the QA organization into accepting sub-standard code. The result can be an inability to properly measure key external metrics that are included in the ISO 9126 quality model, which may lead to releasing code into production that does not properly support the mission of the business.

Finally, quality in-use metrics are metrics that can only be gathered once the software has been released into production (Basu, 2005). Therefore, by definition, these metrics require execution of the actual source code in a real-world environment. Key activities involved in collecting these metrics may include, monitoring hard copies of printouts before delivering to customers, monitoring hardware resources (i.e. memory consumption, CPU utilization, etc.), reviewing customer satisfaction reports, tracking customer support tickets and monitoring sales activity (Florac, 1992). The gathering of this information will more than likely involve several different groups. For example, sales information will probably come from the sales team, whereas customer service ticket information will probably come from the helpdesk organization. This information is vital to the QA organization so that root cause analysis can be performed and modifications to test plans can be made to reduce the number of residual defects.

In this instance the QA organization must have ARMS that extend into the customer support and sales organizations. These types of metrics may be of particular importance for companies who rely on customer retention for their existence. The inability to capture these quality metrics could decrease customer satisfaction and cause customers to seek other options. The net result can be the loss of much needed revenue streams. Therefore, the ISO 9126

quality standard emphasizes the need for active participation and collaboration by members of the QA organization and adjacent groups throughout the entire SDLC. Table 3-1 shows the quality characteristics and sub-characteristics that are included in the ISO 9126 quality standard and are an integral part of software quality. In spite of its wide acceptance, the model does not adequately address the nuances of a centralized and decentralized QA organization to determine if some measurement activities can be better performed in one or the other. It is this area that this research sought to focus and bring to the fore for QA professionals engaged in the software development process.

Table 3-1 ISO 9126 -1 Software Quality Model

Characteristics	Sub-characteristics	Definitions
Functionality	Suitability	This is the essential Functionality characteristic and refers to the appropriateness (to specification) of the functions of the software.
	Accurateness	This refers to the correctness of the functions, an ATM may provide a cash dispensing function but is the amount correct?
	Interoperability	A given software component or system does not typically function in isolation. This sub-characteristic concerns the ability of a software component to interact with other components or systems.
	Compliance	Where appropriate certain industry (or government) laws and guidelines need to be complied with, i.e. SOX. This sub-characteristic addresses the compliant capability of software.
	Security	This sub-characteristic relates to unauthorized access to the software functions.
Reliability	Maturity	This sub-characteristic concerns frequency of failure of the software.
	Fault tolerance	The ability of software to withstand (and recover) from component, or environmental, failure.
	Recoverability	Ability to bring back a failed system to full operation, including data and network connections.
Usability	Understandability	Determines the ease of which the systems functions can be understood, relates to user mental models in Human Computer Interaction methods.
	Learnability	Learning effort for different users, i.e. novice, expert, casual etc.
	Operability	Ability of the software to be easily operated by a given user in a given environment.
Efficiency	Time behavior	Characterizes response times for a given thru put, i.e. transaction rate.

Characteristics	Sub-characteristics	Definitions
	Resource behavior	Characterizes resources used, i.e. memory, cpu, disk and network usage.
Maintainability	Analyzability	Characterizes the ability to identify the root cause of a failure within the software.
	Changeability	Characterizes the amount of effort to change a system.
	Stability	Characterizes the sensitivity to change of a given system that is the negative impact that may be caused by system changes.
	Testability	Characterizes the effort needed to verify (test) a system change.
Portability	Adaptability	Characterizes the ability of the system to change to new specifications or operating environments.
	Installability	Characterizes the effort required to install the software.
	Conformance	Similar to compliance for functionality, but this characteristic relates to portability. One example would be Open SQL conformance which relates to portability of database used.
	Replaceability	Characterizes the *plug and play* aspect of software components, that is how easy is it to exchange a given software component within a specified environment.

The QA Organizational Models

The key distinction between a centralized and decentralized QA organization, as with most other organizations is the general degree to which delegation exists (Certo, 2000). In a centralized QA organization, decisions are made by a single governing body and delegation is minimized such that subunits have no autonomy or decision making authority (Certo, 2000). Consider a centralized QA organization that may have separate groups dedicated to the various LOBs, all decisions are still made by the same governing body. However, in a decentralized QA organization, the degree of delegation is greater so that each LOB has the ability to establish its own processes, standards, and procedures.

Figures 3-5 and 3-6 show both QA organizational models.

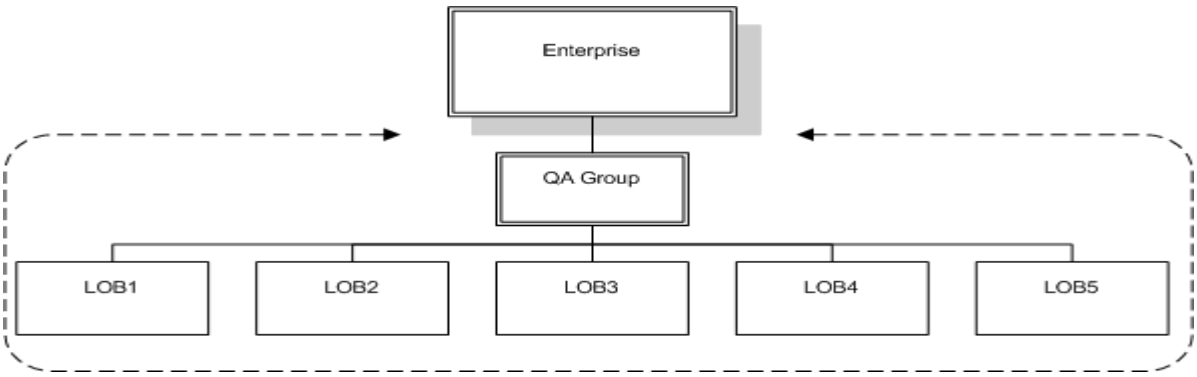


Figure 3-5 Centralized QA Organization

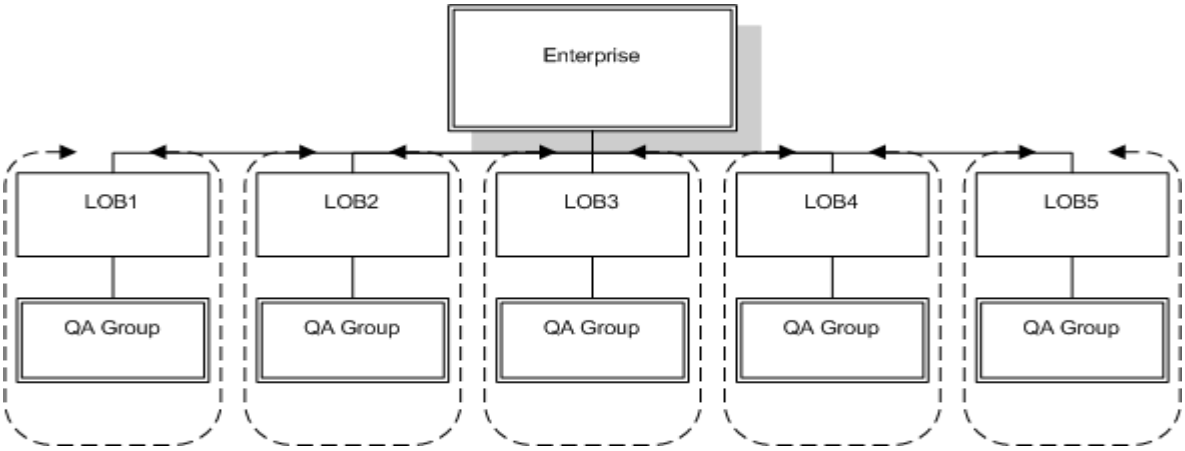


Figure 3-6 Decentralized QA Organization

In a centralized QA model, a single QA group services all LOBs in the enterprise. The centralized model is sometimes referred to as a QA center of excellence (CoE) and has to be prepared to address the demand for services across the enterprise so that QA resources can be provisioned and allocated to LOB projects as they are initiated. In a decentralized model, each LOB is serviced by its own, dedicated QA group. Each QA group may or may not adhere to the same QA practices and may be governed by different processes.

The Measurement Framework

As stated earlier, the ISO 9126 quality model identifies three types of metrics, internal, external and in-use. Each type of measurement may occur during different stages of the SDLC

and are intended to detect defects in different project artifacts. Table 3-2 shows the measurement framework established by the Software Engineering Institute (SEI) (Florac, 1992) and has been modified to show the relationships to the metrics set forth in the ISO 9126 quality standard. The table is not intended to be a comprehensive listing of all measurement activities, but to provide a framework that helps determine what type of activities can be performed to measure each type of metric. Additional activities could include monitoring and joint application design sessions (JADs) to name a couple.

Table 3-2 Measurement Framework

Activities	Project Deliverable	ISO 9126 Metric Type
Product Synthesis	Requirements Specifications Design Specifications Source Code User Publications Test Procedures	Internal Metrics
Inspections	Requirements Specifications Design Specifications Source Code User Publications Test Procedures	Internal Metrics
Formal Reviews	Requirements Specifications Design Specifications Implementation Installation	Internal Metrics
Testing	Modules Components Products Systems User Publications Installation Procedures	External Metrics
Customer Service	Installation procedures Operating procedures Maintenance updates Support documents	In-Use Metrics

The measurement framework for both a centralized and decentralized QA organization are the same and therefore, serves as a guideline when determining the difficulty measuring certain software quality characteristics. For example, when determining how difficult it is to

measure accurateness, considerations may include QA's involvement in various activities (i.e. product synthesis, Inspections, etc.), as well as the type of metrics that are collected (i.e. internal, external, in-use, combination.), and the deliverable that is being acted on (i.e. requirement Specifications, design Specifications, etc.). The measurement framework also helps to determine which stakeholders need to be involved so that the appropriate organizations can be engaged to participate in the activity. This understanding is the first step in ensuring that the QA organization has ARMS.

The activities shown in Table 3-2 are those identified by Florac (1992) as part of the software QA process without regard to organizational structure. The challenge is to determine which type of QA organization, centralized or decentralized, creates an environment where QA involvement in these activities is seamless and therefore, enables the measurement of certain quality characteristics. It is important to note, that Table 3-2 also implies that if quality characteristics can be measured by multiple types of metrics, measurement activities will not only happen at different phases of the SDLC, but will also require interacting with other groups in the enterprise.

Analysis Framework

Included as part of the analysis discussion for each quality sub-characteristic is an interpretation of which potential organizational advantages may explain the findings. The data for both QA organizations was consolidated into a single table. Each table is grouped by ISO quality characteristic and contains the measurements for all related sub-characteristics. Each Table contains 11 columns of information on each sub-characteristic that are defined as follows:

- Column 1: Sample mean for the centralized QA organization.

- Column 2: Sample mean for the decentralized QA organization
- Column 3: Variance for centralized QA organization
- Column 4: Variance for decentralized QA organization
- Column 5: Standard deviation for centralized QA organization
- Column 6: Standard deviation for decentralized QA organization
- Column 7: Calculated *t* value for centralized QA organization
- Column 8: Calculated *t* -value for decentralized QA organization
- Column 9: Statistical significance measured against a *t*-value of 1.96. This value is binary and can either be a yes or a no. A yes suggests that the null hypothesis should be rejected; a no suggests that there is not enough information to reject the null hypothesis.
- Column10: The direction of impact for the centralized organization. This value can be negative or positive as indicated by a plus or minus sign. It is calculated by subtracting the sample mean of the decentralized QA organization from the sample mean of the centralized QA organization. If the result is negative it suggests that a decentralized organization is more appropriate for measuring this aspect of quality and vice versa.
- Column 11: The direction of impact for the decentralized organization. This value can be negative or positive as indicated by a plus or minus sign. It is calculated by subtracting the sample mean of the decentralized QA organization from the sample mean of the centralized QA organization. If the result is negative it suggests that a

centralized organization is more appropriate for measuring this aspect of quality and vice versa.

Note that there are many activities that companies can perform to collect internal, external and quality in-use metrics. In discussing each quality characteristic, the activities mentioned are only intended to be examples that aid in interpreting the results and do not imply that these are the only activities that can be performed to collect any type of metric.

Calculating Difficulty

The act of measuring anything requires the clear understanding of how to measure and what to measure (Certo, 2000). For example, if we are considering how to measure a janitor's performance, we must first define the units of measure. In the case of a janitor, the units of measure may include floors swept and windows washed. The janitor who washed more windows and swept more floors would be considered a higher performer. In the case of this research, we were measuring the change, if any, in the difficulty measuring software quality characteristics between a centralized and decentralized QA organization. The unit of measure then, was the delta between the difficulty rating measuring each quality sub-characteristic in a centralized QA organization and the difficulty rating measuring each quality sub-characteristic in a decentralized QA organization. Explicitly stated, changed was measured by the difficulty measuring software quality characteristics in a centralized organization minus the difficulty measuring software quality characteristics in a decentralized organization and thus, the greater the delta, the greater the impact to the quality characteristic.

This research generated two scores for each ISO 9126 quality sub-characteristic, one for a centralized QA organization, the other for a decentralized QA organization. The score was based on a ten-point rating scale designed to measure how difficult it is to measure each of

the twenty-one sub-characteristics. Survey choices 1 thru 10 were weighted accordingly. That is, a choice of one had a weight of one; a choice of two had a weight of two, and so on. Each question also had a "Do not know" option that was assigned a weight of zero. The survey was symmetrical in that the respondent was asked to respond to the same questions twice, once for centralized QA organization, and once for a decentralized QA organization. In both cases, the rating average and response count was calculated and tracked for each sub-characteristic. Response count was a simple tally of how many people responded to the question and was used to determine the rating average.

The rating average for each sub-characteristic was calculated by the formula:

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

These calculations yielded two real numbers between 1 and 10 rounded to the nearest hundredth for each quality sub-characteristic. One number was for a centralized QA organization and the other was for the decentralized QA organization. The difference (*D*) on each sub-characteristic was then calculated using the formula:

$$\left| D = \frac{1}{n}\sum_{i=1}^{n} x_{i\,(Centralized)} - \frac{1}{n}\sum_{i=1}^{n} x_{i\,(Decentralized)} \right|$$

The formula calculated the absolute value of the delta to determine the difference in the degree of difficulty measuring a software quality sub-characteristic in a centralized and decentralized QA organization. By removing the absolute value symbols we are left with a calculation that determines the polarity (*P*) of change:

$$P = \frac{1}{n}\sum_{i=1}^{n} x_{i\ (Centralized)} - \frac{1}{n}\sum_{i=1}^{n} x_{i\ (Decentralized)}$$

The polarity calculation determined if the change was positive or negative. In other words, it determined if it would be easier to measure a software quality sub-characteristic in a centralized or decentralized QA organization. The two key pieces of information thus produced for each quality sub-characteristic by the above calculations were change and direction.

Test for Statistical Significance

The statistical significance of the difference in sample means was determined by using a two-tailed, t-test shown by the equation below:

$$t = \frac{\left| \bar{X}_{Centralized} - \bar{X}_{Decentralized} \right|}{\sqrt{\frac{s^2_{Centralized}}{n_{Centralized}} + \frac{s^2_{Decentralized}}{n_{Decentralized}}}}$$

Where *s* was the centralized or decentralized sample variance calculated by:

$$s^2 = \frac{\sum (x - \bar{X})^2}{n - 1}$$

A *t* -test was deemed appropriate given the nature of the variables that were collected and analyzed. That is, the moderating variable was categorical and the independent variable was quantitative. A two-tailed *t*-test was chosen as the null hypothesis only stated that there was no difference between the two means and did not stipulate the direction of the change. However,

since there was sufficient data to determine direction, this information was included in this thesis so that further analyses could be performed.

Research Tools

The study used an online survey as it was deemed an appropriate means of data collection given the technical background of the target audience. The primary data collection tool used to distribute the survey and collect responses was SurveyMonkey (www.surveymonkey.com), an online tool created in 1999 that must be subscribed to and is now used by more than 80% of the Fortune 100 to collect the information that management needs to make informed decisions. SurveyMonkey created unique identifiers that were attached to each e-mail address, so that each survey recipient could only respond to a single survey. This functionality was very useful in ensuring that multiple surveys were not completed by the same individual, which further ensured that the data collected was valid.

The primary tools used to perform data analyses were MS Excel, a variance calculator, and a two-tailed, *t*-value calculator. Data from SurveyMonkey was exported into MS Excel where it could be massaged into various formats and charts that could be edited. However, due to the time and complexity involved with manipulating the raw data, two ready-made calculators were used to calculate the variances, standard deviations and the resulting *t*-values for each sample set. Raw data from SurveyMonkey was first entered into a variance calculator provided by GroundwaterSoftware.com (www.groundwatersoftware.com). For each QA organization, difficulty ratings data for a specific quality characteristic was entered and the calculator did the rest. As each value was added, the calculator made real-time updates that yielded the current sample size, mean, variance and standard deviation. The calculator was also very helpful in verifying some of the information that was produced by SurveyMonkey, such as sample mean.

The output from the variance calculator provided by GroundwaterSoftware.com was then used as input to the online *t*-value calculator provided by Dimension Research, Inc (www.dimensionresearch.com). The two-tailed, *t*-test calculator asked for the mean, sample size, and standard deviation of each sample set as input. It also asked the user to select a confidence level from a pre-populated drop-down box. Upon entering all of the required information, a calculate button was then pushed which yielded the *t*-value, degrees of freedom, an actual confidence level, and whether or not the change was statistically significant.

The combination of SurveyMonkey, the variance calculator provided by GroundwaterSoftware.com, and the two-tailed, online *t*-value calculator provided by Dimension Research, Inc. not only helped to remove the mystery of complicated statistical calculations, but also helped to save time that was then put toward more in-depth analysis.

Data Reliability and Validity

Every completed survey was reviewed for completeness and to identify any trends that may suggest that the survey was completed in an unconstructive way (i.e. a rating of 5 for every sub-characteristic). No such anomalies were found in the process of this review. Furthermore, SurveyMonkey only uses completed surveys for any of its calculations, which ensures that all calculations were consistent for all survey questions. Data validity was also inherently tested in the analysis process which required re-entering all survey data into various calculators, which generated the same calculations made by SurveyMonkey. In all cases the results were the same.

Research Assumptions

The following section describes the assumptions that must be acknowledged in the design of this research.

- All survey respondents completed the survey to the best of their ability increasing the reliability of the data.
- A response of "Do not know" implied that the metric is not relevant to the respondent's business and is therefore, not collected. As such, a weight of zero was assigned for this response when calculating sample means and other key statistics.
- All respondents had experience working with both types of QA organizations.
- There was no inter-dependency on a respondents rating between a centralized and decentralized QA organization. In other words, respondents' rating of difficulty obtaining measurement in a centralized QA organization did not affect how the same respondents rate a decentralized QA organization and vice-versa. Therefore, in determining degrees of freedom, survey respondents can be treated as two independent populations.
- Survey recipients were more likely to complete an online survey than a paper-based survey.
- Three solicitations were sufficient to determine whether or not a survey recipient would have responded to the survey.
- Difficulty measuring a software quality characteristic implied that the QA organization cannot properly perform the activities, or is not included in the activities, which are necessary to obtain measurement.
- Surveying all key stakeholders captured a more accurate perspective than surveying a single set of stakeholders (i.e. QA managers, developers, etc.). The net result was more reliable data.

Research Limitations

In addition to the assumptions included in this research, there were a few limitations that were known going into this research. One known limitation was the immaturity of the QA profession. In fact, QA was considered an afterthought by some as recently as 2007 (Portnov, 2007). QA is still very much an evolving profession, which makes finding qualified survey participants very challenging and will undoubtedly have an effect on response rate.

Another known limitation was the complexity of the ISO 9126 software quality standard. The standard contains terms that may be foreign to some survey recipients and may require that they review reference materials to familiarize themselves with the standard and other relevant information. Such complexity may discourage survey recipients from completing the survey due to the unreasonable time commitment, again having a negative impact on survey response rate. Thus, a critical success factor is finding a large enough respondent population to compensate for the aforementioned research limitations.

The researcher had no way of verifying the respondents' identities and level of experience with each type of QA organization. Therefore, the data collected should not be used to represent expert opinions on the design of QA organizations or on measuring software quality. Finally, this research cannot be used to draw any definitive conclusions on whether or not a centralized QA organization is better than a decentralized QA organization or vice versa. The study was designed to simply explore the potential relationship between QA organizational design and measuring software quality. It lays the foundation for future studies that seek to determine the reasons why software quality may be poor or inconsistent across an enterprise.

Chapter Summary

The three variables of this research design was the ISO 9126 software quality model which has six characteristics and twenty-one sub-characteristics, the QA organizational model,

and the change in difficulty measuring a software quality sub-characteristic. The research design included the use of several tools including an online survey tool provided by SurveyMonkey to collect and track data, a variance calculator provided by GroundwaterSoftware.com, and a two-tailed, online *t*-value calculator provided by Dimension Research, Inc.

Other key elements of the research design included a measurement framework that established the context in which quality measurements can be taken and helped to identify the groups that may play a role in obtaining quality measurements throughout the entire SDLC. The measurement framework lays additional foundation for the interpretation discussion included in Chapter 5. The research was limited by the researcher's ability to verify the respondents' actual experience with QA organizational design and software quality. However, several steps were taken to ensure that the data received was valid and that any derived data was accurate.

A low response rate was also anticipated, which made finding a large enough respondent population a critical success factor in measuring statistical significance with a 95% confidence level. Chapter 4 describes the survey participants and the methods used for data collection in more detail including sources of data, survey considerations, and response rates.

Chapter 4

Data Collection

An online survey was the chosen mode of data collection for this research. This mode was deemed suitable for a population assumingly familiar with internet technologies and experienced with this mode of data collection. Furthermore, a survey that did not involve direct human interaction or intervention minimized the amount of researcher interference included in the study (Barribeau, 2005). Using a survey standardized questions and ensured that the same data was collected from various groups which is the ideal situation for comparing one group to another (Barribeau, 2005). The following sections describe other key considerations of the data collection process.

Survey Participants

Data sources for this research can be placed into three categories; category one was a personal distribution list of contacts accumulated by the researcher throughout the course of his career, category two was a distribution list of IT professionals obtained through various business partners and category three were publicly available distribution lists available from scguild.com. In all cases, there was no direct interaction or intervention with anyone on any of the distribution lists since SuveyMonkey was used as the delivery mechanism. Furthermore, no identifiable information about the researcher or respondent was solicited or exchanged. All three categories were comprised of IT professionals from various backgrounds (See Appendix D – Survey Respondents), who were assumed to have experience working in or with both types of QA organizations.

Surveying an IT population of diverse backgrounds was intended to develop a balanced perspective on the difficulty measuring software quality. For example, someone from the sales

organization may find that it is extremely easy to measure usability. This may be due to the fact that information on customer complaints or helpdesk tickets is more readily available to the sales department. On the other hand, a QA analyst may find it much more difficult to measure usability because they are often far removed from the end customer. The two perspectives combined will present a truer rating of how difficult it is to collect quality in-use metrics. Stakeholders are those groups and individuals that are involved in the development process and will almost always include representatives from the development organization, appropriate lines of businesses (LOBs), QA organization and end-user population (Koning, 2009). However, here again, it is perfectly acceptable for key stakeholders as well as the amount of participation required by each to vary from company to company (Mullaney, 2008).

Survey Considerations

Survey considerations for this research included choosing the appropriate rating scale, required response type, survey length, the instructions to include in the survey, and the reference materials that should be provided. All of these considerations posed unique challenges. Furthermore, given the research limitations discussed in Chapter 3, it was extremely important to be thoughtful about survey design, since all of these considerations had the potential to further reduce survey response rate.

The survey questions used a ten-point rating scale for several reasons. CustomerSat (www.customersat..com), the leading provider of Enterprise Feedback Management solutions has an extensive history of developing surveys that are designed to help companies retain at-risk customers, valuable employees and partners, optimize overall business performance and save millions of dollars every year. According to CustomerSat best practices a ten-point scale is familiar as most everyone lives in a base-ten world and survey recipients are therefore comfortable selecting a rating that is in line with their actual experience. The ten point scale

also encourages discrimination and increased sensitivity. In other words, a ten-point scale encourages the respondent to not only rate that measurement was difficult, it forces the respondent to think about the degree of difficulty. Finally, a ten-point rating scale is suitable for correlation, regression and other statistical analyses.

The survey carefully considered the type and quantity of questions to include. Survey Monkey was able to accommodate both open-ended and multiple choice questions thereby making it possible to create a survey that would best address the needs of the survey in a manner that was user-friendly. The researcher then sought a number of participants to produce statistically significant results. This survey was also designed to allow users to assign the same rating to multiple quality sub-characteristics. In other words, there was no forced ranking of difficulty among the quality sub-characteristics.

Survey length, given the complexity of the subject matter was another important consideration. According to www.zarca.com, experts in data collection methods, the shorter the survey, the more likely your survey participants are to complete it. A lengthy survey can serve as a deterrent and drastically reduce response rate, which was an original concern of this research. To this end, the survey was kept to a single page that only required the user to scroll down. Additionally, the survey used a progress bar, functionality provided by SurveyMonkey, to show the user's progress as they completed the survey. Thus, the user received instant feedback with the intent of escalating their commitment to completing the survey with each question answered.

Due to the nature of the sample population, it was important to provide the respondents of this survey with a framework to consider when deciding on the difficulty of measuring a sub-characteristic. Though respondents may have had experience working with the ISO quality model, their role may not have allowed them to become completely familiar with all

aspects of the quality standard. Respondents therefore, needed enough reference information in order to close any gaps in their understanding of the terms used by the quality model, such as the meaning of external, internal and quality in-use metrics (See Appendix A – Data Collection Survey). To accomplish this, instructions were included at the very top of the survey that provided a link to the ISO 9126 standard, a definition of internal, external and quality in-use metrics, and verbiage asking the user to consider all types of metrics when determining the difficulty of measuring a sub-characteristic.

One noteworthy limitation of SurveyMonkey was the inability to insert a hyperlink into the survey. To view the ISO 9126 quality standard, respondents were forced to select the url provided and manually paste it into the browser of their choice. Finally, it was also important to mention that the survey did not have to be completed in one sitting. Respondents could save their information and return to survey at a later date. Furthermore, they were able to do this as many times as necessary to complete the survey.

Survey Pilot

A survey pilot was conducted using a small slice of respondents, who were asked to take the survey and provide their feedback on how user friendly the survey was as well as any other feedback that they may have. The pilot participants were not part of the research population and did not contribute any data to the study. The pilot proved to be a critical step in the data collection process as deficiencies were uncovered including confusing question sequence, forced rankings, and insufficient data to be able to respond to the survey. In addition to correcting defects with the survey, the pilot also helped in understanding how to properly set up survey collectors, which are repositories for gathering completed surveys, and further leverage the functionality available through the SurveyMonkey tool. The final survey is included in Appendix A – Collection Survey.

Survey Statistics

Tables 4 –1 through 4-4 contain various statistics related to the survey responses. Table 4-2 summarizes the survey statistics for the personal distribution list, Table 4-3 summarizes the survey statistics for the referred distribution list, and Table 4-4 summarizes the survey statistics for the public distribution list. Each grouping represents a set of survey collectors that were sent to different respondents. A collector was the term used by SurveyMonkey to describe a distribution list that data can be collected from. There were a total of thirteen collectors distributed.

Table 4-1 Category 1 Survey Statistics

Category 1	
Total Surveys Sent:	**57**
Total Response:	**7**
Response Rate:	**12.29%**
Number of Collectors:	**3**

Table 4-2 Category 2 Survey Statistics

Category 2	
Total Surveys Sent:	**4,451**
Total Response:	**70**
Response Rate:	**1.57%**
Number of Collectors:	**2**

Table 4-3 Category 3 Survey Statistics

Category 3	
Total Surveys Sent:	**707**
Total Response:	**29**
Response Rate:	**4.10%**
Number of Collectors:	**8**

Table 4-4 Survey Totals

Survey Grand Totals	
Grand Total of Surveys Sent:	**5,216**
Grand Total of Responses:	**106**
Overall Response Rate:	**2.03 %**

Survey Grand Totals	
Total Number of Collectors:	**13**

Respondents from each category received three e-mails requesting their assistance in completing the survey. All survey collectors were closed approximately three weeks after the initial e-mail was sent. Closing a collector prevented the receipt of additional responses so that results were not changed during the data analysis process.

Chapter Summary

An online survey was chosen as the mode of data collection. This mode was deemed suitable for the respondent population who was considered well versed in web technologies and assumed to be familiar with this mode of data collection. Besides the survey tool, additional considerations included the appropriate sample size, a survey pilot, and other elements of crafting a usable survey. Survey respondents included seasoned IT professionals from distribution lists accumulated by the researcher, distribution lists obtained through business partners, and publicly available distribution lists. A total of 5,216 surveys were distributed and 106 were completed for an overall response rate of 2.03%.

Chapter 5 elaborates on the data collected about the survey respondents and each of the 21 ISO 9126 quality sub-characteristics. For each quality sub-characteristic the findings are discussed and an interpretation of the data is given that attempts to explain the rationale behind the actual difficulty ratings and if any of the QA organizational benefits documented in Chapter 2 could have potentially played a role.

Chapter 5

Findings and Analysis

The data collected from the surveys on the 21 ISO 9126 quality sub-characteristics suggested that either type of QA organization, centralized or decentralized, present the same level of difficulty for measuring 20 of the 21 sub-characteristics. The only exception was the suitability sub-characteristic, which is part of the functionality characteristic. The results suggested that a decentralized QA organization has better ARMS to measure software suitability. Therefore, there was only one instance where there was sufficient evidence not to accept the null hypothesis. The implication to quality conscious enterprises is that they have a multitude of solutions available to them in deciding how to build a QA organization that is aligned with their overall mission.

The survey results proved to be very insightful in not only testing the null hypothesis, but also in uncovering some perceptions that were unexpected. One such notable perception included a survey respondent who did not believe that it was the role of the QA organization to measure certain quality sub-characteristics. See Appendix F – Supporting Information Raw Data for additional comments shared by survey respondents. Each section of this chapter will present the findings on each quality sub-characteristic, followed by the analysis and interpretation for each. The following sections also discuss the survey findings with respect to the demographics of the survey respondents and the ratings assigned to each quality sub-characteristic.

Demographics

Questions one through five dealt with the demographics of the respondent population. The following sections describe the respondent population in more detail to include self classification, total years of IT experience, and the amount of experience working in a

centralized and decentralized QA organization. It also presents some explanation of the response rates achieved by certain groups.

Findings.

The four roles that are almost always involved in the SDLC to include maintenance activities are project managers, business analysts, QA analysts, and software developers (McManus, 2005). Figure 5-1 shows that among these four, the largest response was received by software developers and the lowest number of responses was received from respondents, who classified themselves as QA professionals. The response rate from respondents, who classified themselves as QA professionals supports the literature that states that the QA profession is relatively new. Project managers and business analysts were relatively close in response count with only three responses separating the two groups.

How would you best classify yourself?

Answer Options	Response Percent	# of Responses
Project Management	17.0%	18
Business Analyst	14.2%	15
QA Professional	11.3%	12
Software Developer	26.4%	28
Other	31.1%	33
Other (please specify)		35
answered question		**106**
skipped question		**1**

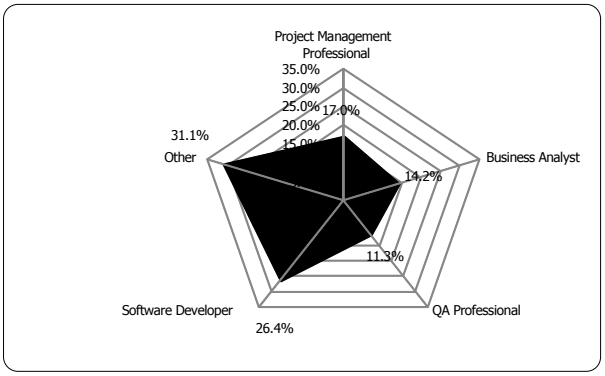


Figure 5-1 Survey Respondents' Self classification

Figure 5-2 summarizes the amount of overall IT experience that each respondent had. The Figure shows that only 2 of the 106 survey respondents had less than 5 total years of IT experience. In fact, over 81 percent of the respondents had more than 15 total years of IT experience, which suggests that our assumption about respondents having experience with both types of QA organizations for the most part holds true and further gives credence to the study.

Note that all graphics displayed will show that 107 surveys were actually started, but only 106 were completed.

How many total years of IT experience do you have?

Answer Options	Response Percent	# of Responses
Less than 5 years	1.9%	2
5 to 9 years	7.5%	8
10 to 15 years	9.4%	10
More than 15 years	81.1%	86
answered question		**106**
skipped question		**1**

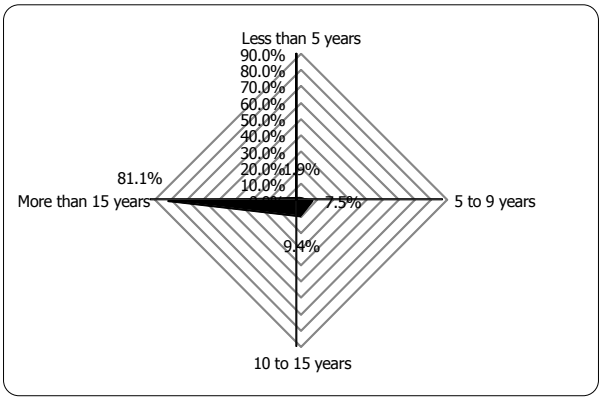


Figure 5-2 Survey Respondents' Years of Experience

Though the experience level with both types of QA organizations was comparable, as seen in Figures5-3 and 5-4, more than 60 percent of respondents in both cases had less than ten years experience. The survey did achieve a balanced response with approximately two-thirds of the response spread across the four key stakeholder groups and the remaining third accounted for by other potential stakeholders shown in Appendix D – Other Survey Respondents.

How many years of experience do you have working in an Enterprise with a centralized QA group that serviced all lines of business?

Answer Options	Answer Options	Response Percent	Response Count
Less than 5 years		37.7%	40
5 to 9 years	5 to 9 years	26.4%	28
10 to 15 years	10 to 15	14.2%	15
More than 15 years		21.7%	23
	answered question		**106**
	skipped question		**1**

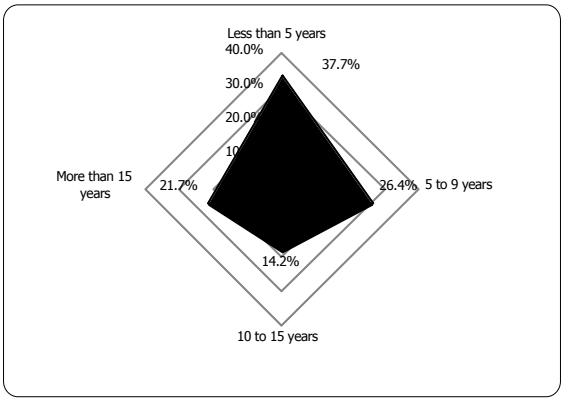


Figure 5-3 Survey Respondents' Experience with Centralized QA Organizations

How many years of experience do you have working in an Enterprise with a decentralized QA organization with separate QA groups that were dedicated to a specific line of business?

Answer Options	Response Percent	# of Responses
Less than 5 years	44.3%	47
5 to 9 years	20.8%	22
10 to 15 years	17.9%	19
More than 15 years	17.0%	18
answered question		**106**
skipped question		**1**

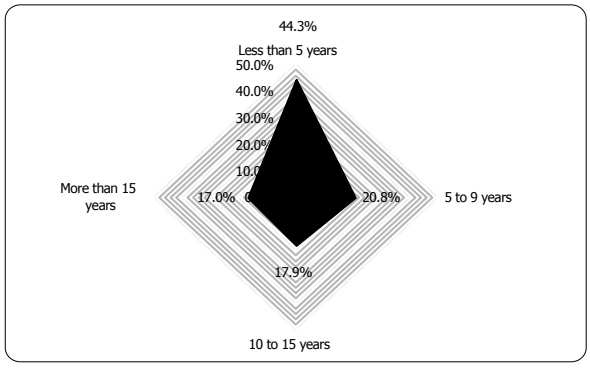


Figure 5-4 Survey Respondents Experience with Decentralized QA Organizations

In short, the survey captured a cross section of IT professionals, the majority of which have had more than five years of experience working with both types of QA organizations. The majority of respondents fall into the four categories that are closest to the SDLC and software maintenance processes. The remaining "Other" category of respondents, more than thirty in all, range from CEOs to database administrators (DBAs).

Analysis and Interpretation.

The audience chosen for the survey was intended to produce a balanced perspective from all parties, who are typically involved in the software development process and other maintenance activities such that they have had a window into the activities performed by the QA organization or have actually been a party thereof. The low response count from QA professionals supported the statements made earlier in this thesis about the maturity of the QA profession. However, the low QA response count may also be misleading as many respondents may have been heavily involved in QA activities, but chose not to classify themselves as a QA professional because they were not officially apart of a QA organization or there was no formal QA title designation.

The data showing experience in both types of QA organizations supported the fact that dedicated QA organizations were difficult to find more than ten years ago even though some QA activities may have been performed (Portnov, 2007). However, more than 60 percent of survey respondents had more than five years of experience working in or with both types of QA organizations, which suggested that the majority of respondents had a good amount of experience on which to base their rating and gave further credence to the study.

Functionality

The functionality characteristic describes the software's ability to perform the transactions that the end-user needs to complete the tasks proscribed by their role. It contains the five sub-characteristics shown in Table 5-1.

Table 5-1 Functionality Characteristic

Characteristics	Sub-characteristics	Definitions
Functionality	Suitability	This is the essential Functionality characteristic and refers to the appropriateness (to specification) of the functions of the software.
	Accurateness	This refers to the correctness of the functions, an ATM may provide a cash dispensing function but is the amount correct?
	Interoperability	A given software component or system does not typically function in isolation. This sub-characteristic concerns the ability of a software component to interact with other components or systems.
	Compliance	Where appropriate certain industry (or government) laws and guidelines need to be complied with, i.e. SOX. This sub-characteristic addresses the compliant capability of software.
	Security	This sub-characteristic relates to unauthorized access to the software functions.

Tabular Results.

Table 5-2 summarizes the data calculated for each functionality sub-characteristic.

Table 5-2 Functionality Measurement Difficulty Ratings

FUNCTIONALITY	$\bar{X}_C$	$\bar{X}_D$	S (C)	S (D)	Std Dev (C)	Std Dev (D)	t value (95 %)	Significant Impact	Δ	C	D
Suitability	6.25	4.80	10.04	9.05	3.17	3.00	3.42	Yes	1.45	-	+
Accuracy	5.55	4.84	7.70	8.57	2.78	2.93	1.80	No	0.71	-	+
Interoperability	5.37	5.50	8.73	9.45	2.95	3.07	0.31	No	-0.13	+	-
Compliance	4.59	5.13	8.68	8.52	2.95	2.92	1.34	No	-0.54	+	-
Security	4.88	5.18	9.29	8.99	3.05	3.00	0.72	No	-0.30	+	-
Overall Rating	26.64	25.45							1.19	-	+

Findings.

In comparing the two organizations, the centralized QA organization received better ratings to measure interoperability, compliance, and security and the decentralized QA organization received better ratings to measure suitability and accuracy. Overall, the decentralized QA organization showed a slightly lower difficulty rating for measuring functionality. Another noteworthy observation was in the area of accuracy, which is one of the more measurable aspects of software quality and was therefore expected to show consistent results across both types of QA organizations. Instead, the decentralized QA organization, though not statistically significant, showed a lower degree of difficulty in measuring software accuracy.

Analysis and Interpretation.

The low difficulty rating given to measuring security in a centralized QA organization may indicate that the activities involved in measuring security metrics such as, requirement reviews and ethical hacking, were more easily performed in a centralized environment. The findings may also indicate that documentation on security standards was not only better maintained by a centralized QA organization, but readily accessible to members of the QA

team. Finally, the findings also implied that the centralized QA team was directly involved in activities involving enterprise wide security issues. The same can be said about the lower difficulty rating given to measuring compliance in a centralized QA organization.

In contrast, a centralized QA organization may pose obstacles to measuring software suitability. For example, the activities involved in measuring suitability may require prototyping pieces of the application that is then shared with the various end-user groups to harvest their feedback, so that improvements can be made to the application. In some cases, the end-user may be located in one of the supported LOBs that the QA organization may or may not have immediate access to for various reasons, such as differing geographies. In other cases, the end-user may be the customer, who is an external entity of the enterprise. The centralized QA organization must then rely on customer feedback to make its way down to the QA organization so that measurement can be taken. Furthermore, customer feedback may take several paths to reach the QA organization such as helpdesk reports or customer satisfaction surveys. The implication here was that a decentralized QA organization is closer to the end-users of the AUT enabling better measurement of suitability. Furthermore, the term closer may not only imply proximity, but may also imply that the end-users have a better level of comfort with the QA analysts from a decentralized QA organization.

Measuring interoperability in a decentralized QA organization, on the other hand, received a higher difficulty rating because it requires enlisting the help of various application teams to test the software's ability to communicate with interfacing systems. However, decentralized QA teams operate in silos with respect to other LOBs. The interaction with other LOBs is necessary only to the extent that there are interfacing applications that require a coordinated effort to collect quality metrics. Therefore, the more integrated applications are, the higher the level of interaction that is required. Relationships with members of other LOBs

then becomes a function of the level of integration required by each application in any given LOB, which may or may not be conducive to measuring interoperability.

One explanation for the lower difficulty rating received by the decentralized QA organization for measuring software accuracy could be the ability of this model to create closer relationships with not only the development organization, but end-users as well. Cultivating good working relationships was one of the documented advantages present in a decentralized QA organization. These relationships can be leveraged during the requirements gathering process to gain a deeper understanding of the software's expected behavior and produce a baselined set of requirements that are free from ambiguity, redundancy and contradictions. A better understanding of the AUT's expected behavior can lead to more accurate measurement.

Reliability

Software reliability is a characteristic of quality that refers to an application's ability to consistently serve its end-user population or other intended purpose without any unexpected disruption in service. The reliability characteristic has three sub-characteristics shown in Table 5-3.

Table 5-3 Reliability Characteristic

Characteristics	Sub-characteristics	Definitions
Reliability	Maturity	This sub-characteristic concerns frequency of failure of the software.
	Fault tolerance	The ability of software to withstand (and recover) from component, or environmental, failure.
	Recoverability	Ability to bring back a failed system to full operation, including data and network connections.

Tabular Results.

Table 5-4 summarizes the data calculated for each reliability sub-characteristic.

Table 5-4 Reliability Measurement Difficulty Ratings

RELIABILITY	$\bar{X}_C$	$\bar{X}_D$	S(C)	S(D)	Std Dev (C)	Std Dev (D)	t value (95%)	Significant Impact	Δ	C	D
Maturity	5.17	5.04	9.63	8.68	3.10	2.95	0.31	No	0.13	+	-
Recoverability	5.05	5.17	8.37	8.83	2.89	2.97	0.30	No	-0.12	-	+
Fault Tolerance	4.95	5.22	9.07	9.10	3.01	3.02	0.65	No	-0.27	-	+
Overall Rating	15.17	15.43							-0.26	+	-

Findings.

In comparing the two organizations, it would appear that a centralized QA organization received better ratings to measure recoverability and fault tolerance. The decentralized QA organization received better ratings to measure maturity. Overall, the decentralized QA organization showed a slightly lower difficulty rating for measuring reliability with a delta of only 1.19. However, testing for statistical significance did not suggest that either type of organization was more suitable for measuring software reliability.

Analysis and Interpretation.

The reason measuring software maturity in a centralized QA organization was rated the most difficult may stem from the fact that centralized QA organizations are responsible for the quality of all software applications within the enterprise. Tracking outages of every application can be very challenging, especially when the applications reside on multiple platforms. Furthermore, the monitoring of each application may be owned by multiple groups outside of the QA organization, such as the infrastructure organization so the QA team's involvement may be only passive in nature.

The lower degree of difficulty in a decentralized QA organization to measure maturity was not surprising, as a decentralized QA organization is typically responsible for fewer applications and QA analysts become very intimate with the applications over time due to their active involvement in the ongoing maintenance activities performed on each application. This

is a documented advantaged of decentralized QA organizations mentioned in Chapter 2. However, the higher difficulty ratings given to measuring fault tolerance and recoverability by a centralized QA organization may be because these activities typically rely on groups outside of the QA organization and are much more passive in nature. For example, the infrastructure team may be responsible for monitoring the software application and reviewing reports about the health of each application over some standard interval of time. These reports contain information about any faults that occurred in the application as well as any application restarts and may not be consistently shared with a centralized QA organization by all LOBs.

The ratings achieved by both organizations, may suggest that whereas a decentralized organization enables better relationships with end-users, a centralized QA organization may be better at cultivating relationships with more technical groups where the tasks involved in measurement do not require the active involvement of the QA team. This could be due to the perception that a centralized QA organization is deemed a peer to other internal organizations and as such, commands a higher level of authority. In contrast, members of a decentralized QA organization may be viewed as just another project team member with no decision making power whatsoever. It was apparent here that the activities involved in measuring software reliability require that the QA organization have ARMS that reached multiple organizations.

Usability

Usability is an aspect of software quality that refers to an applications ability to be readily understood and the ease of which it supports the transactions performed by the end-user. That is, it measures the software's user-friendliness across the sub-characteristics shown in Table 5-5.

Table 5-5 Usability Characteristic

Characteristics	Sub-characteristics	Definitions
Usability	Understandability	Determines the ease of which the systems functions can be understood, relates to user mental models in Human Computer Interaction methods.
	Learnability	Learning effort for different users, i.e. novice, expert, casual etc.
	Operability	Ability of the software to be easily operated by a given user in a given environment.

Tabular Results.

Table 5-6 summarizes the data calculated for each usability sub-characteristic.

Table 5-6 Usability Measurement Difficulty Ratings

USABILITY	$\bar{X}_C$	$\bar{X}_D$	S (C)	S (D)	Std Dev (C)	Std Dev (D)	t value (95 %)	Significant Impact	Δ	C	D
Learnability	5.03	4.64	8.10	8.14	2.85	2.85	0.10	No	0.39	-	+
Understandability	4.99	4.62	8.41	8.26	2.90	2.87	0.38	No	0.37	-	+
Operability	5.00	4.76	7.71	7.57	2.78	2.75	0.37	No	0.24	-	+
	15.02	**14.02**							1.00	-	+

Findings.

In comparing the two organizations, it showed that a decentralized QA organization received better ratings to measure all usability sub-characteristics. Overall, the decentralized QA organization showed a lower difficulty rating by a difference of 1.0. However, testing for statistical significance did not suggest that either type of organization was more suitable for measuring any usability sub-characteristic.

All three ratings for the centralized QA organization were very close to each other with only .04 separating the lowest rating from the highest rating. This may indicate that the activities involved in measuring these characteristics are the same or very comparable. Furthermore, the ratings were all very close to the middle of the scale, which again implies a

likeness in the tasks involved in measurement. Though the margin was a little wider, .12, the same observation can be made about the ratings for a decentralized QA organization.

Analysis and Interpretation.

Usability may be one of the few quality characteristic where the tasks involved in measuring each sub-characteristic are not only similar in both QA organizations, but are also similar in nature. In contrast, the tasks involved in measuring functional sub-characteristics can be starkly different in nature. For example, measuring accuracy may involve executing a set of test cases and comparing the actual results to the expected results. In contrast, measuring interoperability may require coordinating efforts with adjacent application teams or LOBs and collaborating on a test plan. Though the activities involved are very different, both sub-characteristics are part of the functionality characteristic.

The activities involved in measuring learnability, understandability and operability on the other hand can be accomplished by the same or very similar activity. For example, one commonly used activity is prototyping that involves using a working model of the application that contains all or a subset of the functionality that will be included in the final build. The model is then either given to the end-user for their feedback or a joint review of the model is performed. In both cases, structured end-user feedback is captured using some type of feedback document. The document is usually designed to capture all aspects of usability including learnability, understandability and operability.

Though not statistically significant, the ratings received were all lower for the decentralized QA organization. Measuring usability requires a good deal of involvement by the end-user population. As stated in Chapter 2, members of a decentralized QA organization only focus on the applications owned by their LOB and are routinely involved in maintenance

activities that are necessary to ensure that any residual defects are resolved and that there is no degradation in application performance. In doing so, members of a decentralized QA organization accumulate a great deal of application knowledge over time and develop meaningful relationships with the end-users and developers of those applications.

In this case, a decentralized QA organization creates a synergetic environment because the end-user gains the confidence that the QA analyst completely understands their requirements given their history with the application and is better prepared to detect internal and external anomalies before they get back to them. On the development side, the relationship with the QA analyst will be less contentious as they become familiar with each other's communication styles. Furthermore, the QA analyst's knowledge of the application puts them in a better position to translate the end-user requirements into technical requirements that the developer uses to create the application source code. Therefore, the ARMS of the decentralized QA organization embrace the end-user and developer in the spirit of friendship.

Efficiency

Efficiency is an aspect of software quality that refers to an applications ability to use hardware and software resources as effectively as possible as to not create an unpleasant experience for the end-user and complete tasks in a reasonable amount of time. The efficiency characteristic has two sub-characteristics shown in Table 5-7.

Table 5-7 Efficiency Characteristic

Characteristics	Sub-characteristics	Definitions
Efficiency	Time behavior	Characterizes response times for a given thru put, i.e. transaction rate.
	Resource behavior	Characterizes resources used, i.e. memory, cpu, disk and network usage.

Tabular Results.

Table 5-8 summarizes the data calculated for each efficiency sub-characteristic.

Table 5-8 Efficiency Measurement Difficulty Ratings

EFFICIENCY	$\bar{X}_C$	$\bar{X}_D$	S (C)	S (D)	Std Dev (C)	Std Dev (D)	t value (95 %)	Significant Impact	Δ	C	D
Time Behaviour	4.57	4.34	9.03	9.29	3.00	3.05	0.55	No	0.23	-	+
Resource Behaviour	4.84	4.73	9.68	10.12	3.11	3.18	0.25	No	0.11	-	+
	9.41	9.07							0.34	-	+

Findings.

In comparing the two organizations, a decentralized QA organization received better ratings to measure both efficiency sub-characteristics with an overall difficulty rating of 9.07 as opposed to an overall difficulty rating of 9.41 for the centralized QA organization. However, testing for statistical significance did not suggest that either type of organization would yield statistically significant improvements. Both efficiency sub-characteristics received the highest number of "Do not know" responses, approximately 16 for both types of QA organizations. Since the "Do not know" response has a weight of zero, in reality, the difficulty rating for measuring these sub-characteristics is slightly higher than Table 5-8 indicates.

Analysis and Interpretation.

Decentralized QA organizations may have received lower ratings to measure software efficiency because these activities include the use of sophisticated monitoring and testing tools that are application and platform specific. Efficiency metrics are collected using monitoring tools that may be bundled with the application, proprietary to the organization, or bought separately as a standalone application. Most tools available in the market place are designed to monitor a specific platform (i.e. mainframe, UNIX, windows). Each tool also requires deep knowledge of the application as well as the operating system, so that the tool can be properly

configured to collect the metrics that are meaningful to the business and other technical groups within the enterprise. Other activities involved in collecting efficiency metrics may include reviewing system generated reports, helpdesk tickets, customer satisfaction surveys and in some cases, executing a random sampling of transactions in the production environment.

In some cases, measuring efficiency characteristics requires the use of sophisticated performance testing tools. These tools are designed to generate a workload of transactions that represent the actual workload that the application will experience in a production environment. The load is generated using a scripting language to record application transactions that can then be replayed by multiple virtual users. The tools are equipped with monitors that collect information about the health of the application and operating system while they are under load. Here again, the tool requires intricate knowledge of the application and the operating system so that it can be installed and relevant metrics about the application and supporting infrastructure can be collected.

Analysts in a decentralized QA organization accumulate a great deal of knowledge about the application and the platforms that they reside on because of their routine involvement in maintenance activities. Furthermore, the suite of applications that they are responsible for in most cases is much more homogeneous in nature than the application portfolio owned by centralized QA organizations. Therefore, the QA analysts in this organization would have become very familiar with reports provided by their system administrators, first because of their homogeneity and second because of familiarity. Also, QA analysts in this type of organization are very familiar with the process and procedures of their LOB and become very adept at performing the activities involved in collecting external metrics. Finally, as we have seen in other cases, a decentralized QA organization breeds familiarity as supporting organizations become comfortable with the communication styles and behavioral pattern of the

analysts responsible for their suite of applications. The combination of all of the above may explain the lower ratings achieved for both sub-characteristics in the decentralized QA organization.

In contrast, analysts in a centralized QA organization may not have the constant interaction with other LOBS to gather quality in-use metrics and the applications may reside on multiple platforms that require different skill sets. The QA analysts then has to make sense of multiple reports that may be in varying formats and include different quality in-use metrics about the application and the operating system. To collect external metrics, the QA analyst in a centralized QA organization must be given access to various hardware components so that software agents designed to collect metrics can be installed on various hardware components. This requires interacting with multiple LOBs that may each have their own process and procedures.

Maintainability

Maintainability sub-characteristics refer to the software's ability to accept scheduled or unscheduled changes without disruption or with minimal disruption to normal business operations. The maintainability characteristic has three sub-characteristics shown in Table 5-9.

Table 5-9 Maintainability Characteristic

Characteristic	Sub-characteristics	Definitions
Maintainability	Stability	Characterizes the sensitivity to change of a given system that is the negative impact that may be caused by system changes.
	Analyzability	Characterizes the ability to identify the root cause of a failure within the software.
	Changeability	Characterizes the amount of effort to change a system.
	Testability	Characterizes the effort needed to verify (test) a system change.

Tabular Results.

Table 5-10 summarizes the data calculated for each maintainability sub-characteristic.

Table 5-10 Maintainability Measurement Difficulty Ratings

Maintainability	$\bar{X}_C$	$\bar{X}_D$	S (C)	S (D)	Std Dev (C)	Std Dev (D)	t value (95 %)	Significant Impact	Δ	C	D
Stability	4.78	4.87	8.72	8.91	2.95	2.99	0.22	No	-0.09	+	-
Anaylyzability	5.14	5.04	9.51	8.59	3.08	2.93	0.24	No	0.10	-	+
Changeability	5.28	5.13	10.17	9.03	3.19	3.00	0.35	No	0.15	-	+
Testability	4.56	4.76	8.51	8.43	2.92	2.90	0.50	No	-0.20	+	-
	19.76	19.80							-0.04	+	-

Findings.

Unlike the other software quality characteristics, there was agreement across both types of organizations as to the ranking of degree of difficulty measuring each sub-characteristic. In both organizations, the difficulty rating for each sub-characteristic, from lowest to highest was testability, stability, analyzability and changeability. In comparing the two organizations, a centralized QA organization received better ratings to measure stability and testability and a decentralized QA organization received better ratings to measure analyzability and changeability. Overall, the two organizations achieved very similar ratings and testing for statistical significance did not suggest that either type of organization was better for measuring maintainability.

Analysis and Interpretation.

Though the difficulty ratings achieved the same ranking for all sub-characteristics, a decentralized QA organization received lower ratings for measuring analyzability and changeability. Analyzability is a clear example of a sub-characteristic that QA analysts in a decentralized QA organization would not only be more familiar with, but have more active

participation in measuring. Over time, QA analysts in decentralized QA organizations become very familiar with their respective applications and begin to understand all of the application's vulnerabilities. In fact, they may become very adept at predicting where failures may occur in the application. Therefore, when a failure occurs, the QA analyst has a head start in determining where the defect occurred and is better prepared to perform root cause analysis. QA analysts in a decentralized organization also become very good at making changes to the system. This is because they become very intimate with all of the processes and people involved in implementing a change. In contrast, the QA analyst in a centralized QA organization may have to traverse a different process for each application and may interface with different individuals each time. The time to make a change in this type of environment will be extended due to the learning curve associated with new processes and communication styles.

Measuring stability and testability both received lower difficulty ratings for a centralized QA organization. Measuring stability requires that the QA team understand the impact to changing an application. This includes any potential impacts to interfacing applications as well as any downstream applications that they may receive data from. Analysts in a centralized QA organization may have a more holistic view of the application portfolio and understand the way data flows to and from each application and is therefore, in a better position to measure the impact of changing an application.

Much like stability, measuring testability also requires having a holistic view of the application portfolio. This is because to get a proper measurement of testing effort the QA analyst must understand what parts of the application need to be tested as well as any interfaces and downstream applications that should be regression tested. Decentralized QA organizations

operate in a much more siloed fashion and is less likely to understand all of the areas of impact that result from making a change to a particular application.

Portability

The portability characteristic refers to an application's ability to operate in multiple environments and the ease at which modifications can be made to the application so that the environment is abstracted. Portability can become very important to an enterprise because it abstracts the hardware from the software application, which enables it to maximize its return on investment by making the most efficient use of available hardware. The portability characteristic has four sub-characteristics shown in Table 5-11.

Table 5-11 Portability Characteristic

Characteristic	Sub-characteristics	Definitions
Portability	Installability	Characterizes the effort required to install the software.
	Replaceability	Characterizes the *plug and play* aspect of software components, that is how easy is it to exchange a given software component within a specified environment.
	Adaptability	Characterizes the ability of the system to change to new specifications or operating environments.
	Conformance	Similar to compliance for functionality, but this characteristic relates to portability. One example would be Open SQL conformance which relates to portability of database used.

Tabular Results.

Table 5-12 summarizes the data calculated for each portability sub-characteristic.

Table 5-12 Portability Measurement Difficulty Ratings

PORTABILITY	$\bar{X}_C$	$\bar{X}_D$	S (C)	S (D)	Std Dev (C)	Std Dev (D)	t value (95 %)	Significant Impact	Δ	C	D
Installability	4.30	4.81	7.07	9.00	2.66	3.00	1.30	No	-0.51	+	-
Replaceability	5.25	5.11	8.10	8.96	2.85	2.99	0.35	No	0.14	-	+
Adaptability	5.44	5.02	9.13	9.12	3.02	3.02	1.01	No	0.42	-	+
Conformance	4.40	4.94	7.52	9.10	2.74	3.02	1.36	No	-0.54	+	-
	19.39	**19.88**							-0.49	+	-

Findings.

In comparing the two organizations, the centralized QA organization received better ratings to measure installability, and conformance and the decentralized QA organization received better ratings to measure replaceability and adaptability. Overall, the centralized QA organization showed a slightly lower difficulty rating with a delta of only 0.49. However, testing for statistical significance did not suggest that either organization was better for measuring portability.

Analysis and Interpretation.

Installability, which is the effort required to install software, is almost never done directly by the QA organization. This is usually due to the security issues involved in getting access to the hardware that the software resides on. Yet measuring installability received the lowest difficulty rating for both the decentralized and centralized QA organizations. The involvement by the QA organization in both cases is passive as they would have to rely on metrics given to them by system administrators from various LOBs or their LOB in the case of the decentralized QA organization.

The lower difficulty rating achieved by the centralized QA organization suggests that this may be easier to accomplish with a centralized model. This can be interpreted to mean that a centralized model commands more authority within the enterprise such that other organizations are more willing to adhere to processes that permit the flow of this information into the organization. On the other hand, members of a decentralized QA model may have a more difficult time trying to extract this information due to unclear processes or lack of responsiveness from their technical counterparts.

Replaceability and adaptability are both slightly different than installability in that they both involve measurements that are taken after initial installation of the application. Both sub-characteristics received lower difficulty ratings for a decentralized QA organization. As with measuring other sub-characteristics, members of a decentralized QA organization become very adept at performing the activities required to measure a quality sub-characteristic over time. This is because the same QA analysts are responsible for a defined suite of applications and are involved in the ongoing maintenance activities that happen on a regular basis once the application goes into production. As a result, they develop meaningful relationships with the various groups required to maintain the application and support their QA activities. So even though a decentralized QA organization may not have clearly defined processes or procedures in place, they compensate for this by developing effective working relationships with their colleagues to get their job done.

The conformance sub-characteristic received a lower difficulty rating for a centralized QA organization. Conformance is similar to the functional sub-characteristic of compliance, but in this context refers to the technical standards used to develop software applications. Though these standards can be created by each LOB, it stands to reason that an enterprise that is truly concerned with the portability of their applications would set these standards at the enterprise level and expect all LOBs to adhere to them. Only then, can you create an enterprise that is able to abstract the hardware from its software applications. In this case, the centralized QA organization is better suited to enforce strict coding standards across all LOBs and in turn measure application conformance.

Chapter Summary

The survey achieved a balanced response with approximately two-thirds of the responses spread across the four key stakeholder groups and the remaining third accounted for

by the "Other" category, which included CEOs, DBAs and system administrators to name a few. Several steps were taken to ensure that survey data was reliable and that the calculations performed by SurveyMonkey were valid. Table 5-13 is the consolidated summary of the findings for each quality sub-characteristic where C represents the centralized QA organization, D represents a decentralized QA organization, *t* is the calculated *t*-value and s is whether or not there was statistical significance.

Table 5-13 Measurement Difficulty Summary Table

FUNCTIONALITY	**C**	**D**	***t***	**S**
Suitability	-	+	3.42	Yes
Accuracy	-	+	1.80	No
Interoperability	+	-	0.31	No
Compliance	+	-	1.34	No
Security	+	-	0.72	No
RELIABILITY				
Maturity	+	-	0.31	No
Recoverability	-	+	0.30	No
Fault Tolerance	-	+	0.65	No
USABILITY				
Learnability	-	+	0.10	No
Understandability	-	+	0.38	No
Operability	-	+	0.37	No
EFFICIENCY				
Time Behavior	-	+	0.55	No
Resource Behavior	-	+	0.25	No
MAINTAINABILITY				
Stability	+	-	0.22	No
Analyzability	-	+	0.24	No
Changeability	-	+	0.35	No
Testability	+	-	0.50	No
PORTABILITY				
Installability	+	-	1.30	No
Replaceability	-	+	0.35	No
Adaptability	-	+	1.01	No
Conformance	+	-	1.36	No

The only quality sub-characteristic that showed statistically significant results was suitability, which is part of the functionality characteristic. Analysis of the suitability

characteristic suggested that a decentralized QA organization has better ARMS to measure suitability. However, the researcher can only speculate as to which advantage of the decentralized QA organization was responsible for the lower rating achieved for suitability. Areas of speculation included the increased complexity of having to deal with multiple LOBs by a centralized organization, which would make measuring suitability more prone to human error. Tests of significance for all other sub-characteristics did not suggest that either QA organization was better suited to obtain their measurement. Furthermore, the researcher could again, only speculate as to which benefit present in either organization could explain the edge that may have been received by one sub-characteristic over the other.

In Chapter 6 the study will be summarized to include the initial intent of the research, the recommendations implied by the findings, limitations of the research and suggestions for future research.

Chapter 6

Conclusions and Recommendations

The data from this research suggested that the difficulty measuring any of the 21 software quality sub-characteristics outlined in the ISO 9126 software quality model is essentially the same in either a decentralized or centralized QA organization. The only exception was the suitability sub-characteristic, which is part of the functionality characteristic. The results showed that a decentralized QA organization would have better ARMS to measure software suitability. Therefore, there was only one instance where there was sufficient evidence not to accept the null hypothesis. The implication to quality conscious enterprises is that they have a multitude of solutions available to them in deciding how to build a QA organization that is aligned with its overall mission.

Recommendations

The findings from this research and the literature review gave some important insights that companies should consider when building their QA organization. The researcher has used these insights to develop a recommended approach to designing a QA organization referred to as the 5Ds (Define, Differentiate, Decide, Distinguish, and Design) of QA organizational design. The 5Ds of QA organizational design are:

1. Define an appropriate software quality model. The quality model helps to standardize the definition of software quality and gives the software developers and end-users a tool that can be used to set expectations about the behavior of the software and establish quality goals. The quality model should define all software quality characteristics, the type of metrics that will be collected for each one, and the relationship of the metrics to each other. The ISO 9126 standard is a great place to start and can be customized to

meet the needs of the enterprise. This is the very first step in creating a QA organization with ARMS.

2. Differentiate your business. Determine the software characteristics that are important to the mission of the business. These are the quality sub-characteristics that differentiate the firm in the marketplace, such as speed of response time or 100 percent availability.
3. Decide what type of metrics to collect. Some characteristics may require the collection of more than one type of metric. This information is also very useful in determining the groups that the QA organization will have to interact with.
4. Distinguish the metrics that should be set and tracked at the enterprise level. There may be certain metrics that are tracked at the enterprise level either because they apply to every LOB, require knowledge of the entire application portfolio or are required by government and other industry regulations and must be strictly enforced.
5. Design accordingly. If the characteristics that differentiate your business require knowledge of the entire application portfolio then a centralized QA organization may be more appropriate for your business. Otherwise, a decentralized QA organization may be more appropriate, because of its ability to accumulate application specific knowledge and develop effective working relationships.

In addition to the 5 Ds, QA professionals and managers should perform deeper analyses of varying perspectives on measuring software quality as it may uncover a broken process that does not allow certain metrics to be measured effectively by the QA Organization. The analyses may also uncover an area where the QA organizational design is lacking and could be used to recommend changes to existing processes, such as involving the QA organization in

sales meetings or assigning a representative from the QA organization, who is responsible for resolving all production issues. Another option could be to create an entirely new QA team whose sole focus is on quality in-use metrics.

Finally, managers should consider the growth strategy of the company when deciding how to design the QA organization. A company that relies on acquisitions to grow must have an infrastructure in place that facilitates knowledge transfer, creates economies and applies consistent rigor. All can be achieved by employing a centralized QA organization, which may or may not continue to exist upon completion of the acquisition.

Summary

QA has emerged into a science onto itself. However, the effect that the QA organizational design has on measuring the various aspects of software quality is a problem that current literature does not address. There have been works published on the pros and cons of various QA models, but none attempted to address which, if any, software quality characteristic is affected by the organizational design. This researched explored QA organizational design as it relates to software quality to determine if a certain type of organization was better suited to measure some characteristic of software quality (Hayes,2003; Topping, 2009).

This research was an exploratory study intended to increase the body of knowledge on decentralized and centralized QA organizations and their effect on measuring a specific characteristic of quality. Increased knowledge in this area will allow managers to design QA organizations that are customized to their type of business. In other words, an organization that conducts commerce on its website may be primarily concerned with measuring the speed of their software to differentiate themselves in the market place. However, an organization that produces expert systems to determine the optimal level of inventory a company should carry

may be less concerned with measuring application speed and more concerned with measuring software accuracy.

The independent variable of this study was the ISO 9126 software quality model that included six characteristics and twenty-one sub-characteristics, the moderating variable was the QA organizational model, which had two values, centralized and decentralized and the dependent variable under study was the difficulty measuring the twenty-one, ISO 9126 software quality sub- characteristics. The dependent variable was calculated by calculating the difference in difficulty ratings between the centralized and centralized QA organization for each quality sub-characteristic.

The research addressed the following problem statement: The role, if any, that organizational structure plays on measuring software quality. Measuring software quality often involves activities that require collaborating with multiple groups within the organization. In some cases, the groups may be internal to the enterprise, while others may be external, such as a customer. Today's companies must design QA organizations that strengthen their presence in the marketplace (Siggelkow, 2003). This can only be accomplished by ensuring that each department, including the QA organization has access to the information it needs and is able to perform the activities that are required to measure the characteristics of software quality that are most relevant to their business (Westfall, 2005). In some cases, this may require decentralization, other cases may require centralization, and yet others may be able to achieve the same results in either model. The literature available on centralized and decentralized QA organizations do not attempt to address this question.

A mature QA organization must have ARMS. However, which type of QA organization created better ARMS to measure all characteristics of software quality was at the heart of this research. The following hypothesis was tested:

H_0: There is no difference in difficulty to measure any ISO 9126 software quality sub-characteristic between a centralized and decentralized QA organization.

The key tools used to collect and analyze data were SurveyMonkey, a variance calculator provided by GroundwaterSoftware.com, and a two-tailed, online t-value calculator provided by Dimension Research, Inc. An online survey with implemented best practices was the chosen mode of data collection for this research. Over 5,200 IT professionals were surveyed and 107 responded for an overall response rate of 2.05 percent.

The only quality sub-characteristic that showed statistically significant results was suitability, which is part of the functionality characteristic. Research findings indicated that a decentralized QA organization had better ARMS to measure software suitability. However, the researcher could only speculate as to which organizational advantage, if any, provided by the decentralized QA organization is responsible for the result. The insights of this research and the literature review have been used to develop a recommended approach to designing a QA organization with ARMS referred to as the 5Ds.

Originality

This research expanded on the existing research on the pros and cons of centralized and decentralized QA organizations. These studies have found that each type of QA organization offers certain advantages including, increased subject matter expertise, better application knowledge, and effective working relationships. However, the studies do not attempt to link the organizational type to the ability to measure a specific quality characteristic. This study dissected the ISO 9126 software quality standard and attempted to map each quality sub-characteristic to the type of QA organization that was best suited to measure it. In fact, this study suggested that though the findings on the advantages of different QA organizational

models may be true, they are not enough to have an effect on measuring any characteristic of software quality.

Contribution to the Body of Knowledge

This study added to the body of knowledge in the relatively young IT field of software quality assurance by expanding on existing studies of QA organizations. It also provided a platform on which future studies can be built. The contemporary QA professional can refer to this study as the starting point to understand why certain applications in their portfolio may be suffering from certain software deficiencies. The understanding gained from this research can be used by QA professionals and corporate managers alike to make changes to existing processes that can improve the software quality characteristics that are critical to their business.

In addition to benefiting the QA community this study may also be beneficial to researchers, who are studying human behavior in the QA organization. More specifically, the findings from this study can be analyzed to determine if decentralized and centralized QA organizations elicit certain human behaviors that are essential for measuring a certain characteristic of software quality. This type of research would be beneficial to a very broad community including QA professionals, corporate managers, sociologists, and anthropologists.

Research Limitations

A key limitation of this research was the inability to determine what organizational advantage, if any, was responsible for obtaining a better result. For example, though findings suggested that suitability could be better measured by a decentralized QA organization, the researcher could only speculate why this result may be true. Speculation could include any number of variables including, relationships, process or expertise. Therefore, the research did address the problem statement by demonstrating that in most cases the difficulty measuring

software quality was the same in either QA organizational model, but it could not explain what caused one organization to have better ARMS than the other.

Another limitation of this research was the use of the ISO 9126 software quality model. This research was very specific to the ISO 9126 standard. Substituting another quality model may have yielded very different results. There are other software quality models available and many others can be created that may offer more relevance to a particular industry. This research should not be used to trivialize the importance of the quality model when determining the best way to design a QA organization. In fact, the first step in designing a QA organization with ARMS is to define an appropriate software quality model.

Finally, the researcher did not have the ability to verify the actual number of years that each survey respondent had with each type of QA organization. As such, the data compiled for this research should be used to represent expert opinions or to make any type of conclusive statements about the suitability of one QA organizational model over another.

Scope of Future Research

Using this study as the foundation, future studies can be designed to test multiple hypotheses including:

1. H_0: Centralized QA organizations are better at measuring software quality sub-characteristics due to more defined processes.
2. H_0: Decentralized QA organizations are better at measuring software quality sub-characteristics due to its ability to cultivate effective working relationships
3. H_0: Decentralized QA organizations are better for user-centric measurement due to its ability to cultivate effective working relationships.
4. H_0: Centralized QA organizations are better for techno-centric measurement due to more defined processes

5. H_0: A hybrid of QA organizational models (centralized and decentralized) produces better quality software than any single model.
6. H_0: Working relationships have an impact on software suitability
7. H_0: Enterprise knowledge is irrelevant to QA organizational Design

The above hypotheses will help to resolve the limitations faced by this study and will take QA professionals, researchers and students of QA further down the path of producing software that consistently meets the objectives of their business.

References

Arthur, J. D., & Nance, R. E. (2000). Verification and validation without independence: A recipe for failure. *Proc. 2000 Winter Simulation Conference*, Orlando, FL.

Arthur, J., & et. al. (1999). Evaluating the effectiveness of independent verification and validation.

Bach, J. (1998). A framework for good enough testing. *Software Realities,* Retrieved from http://www.satisfice.com/articles/good_enough_testing.pdf

Barribeau, P., et al. (2005). *Survey research.*

Bartlett, J. E. I., Kotrlik, J. W., & Higgins, C. C. (2001). Organizational research: Determining appropriate sample size in survey research. *Information Technology, Learning, and Performance Journal, 19*(1), 43.

Basu, A. D. (2005). Assuring software quality with ISO 9126. *IT Management, ,* 28.

Bowers, A. (2003). *The case for a central support group, professional tester.* Retrieved May 27, 2006, from www.professionaltester.com/magazine/backissue/16/ProTesterOct2003-Bowers.pdf

Burnstein, I., Homyen, A., Suwanassart, T., Saxena, G., & Grom, R. (1999). A testing maturity model for software test process assessment and improvement. *SQP 99 1, ASQ, 4*

Certo, S. C. (2000). *Modern management, diversity, quality, ethics, and the global environment* (8th ed.) Prentice-Hall, Inc.

Clark, T. (1999). *eBay online again after 14-hour outage - CNET news.* Retrieved 11/25/2008, 2008, from http://news.cnet.com/eBay-online-again-after-14-hour-outage/2100-1017_3-229518.html

Elfield, D. (2004). *Getting our own house in order, professional tester.* Retrieved May 27, 2006, from www.professionaltester.com/magazine/backissue/17/ProTesterJan2004.Elfield.pdf

Fajardo, J. (2008). *Article info : Outsourcing testing tasks: A competitive advantage.* Retrieved 11/25/2008, 2008, from http://www.stickyminds.com/sitewide.asp?Function=edetail&ObjectType=ART&ObjectId=7837

Feldman, S. (2005). *Software quality assurance is more than testing and IEEE standard 12207.* Retrieved 5/3/2009, 2009, from http://elsmar.com/Forums/showthread.php?t=11148

Florac, W. (1992). *Software quality measurement: A framework for counting problems and defects.* Retrieved May 28, 2009, from www.sei.cmu.edu./publications/documents/92.reports/92.tr.022.html

Ganssle, J. G. (2001). A guide to code inspections. Retrieved from http://www.ganssle.com/inspections.pdf

Glass, R. (1997). *Software runaways: Lessons learned from massive software project failures* Prentice-Hall PTR.

Hayes, L. (2003). Column info : Test organization strategies: Centralized, distributed, or hybrid?*2008*(11/25/2008)

Holub, E, Pultz, J., Scott, D. (2006). Organizing IT infrastructure and operations: Trends and best practices [Abstract].

Honda, N. (2009). Beyond CMMI level 5: Comparative analysis of two CMMI level 5 organizations. *4 SQP*, *11*(4)

Johnson, J. (1995). Chaos: The dollar drain of IT project failures, application development trends. (pp. 41-41-47)

Kasabe, N. (2005). *Software testing: After thought necessity.* Retrieved May 1, 2006, from www.dqindia.com/content/special/2005/105012501.asp

Koning, L. (2009). *Project stakeholders, projects and the key people.* Retrieved July/19, 2009, from http://business-project-management.suite101.com/article.cfm/project_stakeholders

Lewis, R. O. (1992). Independent verification and validation: A life cycle engineering process for quality software.

Liversidge, E. (2005). *The death of the V-model.* Retrieved 5/5/2009, 2009, from http://www.harmonicss.co.uk/index.php/tutorials/software-engineering/56?task=view

Lubinsky, R. G. (2009). *Benchmarking: You can't control what you don't Measure* . Retrieved May, 2009, 2009, from http://architechmag.com/News/emATWeeklyem/ArticleDetails/tabid/171/ArticleID/8428/Default.aspx

McClure, R. M. (1968). Introduction to the 1968 NATO software engineering conference.

McManus, J. (2005). Selected essays in stakeholder management. *Managing stakeholders in software development projects* (pp. 139) Butterworth-Heinemann.

Mullaney, J. (2008). *Software quality assurance more than just testing.* Retrieved 5/5/2009, 2009, from http://searchsoftwarequality.techtarget.com/news/article/0,289142,sid92_gci1317775,00.html

NASA. (2009). *Software definitions.* Retrieved 5/5/2009, 2009, from http://ezproxy.library.nyu.edu:13976/office/codeq/software/umbrella_defs.htm

NASA IV&V overview

NASA policy directive NPD 8730.4A, effective august 1, 2001(2001).

Neslson, J. G. (Ed.). (1979). *Software testing in computer driven systems, in software quality management* Petrocelli Books.

Nindel-Edwards, J., & Steinke, G. (2006). A full life cycle defect process model that supports defect tracking, software product cycles, and test iterations. *Communications of the IIMA, 6*(1), 3.

Ogale, C. (2005). *Testers vs programmers.* Retrieved 5/26, 2009, from http://www.indicthreads.com/1330/testers-vs-programmers-2/

Patton, R. (2006). *Software testing* (2nd ed.). 800 East 96th Street, Indianapolis, Indiana 46240: Sams Publishing.

Perry, W. (2006). *Effective methods for software testing* (3rd ed.). Indianapolis, Indiana: Wiley Publishing, Inc.

Podlogar, Y. (2002). *Taking advantage of a centralized QA organization.* Retrieved May 28, 2009, 2009, from http://www.stickyminds.com/s.asp?F=S3516_ART_2

Portnov, M. (Producer), & Portnov, M. (Director). (2007). *Software testing - career training @ portnov school - 2.* [Video/DVD]

QA Systems. (2003). *QA systems / concepts.* Retrieved 5/5/2009, 2009, from http://www.qa-systems.com/concepts/iso9126.html

Rakitin, S. (2001). *Software verification and validation for practitioners and managers* (2nd ed.) Artech House.

Rice, R. (1996). *"Defects are good - randy rice's software testing site".* Retrieved 11/25/2008, 2008, from http://www.riceconsulting.com/articles/defects-are-good.htm

Schulmeyer, C. G., & Mackenzie, G. R. (2000). *Verification and validation of modern software-intensive systems* Prentice-Hall PTR.

Siggelkow, N., & Levinthal A., D. (2003). Temporarily divide to conquer: Centralized, decentralized, and reintegrated organizational approaches to exploration and adaptation.*14*(6), 650-650-669.

Software engineering, report on a conference sponsored by the NATO science committee, garmisch, germany, october 7-11, 1968(1968).

Tavassoli, D. (2008). Strategic QA. steps to effective quality assurance [Abstract].

TechTarget. (2007). *What is V-model? - a definition from whatis.com - see also: Vee-model.* Retrieved 5/5/2009, 2009, from http://searchsoftwarequality.techtarget.com/sDefinition/0,,sid92_gci1263512,00.html

Topping, S. (2009). Organizing localization in large companies, achieving balance between centralized and decentralized models. *Multilingual Computing and Technology, 10*(6)

Unknown. (2002). *Software bugs cost U.S. economy $59.6 billion annually, RTI study finds - RTI international.* Retrieved 11/25/2008, 2008, from http://www.rti.org/page.cfm?nav=350&objectid=DA7FBFE6-4A4F-4BFD-B77E0FA3C04D9E22

Wallace, D. R., & Fuji, R. U. (1989). Software verification and validation: Its role in computer assurance and relationship with software project management standards.

Warren, L. (2007). *You cannot control what you do not measure.* Retrieved 5/29, 2009, from http://www.thefabricator.com/TestingMeasuring/TestingMeasuring_Article.cfm?ID=1756

Webopedia. (2009). Retrieved 5/5/2009, 2009, from http://www.webopedia.com/TERM/S/Software_Quality_Assurance.html

Weiner, L. (1993). Digital woes: Why we should not depend on software.

Westfall, L. (2005). 12 steps to useful software metrics., 2009.

Appendix A – Data Collection Survey

1. Centralized vs Decentralized QA Effectiveness

For questions 5 thru 16, please refer to http://www.qa-systems.com/concepts/iso9126.html as needed for definitions of each quality characteristic and when considering how a characteristic is measured please account for the following types of metrics:

INTERNAL METRICS:
Internal metrics are those which do not rely on software execution (static measures).

EXTERNAL METRICS
External metrics are applicable to running software.

QUALITY IN-USE METRICS
Quality in use metrics are only available when the final product is used in real conditions.

1. How many total years of IT experience do you have?

- ◯ Less than 5 years
- ◯ 5 to 9 years
- ◯ 10 to 15 years
- ◯ More than 15 years

2. How would you best classify yourself?

- ◯ Project Management Professional
- ◯ Business Analyst
- ◯ QA Professional
- ◯ Software Developer
- ◯ Other

Other (please specify)

3. How many years of experience do you have working in an Enterprise with a centralized QA group that serviced all lines of business?

- ◯ Less than 5 years
- ◯ 5 to 9 years
- ◯ 10 to 15 years
- ◯ More than 15 years

4. How many years of experience do you have working in an Enterprise with a decentralized QA organization with separate QA groups that were dedicated to a specific line of business?

- ◯ Less than 5 years
- ◯ 5 to 9 years
- ◯ 10 to 15 years
- ◯ More than 15 years

5. SOFTWARE FUNCTIONALITY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Suitability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Accuracy	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Interoperability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Compliance	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Security	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

6. SOFTWARE RELIABILITY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Maturity	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Recoverability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Fault Tolerance	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

7. SOFTWARE USABILITY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Learnability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Understandability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Operability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

8. SOFTWARE EFFICIENCY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Time Behaviour	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Resource Behaviour	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

9. SOFTWARE MAINTAINABILITY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Stability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Analyzability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Changeability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Testability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

10. SOFTWARE PORTABILITY [CENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Installability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Replaceability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Adaptability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Conformance	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

14. SOFTWARE EFFICIENCY [DECENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Time Behaviour	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Resource Behaviour	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

15. SOFTWARE MAINTAINABILITY [DECENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Stability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Analyzability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Changeability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Testability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

16. SOFTWARE PORTABILITY [DECENTRALIZED QA ORGANIZATION]

On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

	1	2	3	4	5	6	7	8	9	10	Do not know
Installability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Replaceability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Adaptability	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Conformance	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Please add any supporting information

Appendix B – Survey Messages

Generic message used for pilot:

We are conducting a survey, and your response would be appreciated.

Here is a link to the survey:
[SurveyLink]

This link is uniquely tied to this survey and your email address. Please do not forward this message.

Thanks for your participation!

Please note: If you do not wish to receive further emails from us, please click the link below, and you will be automatically removed from our mailing list.
[RemoveLink]

Final message used for actual data collection:

Dear Colleague,

I am a New York University Graduate student conducting research on software quality and would greatly appreciate your assistance.

You are receiving this e-mail, because one of your colleagues recommended you as having input that is pertinent to this very important research on software quality.

I am conducting a brief survey that should take no more than 10 minutes of your time and would greatly appreciate your response. Please take 10 minutes to complete this very important survey.

Here is a link to the survey:
[SurveyLink]

This link is uniquely tied to this survey and your email address. Please do not forward this message.

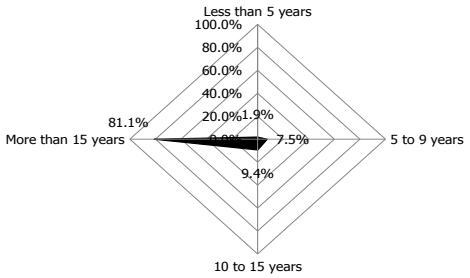
Thanks for your participation!
[RemoveLink]

Appendix C – Survey Demographic Summary

Question 1:

How many total years of IT experience do you have?

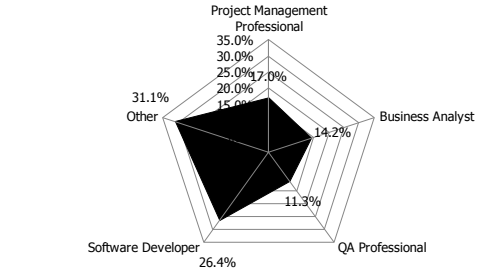
Answer Options	Response Percent	Response Count
Less than 5 years	1.9%	2
5 to 9 years	7.5%	8
10 to 15 years	9.4%	10
More than 15 years	81.1%	86
answered question		**106**
skipped question		**1**



Question 2:

How would you best classify yourself?

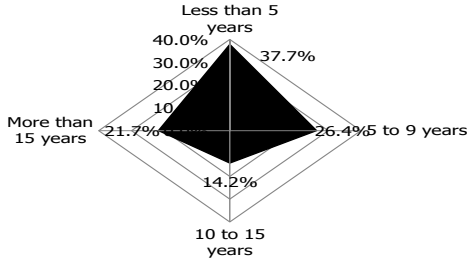
Answer Options	Response Percent	Response Count
Project Management	17.0%	18
Business Analyst	14.2%	15
QA Professional	11.3%	12
Software Developer	26.4%	28
Other	31.1%	33
Other (please specify)		35
answered question		**106**
skipped question		**1**



Question 3:

How many years of experience do you have working in an Enterprise with a centralized QA group that serviced all lines of business?

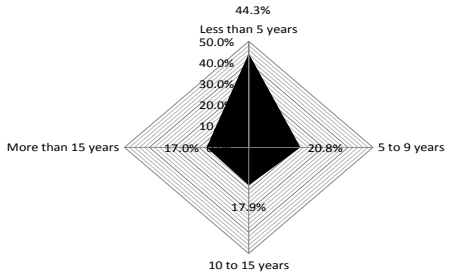
Answer Options	Response Percent	Response Count
Less than 5 years	37.7%	40
5 to 9 years	26.4%	28
10 to 15 years	14.2%	15
More than 15 years	21.7%	23
answered question		**106**
skipped question		**1**



Question 4:

How many years of experience do you have working in an Enterprise with a decentralized QA organization with separate QA groups that were dedicated to a specific line of business?

Answer Options	Response Percent	Response Count
Less than 5 years	44.3%	47
5 to 9 years	20.8%	22
10 to 15 years	17.9%	19
More than 15 years	17.0%	18
answered question		**106**
skipped question		**1**



Appendix D – Other Survey Respondents

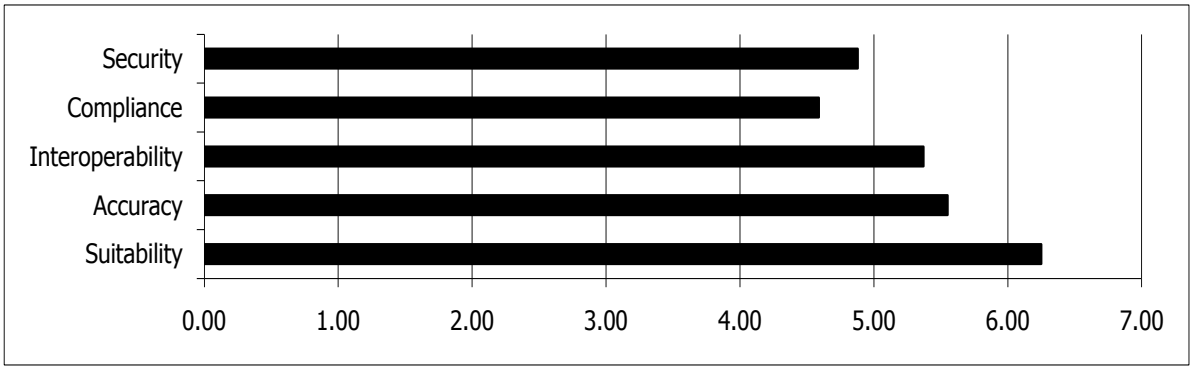
1. Senior IT Professional
2. & Process Analyst
3. IT Management
4. IT Procurement Analyst
5. Manager
6. Architect
7. Systems and B/I architect
8. IT Manager
9. Consultant
10. Project Mgt (not certified) / Projects coordinator
11. Implementation Consultant
12. Application Architect/IT Manager
13. Database Manager / Business Officer
14. Director
15. All of the above
16. Consultant, architect
17. QA Manager
18. Information Engineer
19. CEO
20. All of the above
21. Multimedia Producer
22. Enterprise Architect
23. Operating Systems Analyst
24. Database administration
25. Instructional Designer
26. Systems Programmer
27. Systems Engineer
28. Senior Computer Consultant
29. Technical writer/business analyst/online help author
30. Systems Analyst/Database Developer
31. System Administrator, DBA
32. Sales
33. Mainframe Systems Analyst
34. All of the above as CEO of Centralized Corporate live models for over 100 Corporations in different businesses
35. Information & Technical Architect
36. IT Solutions sales & marketing

Appendix E – Survey Response Details

Question 5:

SOFTWARE FUNCTIONALITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

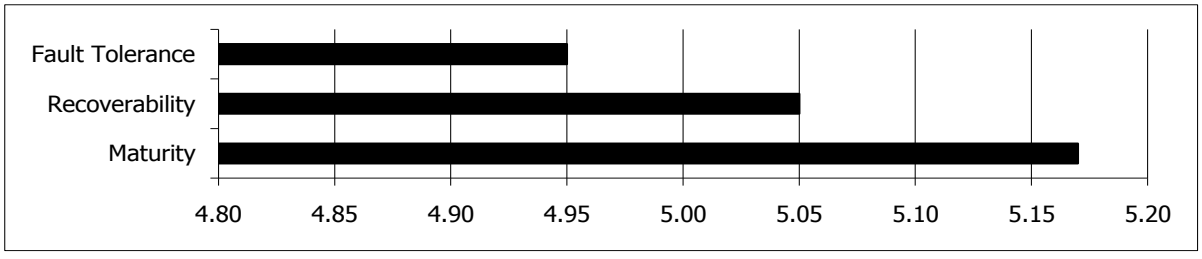
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Averag	Response Count
Suitability	3	4	5	2	13	7	10	20	18	13	11	6.25	106
Accuracy	5	9	5	7	13	9	20	14	9	7	8	5.55	106
Interoperability	5	8	4	10	16	9	10	18	12	4	10	5.37	106
Compliance	13	9	15	6	16	5	11	10	10	3	8	4.59	106
Security	10	9	11	6	19	8	7	10	10	7	9	4.88	106
Please add any supporting information													18
												answered question	**106**
												skipped question	**1**



Question 6:

SOFTWARE RELIABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

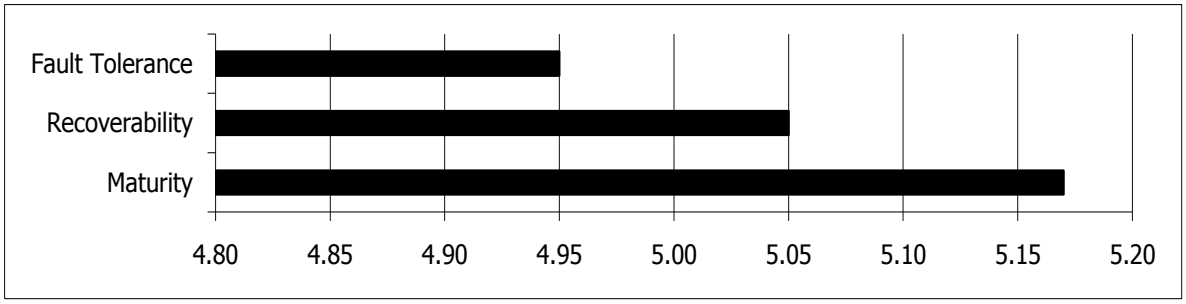
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Maturity	5	8	7	5	18	8	10	17	8	7	13	5.17	106
Recoverability	5	10	10	9	15	10	17	6	8	7	9	5.05	106
Fault Tolerance	6	11	10	8	16	9	10	11	7	8	10	4.95	106
Please add any supporting information													11
											answered question		**106**
											skipped question		**1**



Question 7:

SOFTWARE USABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

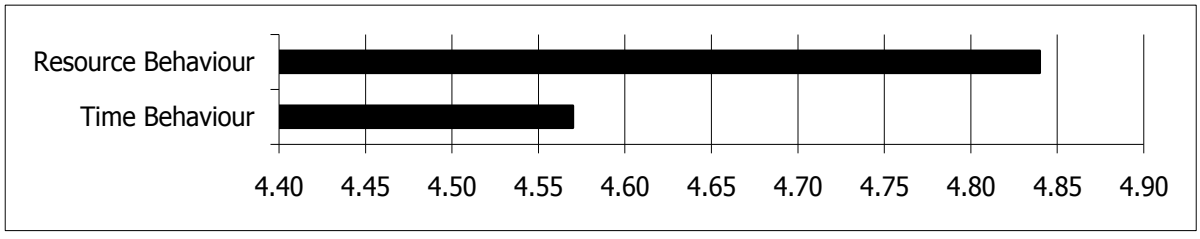
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Learnability	3	10	9	10	18	12	12	7	9	6	10	5.03	106
Understandability	3	10	14	9	12	10	17	8	5	8	10	4.99	106
Operability	2	11	11	11	20	7	9	15	6	5	9	5.00	106
Please add any supporting information													13
												answered question	**106**
												skipped question	**1**



Question 8:

SOFTWARE EFFICIENCY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

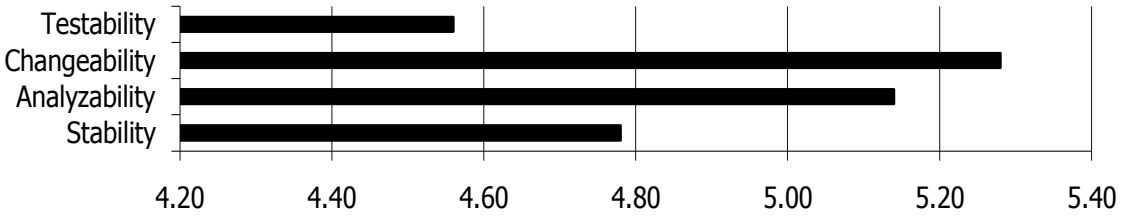
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Time Behaviour	4	9	10	11	20	2	13	9	7	5	16	4.57	106
Resource Behaviour	5	5	12	6	17	8	11	13	6	7	16	4.84	106
Please add any supporting information													7
												answered question	**106**
												skipped question	**1**



Question 9:

SOFTWARE MAINTAINABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

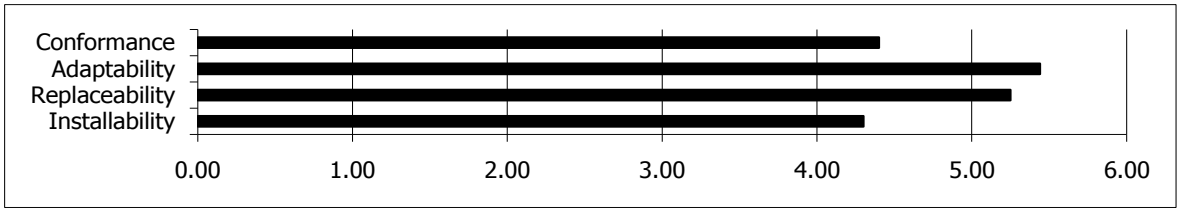
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Stability	6	9	16	9	15	8	10	7	10	6	10	4.78	106
Analyzability	7	5	9	6	19	7	13	11	9	8	12	5.14	106
Changeability	5	9	7	7	16	5	12	13	10	10	12	5.28	106
Testability	9	10	15	8	15	8	12	6	8	5	10	4.56	106
Please add any supporting information													7
												answered question	**106**
												skipped question	**1**



Question 10:

SOFTWARE PORTABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

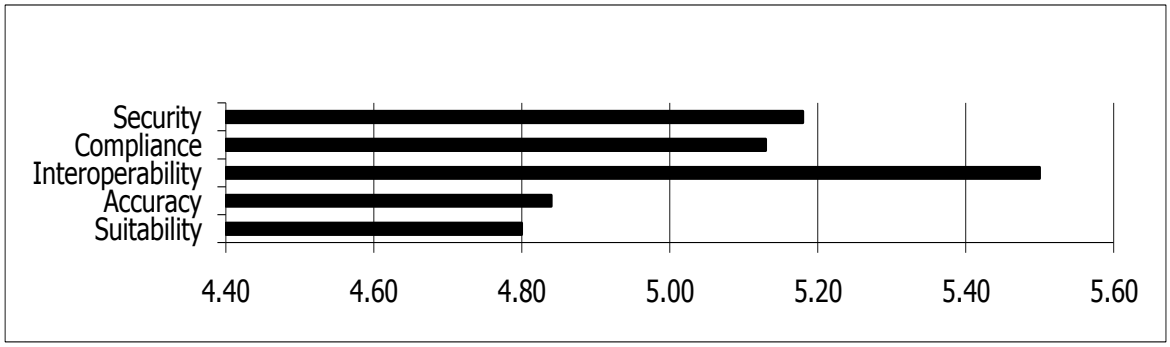
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Installability	8	14	13	11	16	11	12	5	3	4	9	4.30	106
Replaceability	4	6	7	12	12	18	14	8	8	7	10	5.25	106
Adaptability	4	6	9	7	12	9	16	15	10	7	11	5.44	106
Conformance	8	12	14	8	17	11	10	8	5	3	10	4.40	106
Please add any supporting information													7
												answered question	**106**
												skipped question	**1**



Question 11:

SOFTWARE FUNCTIONALITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business

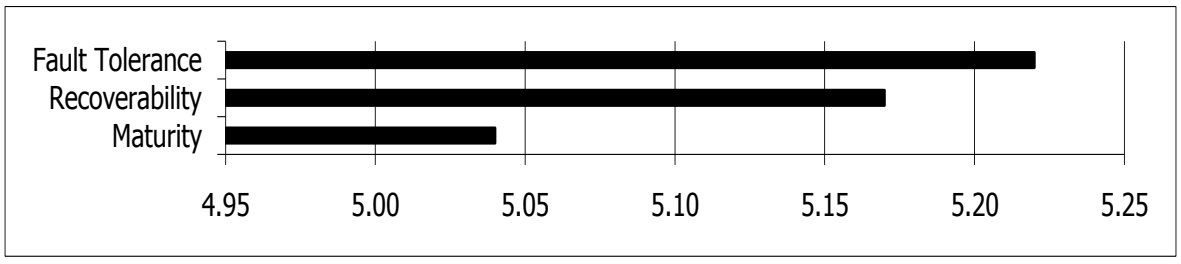
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Suitability	6	11	12	8	16	6	12	9	9	6	11	4.80	106
Accuracy	5	13	13	9	14	8	11	10	8	6	9	4.84	106
Interoperability	5	6	8	8	13	13	8	13	13	9	10	5.50	106
Compliance	5	9	9	11	15	10	11	11	10	6	9	5.13	106
Security	4	7	10	9	9	11	18	13	6	7	12	5.18	106
Please add any supporting information													10
												answered question	**106**
												skipped question	**1**



Question 12:

SOFTWARE RELIABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

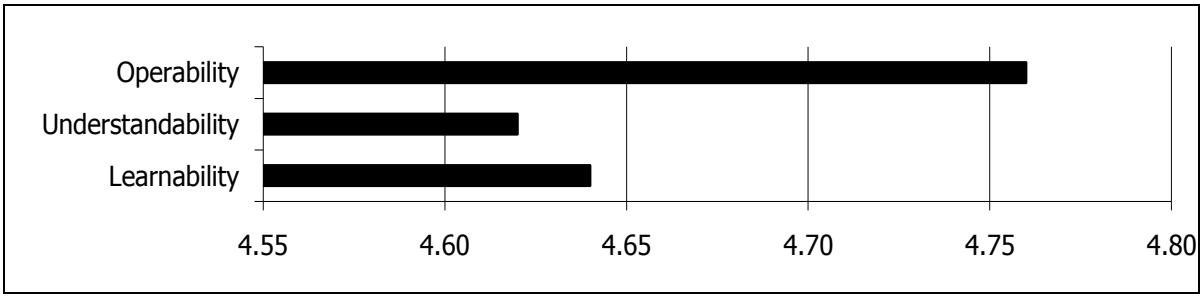
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Maturity	5	6	6	16	12	14	11	7	12	5	12	5.04	106
Recoverability	4	7	11	9	14	8	15	10	12	5	11	5.17	106
Fault Tolerance	5	8	7	10	13	10	13	12	11	6	11	5.22	106
Please add any supporting information													6
												answered question	**106**
												skipped question	**1**



Question 13:

SOFTWARE USABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

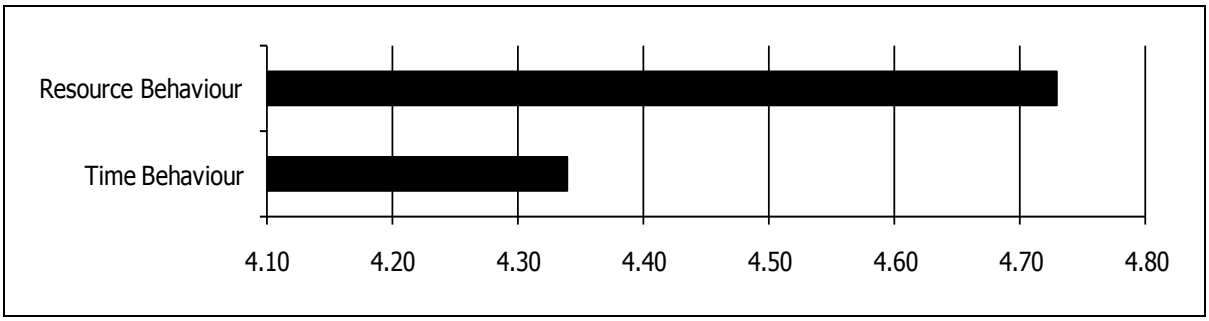
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Learnability	6	11	11	10	16	12	7	12	6	4	11	4.64	106
Understandability	8	13	10	8	13	13	9	14	4	4	10	4.62	106
Operability	4	12	10	9	17	15	9	11	5	4	10	4.76	106
Please add any supporting information													7
												answered question	**106**
												skipped question	**1**



Question 14:

SOFTWARE EFFICIENCY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

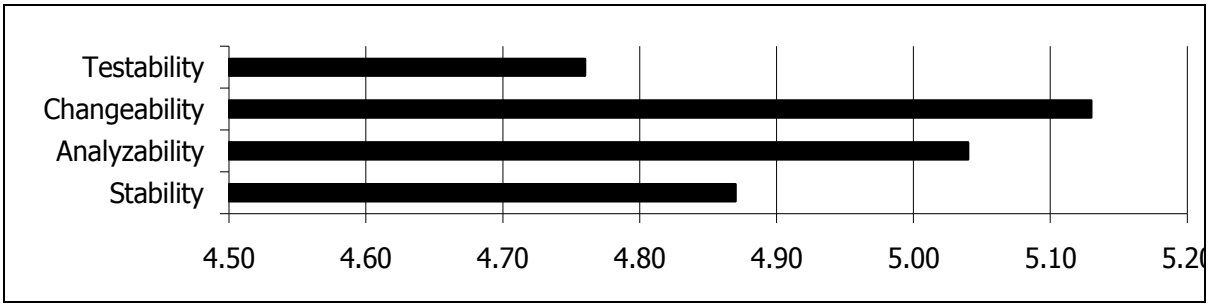
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Time Behaviour	7	12	17	5	11	6	11	12	6	4	15	4.34	106
Resource Behaviour	6	10	9	6	13	9	11	14	4	8	16	4.73	106
Please add any supporting information													4
												answered question	**106**
												skipped question	**1**



Question 15:

SOFTWARE MAINTAINABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of busine

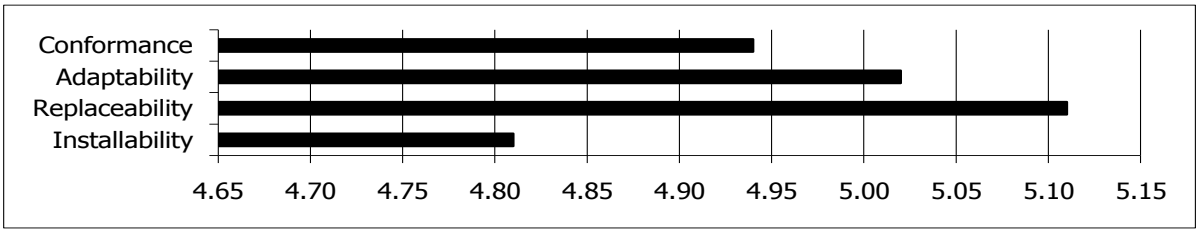
Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Stability	5	12	11	7	16	4	17	10	7	6	11	4.87	106
Analyzability	5	9	7	10	15	11	13	14	4	7	11	5.04	106
Changeability	5	8	7	12	15	8	12	12	9	7	11	5.13	106
Testability	7	10	12	10	13	11	8	14	7	4	10	4.76	106
Please add any supporting information													3
												answered question	**106**
												skipped question	**1**



Question 16:

SOFTWARE PORTABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

Answer Options	1	2	3	4	5	6	7	8	9	10	Do not know	Rating Average	Response Count
Installability	7	8	11	3	18	9	14	11	8	4	13	4.81	106
Replaceability	6	6	9	7	15	9	14	13	12	3	12	5.11	106
Adaptability	4	6	12	7	12	9	13	18	7	4	14	5.02	106
Conformance	4	9	10	9	15	6	12	17	5	6	13	4.94	106
Please add any supporting information													5
												answered question	**106**
												skipped question	**1**



Appendix F – Supporting Information Raw Data

Question 5: SOFTWARE FUNCTIONALITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. I have had experience with a Human Factors group that went from centralized to decentralized. Both have their drawbacks. Centralized groups are too far away from each line of business to understand it well, and there is no getting around that.

2. You lose systems knowledge in a centralized QA environment. If you are running multiple environments this becomes very challenging.

3. It all depends on the quality of the requirements definitions, specifications, and test plans-- which all come from the end-user organizations.

4. A centralized organization allows for a greater knowledge base for interoperability, compliance, and security measurements because of easy access to the collective QA staff experience. Suitability and Accuracy measurements are more influenced by the relationship between BAs, Dev, and QA.

5. Read the documentation before starting and then follow them to the letter. Then after these tests, do everything wrong and see what happens.

6. QA needs to be a separate organization and testing needs to be a separate location to keep impartial.

7. I'm referring to a centralized QA organization within a business entity, but within a QA organization there may be sub-groups that focus on specific lines.

8. Compliance and security would generally be standard across the lines.

9. Any function in a centralized organization will be difficult to measure... particularly if there is no baseline or requirements to measure against.

10. The development of technology makes a centralize QA more appealing and ROI is a lot better. Staff training in higher level technology across various disciplines makes it a challenge.

11. Generally, centralized QA organizations lack the subject matter expertise to validate Suitability, Accuracy and interoperability of systems deployed in an enterprise.

12. Professionals in such an organization either need to be exceptionally experienced, in order to understand diverse business lines enough OR need to have the time, knowledge and communication skills necessary to gain an understanding. In my

experience, organizations don't want to pay anyone to attain the required level if understanding.

13. Often, subtle system interactions are difficult to communicate through from the developer->QA handoff. Unless there's an integrated birth->deploy documentation endemic in the entire organization, details *will* get lost. It would be inappropriate for me to answer further questions, as they don't directly relate to my experiences.

14. Compliance and Security are relatively easy because of the specific requirements to test. Accuracy and Interoperability are more challenging because of the environments required in which to test them. Test environs are often extremely different than production and present challenges such that testers don't know if the root cause is the environment or the code.

15. It depends on how large the overall org is and how many applications the QA group has to support.

16. If they do not understand the process, they cannot evaluate the quality.

17. Subject Matter Expertise: Developers have to come to this to be able to write the software. Can a centralized QA group have within it the people to do this? Can this be achieved in light of the fallacious theory that one person can replace another? I deal with QA personnel today who claim to know AS/400 systems, and they are mediocre at best. Wanna discuss z/VM or z/OS? And taking a M/S oriented person off to *nix systems... please pass the migraine medication.

18. A central Unified, Compliant and Complete Operational - Financial - Quality Live Corporate Model of the Live Business meets all of the above functionality and more.

Question 6: SOFTWARE RELIABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. Not knowing the individual lines of business the Centralized QA is more of a job shop so it would difficult to judge the maturity of a software package.

2. It is expensive to have experts in-house so sharing a central person across the enterprise is better.

3. Time required to complete tasks is not accounted for when selecting a challenge level.

4. Well written Regression test scripts should be designed to achieve these results.

5. What is the definition you're using for maturity?

6. Not sure what you mean by maturity in this context.

7. I'm not entirely sure that the structure of the QA organization would have a significant impact on the QA organization's ability to measure these attributes. My assumption is that QA professionals would be trained to use and thus *would* use the most appropriate tools and techniques to make these measurements regardless of the corporate structure. However, I'm inclined to believe that in many cases and independent of organizational structure, QA organizations are often limited as to the types of software quality attributes they are allowed to measure. These limitations could be imposed by management levels of the QA organization or of the corporate structure as a whole due to cost considerations or philosophical considerations as to the suitability or desirability of the QA organization undertaking the specific type of measurement.

8. It depends on how large the overall org is and how many applications the QA group has to support. Also it has varied on how knowledgeable a QA is about getting into the data and systems tables/files.

9. See above

10. If one doesn't know the underlying hardware architecture and understand, fully, the software architecture built on top of it, then one can't determine Maturity because faults detected may not be the product's [being tested] problems, but the O/S or even devices.

11. A central Unified, Compliant and Complete Operational - Financial - Quality Live Corporate Model of the Live Business meets all of the above with total Continuity.

Question 7: SOFTWARE USABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. Not knowing the individual lines of business the Centralized QA is more of a job shop so it would difficult to judge the usability of a software package.

2. The using organizations have to provide a lot of this info.

3. Time required to complete tasks is not accounted for when selecting a challenge level.

4. I have never worked in an organization that could get their hands around these measurements.

5. Depends on the funding & intelligence of the project management team & software.

6. Depends on the complexity of the applications and the target markets/users

7. Not sure what these -ilities are supposed to encompass in this context

8. Again - centralized organizations seldom have the level of understanding necessary to make these assessments.

9. In my experience, test conditions will mirror exactly what the users would do to operate the system. These three attributes could be measured on a user-by-user basis if they were involved in testing the system. Learning and testing at the same time goes a long way to make sure it works in prod.

10. My inclination is to believe that few QA organizations are tasked with undertaking the measurement of these particular attributes.

11. If you do not have a background in computers, you will not understand its function

12. Again, much of this is based on the architectures that the application is running on. So, if you are well versed in M/S products, and you are testing a Mainframe application, how that product relates to the environment and the tester's understanding of the environment affects all of this.

13. With system having available Unified Compliant and Reliable live model of the live business 24/7 it is easy to learn, understand, operate and manage a business as well as its information system.

Question 8: SOFTWARE EFFICIENCY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. The ability to measure these attributes is not really affected by the QA organizational structure

2. Assuming you mean measuring TPS and resource usage under load (why not just say what you mean?) It is typically challenging for any QA organization - central or not - to measure these metrics on enterprise scales. Particularly in heterogeneous environments.

3. Again, I don't believe the measurement of these attributes should be influenced by the organizational structure.

4. When I was with IRS, we would first have to train the QA's on what the system was and how it was supposed to function.

5. Please refer to my prior comments. Comparing a road-racing motorcycle to an ocean going cargo ship...

6. With a Unified Corporate model that is compliant to business operational, financial and accounting rules and is transparent to managers, auditors and employees at the same time and in real time specification, development, correction and maintenance are

relatively simple, fast and inexpensive with problems and other issues being corrected in real-time before they become crises.

Question 9: SOFTWARE MAINTAINABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. Time required to complete tasks is not accounted for when selecting a challenge level.

2. QA software has advanced greatly over the last few years. Education is dropping.

3. Again, you refer to -ilities that have undefined meaning in this context.

4. Since the topic here is maintainability, I'm assuming the subtopics stability, etc. refer to the code. Not being a QA professional myself, I can't speak to the norm but in most software development organizations I've been involved with, QA has nothing to do with the raw code; software maintainability has in my experience been the sole responsibility of the development team rather than that of the QA organization.

5. Unclear on QA org role vs developers' roles. Also some QA/testers have the ability to get into backend testing others only seem capable of more simple front end test/evaluation

6. See prior comments.

7. With a Unified Corporate model that is compliant to business operational, financial and accounting rules and is transparent to managers, auditors and employees at the same time and in real time specification, development, correction and maintenance are relatively simple, fast and inexpensive with problems and other issues being corrected in real-time before they become crises.

Question 10: SOFTWARE PORTABILITY [CENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a centralized QA organization to measure the following software quality attributes for each line of business?

1. A centralized organization may be better at installibility because you need a certain amount of ignorance to ensure that installability is idiot proofed

2. Again, they need a baseline to measure most of these. Most organizations don't have baseline measurements.

3. These are all based of the difficulty of the algorythms & procedures of the various lines of business

4. While creating a questionnaire with all of these -ilities may seem clever, there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

5. I'm not sure I see adaptability and conformance as QA issues but rather as development issues.

6. It is NOT QA's job to determine these things. As a developer, I have already seen issues with COBOL being migrated across platforms (I have used COBOL on ASCII based systems, EBCDIC based systems, as well as Field Data based systems). Data handling is an issue. Data display is an issue. How many people in QA have engineering experience that crosses multiple architectures and back?

7. With the Corporate model and the Business being central and controlling, software portability becomes a secondary issue.

Question 11: SOFTWARE FUNCTIONALITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. Our QA groups look only at accuracy.

2. Decentralized organizations lack the global view, so interoperability is a challenge

3. My experience has been that a centralized QA group has a good expectation of the overall business needs and requirements where a decentralized one does not. Therefore a decentralized QA group has a tendency to unknowingly build in more interoperability problems.

4. Emphasize that you are now switching to DEcentralized.

5. Both centralized and decentralized still must make observations of the difficulty of the algorithms & procedures of the various lines of business in order to have an accurate response. You may begin with one and end up using the other because it fits the environment.

6. While creating a questionnaire with all of these -ilities may seem clever, there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

7. When a testing org is siloed, making sure all of the parts fit in the bigger org is extremely challenging. These attributes are difficult to measure if not impossible. The risk to the business is considerably higher when testing is decentralized.

8. For this and subsequent topics, see my comments to the similar topics above.

9. Taking people who work with a specific architecture all the time cuts the training time to figure out these issues.

10. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Question 12: SOFTWARE RELIABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. Our QA groups look only at accuracy

2. De-centralization leads to variation in assessment results

3. TO me, these are just silly questions with what I have told you already.

4. While creating a questionnaire with all of these -ilities may seem clever; there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

5. A decentralized organization is typically closer in touch with the supported business line and understands what they are supporting from a business perspective - rather than just a technical perspective.

6. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Question 13: SOFTWARE USABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. See comment above.
2. In general, they are too accustomed to their own products

3. This is based on the assumption the learnability & understanility are limited to the specific line only.

4. While creating a questionnaire with all of these -ilities may seem clever; there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

5. These wouldn't be particularly challenging, but again the risk to the business is higher when users can't see the bigger picture. They don't know what they don't know (have you heard that saying before?)

6. These can be lower because they may be so close to the application that they may not see it from the outside user perspective.

7. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Question 14: SOFTWARE EFFICIENCY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. See prior comments

2. The ability to measure these attributes is not really affected by the QA organizational structure

3. While creating a questionnaire with all of these -ilities may seem clever, there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

4. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Question 15: SOFTWARE MAINTAINABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. See above

2. While creating a questionnaire with all of these -ilities may seem clever; there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

3. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Question 16: SOFTWARE PORTABILITY [DECENTRALIZED QA ORGANIZATION] On a scale of 1 to l0, 10 being most challenging, how challenging is it for a decentralized QA organization to measure the following software quality attributes for their respective lines of business?

1. See above

2. This is a dumb survey.
3. All responses the complexity and access to the environments. Defect activities (open, closed, time to closure, etc.) should play a key roll in assisting with each of the areas. Solid Requirements will also play key role in assessing each of the areas.

4. While creating a questionnaire with all of these -ilities may seem clever; there is insufficient context in which to formulate answers. Therefore any data obtained is so tainted as to be useless.

5. Decentralized IT is like decentralized business. I have never seen a single reason for decentralizing and complicating. Instead I used centralized approach to make available the best quality of information. COMPLETE, COMPLIANT, CORRECT, LIVE and AFFORDABLE

Appendix G – *t* Table

T-Distribution Table

df	α = 0.1	0.05	0.025	0.01	0.005	0.001	0.0005
∞	t_α = 1.282	1.645	1.960	2.326	2.576	3.091	3.291
1	3.078	6.314	12.706	31.821	63.656	318.289	636.578
2	1.886	2.920	4.303	6.965	9.925	22.328	31.600
3	1.638	2.353	3.182	4.541	5.841	10.214	12.924
4	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	1.476	2.015	2.571	3.365	4.032	5.894	6.869
6	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	1.314	1.703	2.052	2.473	2.771	3.421	3.689
28	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	1.311	1.699	2.045	2.462	2.756	3.396	3.660
30	1.310	1.697	2.042	2.457	2.750	3.385	3.646
60	1.296	1.671	2.000	2.390	2.660	3.232	3.460
120	1.289	1.658	1.980	2.358	2.617	3.160	3.373
∞	1.282	1.645	1.960	2.326	2.576	3.091	3.291