TESTQUAL: Conceptualizing Software Testing as a Service

Yang Yang
First Tennessee Bank, yangum@gmail.com

Colin Onita
University of Memphis, cgonita@memphis.edu

Jasbir Dhaliwal
University of Memphis, jdhaliwl@memphis.edu

Xihui Zhang
University of North Alabama, xzhang6@una.edu

TESTQUAL: Conceptualizing Software Testing as a Service

Yang Yang
First Tennessee Bank
yangum@gmail.com

Colin Onita
University of Memphis
cgonita@memphis.edu

Jasbir Dhaliwal
University of Memphis
jdhaliwl@memphis.edu

Xihui Zhang
University of North Alabama
xzhang6@una.edu

ABSTRACT

Software testing has emerged as a distinct responsibility in software development. This paper argues that software testing should be conceptualized as a service rather than being viewed as a sequential line of responsibility in software development. Testing as a service has two key aspects: (1) a service to developers, and (2) a service to end users. This paper draws from the SERVQUAL and software quality literature in information systems to propose a structured measurement tool for testing as a service and to briefly discuss its relationship with software quality. The TESTQUAL framework is described, and its components are defined and exemplified. The practical implications of TESTQUAL are also discussed.

Keywords

Software testing, software development, service quality, software quality, SERVQUAL, TESTQUAL.

INTRODUCTION

Recent thinking about the nature of software architecture in organizations has been significantly influenced by models such as service-oriented architecture (SoA) and software-as-a-service (SaaS) (Goth, 2008). This is partly a result of the fact that it is often difficult for developers in disparate organizations to determine which software component is interfacing with others. At the same time, today's end users, who have become more technologically sophisticated (Davis, Kettinger, and Kunev, 2009), have significantly higher expectations about the quality and functionality of software. Traditional testing practices, which have not changed significantly over the years, are unable to satisfy these emerging requirements. Therefore, it is imperative to consider the relationships between developers, testers, and end users from new perspectives. One such new perspective is that of "testing as a service," which is in line with the general trend in information technology (IT) management to decompose the value of an IT unit in terms of a structured set of services rendered to both internal and external customers of an organization. In both academic and professional circles, this is often referred to as the "IT services" paradigm. This paradigm argues that organizations need to consider each value bundle provided by the corporate IT unit from a service provider and a service consumer relational perspective (Jia, Reich, and Pearson, 2008). This then leads to a focus on measuring and managing the service quality in such relations.

Testing can be viewed as a service provided to software developers in relation to error minimization, validation, and verification (Bentley, 2005). Without the services provided by testers, the burden of error finding, verification, and validation would fall onto developers, who may not be equipped with needed capabilities and technical resources to properly handle a comprehensive testing process. This is especially critical as it has been argued that developers are more technically oriented while testers are more process and business oriented (Cohen, Birkin, Garfield, and Webb, 2004). Testers are the last arbiters who ensure that the software product is of high quality and meets end users' business needs.

This testing-as-a-service perspective is an important paradigm shift from the old perspective where testing is simply a technical quality assurance mechanism. In this new paradigm, issues such as software usability, user satisfaction, mapping to business needs, and information systems coherence become part of the testing agenda leading to a more holistic role for testers beyond just technical testing of the code. To measure and manage the service provided by testing, business and IT managers require a tool that allows for quantification of the quality of service provided. This paper attempts to fill this need by adapting an academic and practitioner validated service measurement tool – SERVQUAL, to the software testing context.

TRADITIONAL APPROACHES TO TESTING

One of the traditional methods for developing complex systems is the systems development lifecycle cycle (SDLC) (Mahmood, 1987), which is usually composed of consecutive stages, where the outputs of each step become the inputs for the subsequent stages. In this SDLC framework, testing usually focuses on technical issues to ensure that the code runs correctly and is compatible with other software components. In addition, testers will briefly check if the code complies with the user requirements and specifications. Typically, testing is performed by an independent group of testers after the software is developed by developers but before it is shipped to the end users. Past studies revealed that testing was separated from the development activities due to this sequential working mode of developers and testers (Whittaker, 2000). Figure 1 shows two groups of arrows with opposite directions: the arrows pointing from left to right are the software products (or code) from Developers to Testers, and from Testers to End Users; the arrows in the other direction are feedback from End Users to Testers, and Testers to Developers.

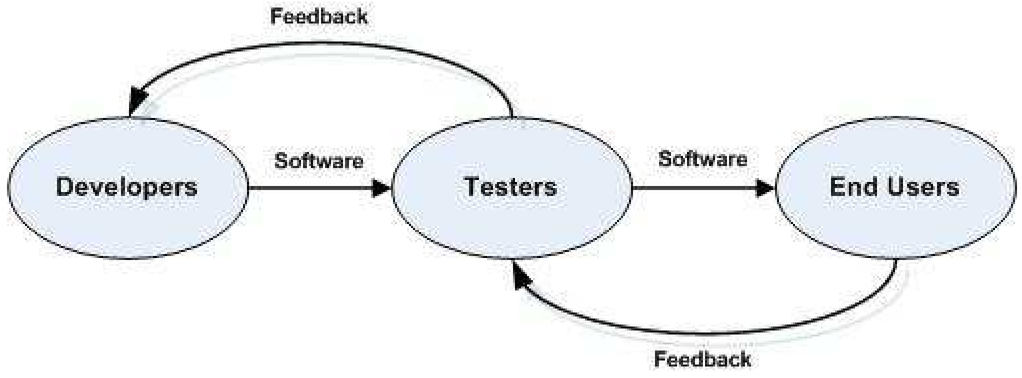

Figure 1. Traditional Workflow among Developers, Testers, and End Users

This sequential development and testing process can lead to conflict between developers and testers (Cohen et al., 2004). Developers and testers have to constantly compete for project time. Often times testing is postponed, and the planned testing time is reduced to meet the delivery schedule. The sequential software development practice often results in compromises of product quality that result from inadequate time being spent on testing the code, validating the requirements, or mapping specifications to business needs. In this out-of-date testing practice, testers are very much focused on the mechanical side of testing, which does not provide the level of testing quality required by today's business environment.

TESTING AS A SERVICE

The "testing as a service" paradigm has the capability of conceptualizing all its activities as distinct services to pertinent constituencies. Software testing encompasses a wide spectrum of different activities, from testing a small piece of code (unit testing), validating a system (acceptance testing), to monitoring a network-centric service-oriented application at runtime. There are different sets of objectives for testing at different levels (e.g., unit test, integration test, and system test) and at different stages of the software production cycle; it is becoming a requisite that testers work together with both developers and end users early and often along the software production life stages to reach these different objectives.

Figure 2 shows that testers provide key services to two distinct constituencies: service to developers (Service I) and service to end users (Service II). Service I is provided to developers and it pertains to error detection, verification, and validation of the code. For example, unit testing by testers is a service that provides to developers to ensure that individual modules work right. Another aspect of this service is exemplified by code review and functional testing services which are to ensure that code written by developers is consistent with business requirements and software specifications. A third aspect of this service is provided by boundary testing, stress testing, and load testing that focus on ensuring that the code will work in the intended, real world business context.

Service II captures all categories of testing services that testers provide to ensure that end users will receive software that meets their needs. Specific examples of these services include requirements validation at early stages, alpha testing, beta testing, and usability testing at later stages of software development. In requirements validation, testers need to understand end users' expectations about the software so that appropriate testing plans and strategies can be implemented to guard the users' interests. Testers will analyze user requirements to construct test cases so that linkages are established between the two. User requirements validation is akin to services provided by testers in regard to business needs validation (functional requirements), software performance (performance requirements), and security (security requirements). Different applications with varying sets of requirements often require a diverse set of testing services. With Alpha testing service, testers engage users directly in system validation by allowing them to interact with the system in controlled testing environments. With Beta testing service, users are engaged through controlled software distribution processes tied to feedback loops that enable users to test software in real world conditions. Testers herein often act as facilitators in dynamic feedback loops designed to customize and localize software to specific user environments. Services I and II, as illustrated in Figure 2 and explained above, form the foundation of the TESTQUAL tool proposed in this paper as a holistic measurement of the testing service quality.

The other three dotted lines illustrate the importance of the three stakeholders' (developers, end users, and testers) combined contributions to the actual software product during software production cycle. Although developers and testers are mostly held accountable for software quality, more recent literature has demonstrated that involving end users in early stages of software development life cycle is beneficial to reduce uncertainty and to improve overall software quality (Hsu, Chan, Liu, and Chen, 2008). We argue that there's an urgent need to elevate testers to the more strategically centralized position in software development, as they are the information integrators of end users' higher level business requirements and developers' intermediate software modules, and more importantly, the last arbiters of the software product. Organizations that rely on software for their central business processes are unlikely to adopt and implement a software system unless it has been adequately tested by qualified testers using sound testing methods.

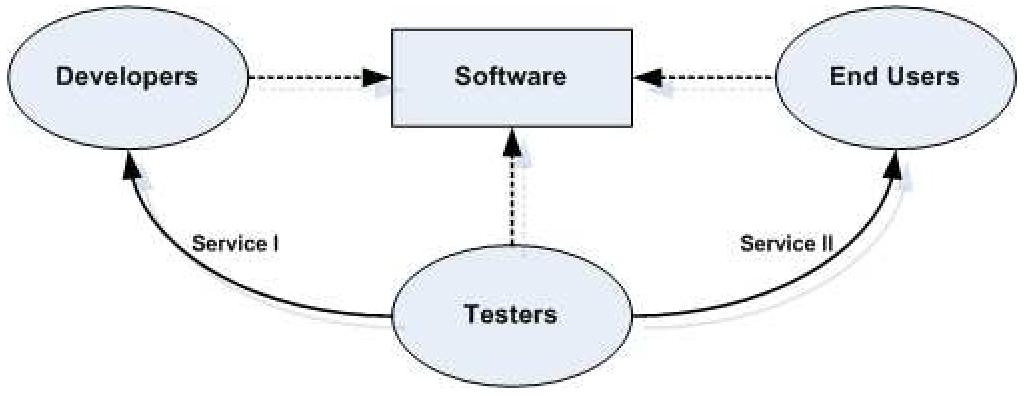

Figure 2. Testers as the Service Providers

SERVQUAL: A REVIEW

Service Quality

This section reviews the literature on SERVQUAL to provide a foundation for developing a structured scale to measure the service quality of software testing. The conventional testing metrics primarily measure testers' productivity and the outcome they deliver in technical terms (e.g., numbers of fixed bugs). The social aspect of testing service has largely been ignored. We advocate that tester provide service from two distinct yet complementary perspectives (to developers and end users) and that testing as a service includes both technical and social aspects of testing service quality.

There are two primary perspectives of service quality in the literature. One is the Nordic perspective (Grönroos, 1982, 1984), which uses global terms to define service quality in teams of functional quality and technical quality. The alternative perspective, prominently used in North America, emphasizes service encounter characteristics that are perceived by the service recipient – such as tangibility, reliability, responsiveness, empathy, and assurance (Parasuraman, Zeithaml, and Berry, 1988, 1991). SERVQUAL includes a widely-utilized measurement instrument for assessing service recipients' expectations and perceptions of service deliveries. Nyeck, Morales, Ladhari, and Pons (2002) have noted that the SERVQUAL measuring tool remains the most complete attempt to conceptualize and measure service quality. Our present exploration of service quality in the software testing context is primarily based on this SERVQUAL perspective.

Parasuraman, Zeithaml, and Berry (1988, 1991) initially developed the SERVQUAL instrument to measure the gap between customer expectations and services received, based on interviews and focus group meetings with managers and customers from large service companies in various industries. They found that customers basically used the same criteria in evaluating service quality and that these criteria spanned virtually all types of service in the consumer and retail sector. This multi-item instrument quantifies customers' global (as against transaction-specific) assessment of a supplier company's service quality. The scale measures the service quality of service personnel along five dimensions: tangibles, reliability, responsiveness, assurance, and empathy. They define these five dimensions as: 1) tangibles - physical facilities, equipment and appearance of service personnel, 2) reliability - ability of service personnel to perform the promised service dependably and accurately, 3) responsiveness - willingness to help and provide prompt service, 4) assurance - knowledge and courtesy of service personnel and their ability to inspire trust and confidence, and 5) empathy – a caring orientation and the provision of individualized attention to service recipients.

There has been some debate over the usefulness of the SERVQUAL's gap measure (perceived service quality minus expected service quality) pertaining to the conceptual and empirical relevance of SERVQUAL versus performance only service scores (SERVPERF). Parasuraman et al. (1991) noted that the SERVPERF scores produced higher adjusted R^2 values when compared to SERVQUAL's gap scores for each of the five dimensions. Van Dyke, Kappelman and Prybutok (1997) also questioned the interpretation and operationalization of the SERVQUAL expectation construct, and the reliability and validity of SERVQUAL dimensionality. In contrast, Pitt, Watson, and Kavan (1995, 1997) demonstrated that the problems of reliability of difference score calculations in SERVQUAL are not nearly as serious as Van Dyke et al. (1997) described, and suggested that the marginal empirical benefit of a perceptual-based (SERVPERF) service quality measure does not justify the loss of managerial diagnostic capabilities found in the SERVQUAL gap measure. Moreover, there are studies that advocate dual levels of service expectations. Parasuraman, Zeithaml, and Berry (1994) recommended two different comparison norms for service quality assessment: one is the level of service a customer believes can and should be delivered, which is "desired service," and the other is the level of service that the customer considers acceptable, which is "adequate service" or "minimum service." The reasons for separating the expected service into these two levels is that customer service expectations are characterized by a range of levels between desired service and minimum service, rather than a single point. The band between desired service and minimum service is termed as the Zone of Tolerance (ZOT). Kettinger and Lee (2005) argued that the ZOT service approach can help provide a more valid basis for diagnosis and judgment concerning service deficiencies and service quality management.

SERVQUAL has been introduced to assess the quality of electronic services. Zeithaml, Parasuraman, and Malhotra (2002) developed an e-SERVQUAL for measuring e-service quality through a three-stage process using exploratory focus groups and two phases of empirical data collection and analysis. The resulting e-SERVQUAL dimensions include efficiency, reliability, fulfillment, privacy, responsiveness, compensation, and contact. Gefen (2002) found that the five standard SERVQUAL dimensions could be reduced to three when conceptualized for the context of online service quality, which are 1) tangible 2) a combined dimension of responsiveness, reliability and assurance, and 3) empathy. Tangibles were found to be the most important dimension in increasing customer loyalty, while the combined "responsiveness" dimension was considered most critical in increasing customer trust. Gefen (2002) also advocated that e-SERVQUAL requires scale development that extends beyond merely adapting offline scales. Variations of the SERVQUAL scale such as WebQual (Liu and Arnett, 2000; Loiacono, Watson, and Goodhue, 2007), SiteQual (Yoo and Donthu, 2001), and eTailQ (Wolfinbarge and Gilly, 2002) have also been developed and applied generally for the contexts of e-retailers providing service to online consumers.

Software Quality

The quality of software developed is a principal concern of every stakeholder in a software development project. Software quality should be interpreted in light of the concept of "purpose of use," considering both internal attributes (product characteristics) and the external attributes (aim of use) (Issac, Rajendran, and Anantharaman, 2003). Therefore, assessment of software quality should take into account multiple viewpoints. From the developers' perspective, software quality can be defined by the conformity to functional and performance requirements that are explicitly fixed, conformity to development standards that are explicitly documented, and conformity to implied characteristics expected of all software developed in a professional manner (Pressman, 1988). From the users' perspective, software quality can be defined with respect to all the properties that satisfy correctly and efficiently the present and future needs of software buyers and users (Issac et al., 2003). These perspectives can be economic as represented by managers, be both technical and social as represented by developers, and be utility-like as represented by users.

The importance of customer-perceived quality is recognized in many fields, especially in manufacturing, marketing, and service organizations; however, there exists no rigorous framework for measuring customer perceptions of software quality. Issac et al. (2003) proposed a conceptual framework for customer-perceived software quality, which is based on the Total Quality Management (TQM) framework and consists of six dimensions: product quality characteristics, operational effectiveness, client focus, process quality, infrastructure and facilities, and employee competence.

Software quality does not improve unless it is measured (Reichheld and Sasser, 1990). Measures of software quality are often difficult to articulate and no single measure is adequate by itself. ISO 9126, developed by International Standards Organization (ISO), is an international standard for the evaluation of software quality. Empirical studies on software quality, although limited in number, all agree that functionality, reliability, usability, efficiency, maintainability, and portability are the six major dimensions of software quality (Barki and Hartwick, 2001; Issac et al., 2003; Ortega, Pérez, and Rojas, 2003).

TESTQUAL: ASPECTS, DEFINITIONS, AND SCALES

Testers are under increasing pressure to demonstrate that their services are customer-focused and that continuous performance improvement is being delivered. Traditional software testing metrics cannot measure up to the developers' and end users' expectations on both testers and software applications. TESTQUAL, as we initiate herein, is a structured scale to measure the quality of testers' service in the context of software testing. It consists of three segments, each intended to measure service quality to developers, service quality to end users, and service quality in relation to software product.

Service Quality to Developers

Even though many tools and automated methods can facilitate much of the testing tasks, the overall testing result ultimately depends on the interpersonal interactions of people producing the software (Cohen et al., 2004). The social interaction between developers and testers becomes evidently necessary in modern testing practice. The measurement for service quality to developers mainly draws upon the SERVQUAL literature, which primarily taps the human aspect of service quality. Five aspects of TESTQUAL that can also be used to measure service to developers are Tangibles, Reliability, Responsiveness, Assurance, and Rapport, as listed in Table 1.

Aspects	Items
Tangibles	I believe the testers be equipped with the right hardware and software to perform the testing service. The testing cases and results were well documented.
Reliability	I believe the testing service performed by the tester(s) is reliable. I believe the tester(s) can perform software testing correctly. I trust the tester(s) to deliver the kind of service I need.
Responsiveness	The tester responds to my needs. In case of any problem, the tester will give me prompt service. The tester will address any of my concerns regarding to the software product.
Assurance	I feel confident about the testing service provided by the tester(s). The tester(s) had answers to all my questions about the testing.
Rapport	The tester(s) can address my various and specific needs. I built a good relationship with the tester(s) through effective communication. The tester(s) and I can resolve conflicts easily.

Table 1. TESTQUAL – Service Quality to Developers

The definition of Tangibles, distinctive from the one by Parasurman, Zeithaml, and Berry (1988, 1991), refers to a set of attributes about the physical testing environment, testing deliverables, and documentation. Definitions for Reliability, Responsiveness, and Assurance are directly adapted from the original SERVQUAL. Reliability refers to the ability of testers to perform the promised service dependably and accurately. Responsiveness refers to testers' willingness to help the developers in a timely manner. Assurance refers to knowledge and skills of testers and their ability to inspire trust and confidence of the developers.

The Rapport aspect, similar to Empathy in the original SERVQUAL, refers to caring and individualized attention to the developers. Since developers are focused on the technical nuances of software design and testers are focused on user requirements (Cohen et al., 2004), they can be at odds with each other, and conflict between developers and testers can arise.

Conflict can adversely affect work process, quality of work, and quality of work life, and thus rapport-building is key to the quality of service rendered to developers.

Service Quality to End Users

This aspect of service relates to confirming the expectations of users regarding software quality. This can be called an "indirect" testing service since it is performed through the medium of the software product with which the users interact. The dimensions that are investigated under this section pertain to three individual characteristics of service quality: tangibles, agency oversight, and user satisfaction. The items are listed in Table 2.

Tangibles refer to the set of attributes about the appearance of software that users come in direct contact with, as well as with the actual software deliverables and documentation.

Testers are agents of end users to control software quality. They need to guarantee that software perform consistently, accurately, and securely according to the expectation of users. Agency oversight pertains to the testers' overall responsibility to ensure that all attributes related to and characteristic of the software and its use are of high quality, meet the requirements of the users, and are connected to real business needs.

User satisfaction is the users' holistic view of software quality that is assured by testers. Alpha and Beta testings put the users in direct contact with the software and allow their satisfaction to be measured. This is the ultimate test of software before it is released to production. The testers have a paramount role during this stage, and the service provided by them deals with final error checking and any concluding validation and verification issues.

Aspects	Scales
Tangibles	The interface was well designed so that it is easy to use. The software was well documented so that I can pursue help on my own easily. The software's interface is pleasing aesthetically.
Agency Oversight	The testers did a good job in terms of catching bugs. The testers ensured that the overall quality of the software was improved and that it meets my needs. Testing ensured that the software meets the business needs of the organization.
User Satisfaction	I am satisfied with performance of the software that is warranted by testers. Testers assure that the software can meet my needs. Overall, I am happy about quality of the testing service.

Table 2. TESTQUAL – Service Quality of Indirect Service to End Users

Software Quality Scales

Software products are expected to demonstrate all the attributes that can measure up to pre-established standards, and testers have a key service responsibility in this regard. Many of these attributes pertain to technical standards that have to be met for certification to be awarded (e.g., ISO 9126). Often in specific industries where software failure may mean life or death consequences, certified testers have to confirm that implemented software products meet strict thresholds for reliability, security, and stability. As aforementioned, the six major dimensions of software quality include functionality, reliability, usability, efficiency, maintainability, and portability (Barki and Hartwick, 2001; Issac et al. 2003; Ortega et al., 2003), as listed in Table 3.

Functionality is a direct reflection of the ability of the software to perform the intended tasks for which it is developed. Testers ensure that the capabilities of the software are adequate to perform the intended tasks that the software is supposed to support.

Reliability investigates the capability of the software to maintain its performance level over different conditions and at different times. Functionality looks at performance, whereas reliability looks at sustainability of performance.

Usability is the degree of effort required for learning and using the software by its intended users. The software is checked to see if typical users are able to properly operate the software. The service provided here has two parts because it looks at the software itself, and also takes into account the users' interaction with the software.

Efficiency refers to the performance and resource use of the software given certain circumstances. Testers verify the responsiveness of the software when answering requests from the users, as well as the responsiveness of the interface. They also check the new system's resource requirements and load footprint. This ensures that the new system will function smoothly and efficiently in various operating environments.

Maintainability pertains to the modularity of software design and the effort required to modify or upgrade the software with additions and improvements.

Portability is the degree to which the software is transferable to multiple environments and configurations. Testers provide a service here that pertains to ensuring that, given future business and technological development, the software developed will allow easy installation/reinstallation, can be replaced or transferred with minimum effort, and can easily adapt to new environments.

Aspects	Scales
Functionality	It is easy to tell whether the software is functioning correctly. The software can recover from errors, accidents, and intrusions. The software can maintain data security and integrity.
Reliability	The software is reliable (it is always up and running, runs without errors, and does what it is supposed to do). The software has the ability to prevent (or minimize) damage, when an error is encountered during the execution of a program.
Usability	The software is easy to use. The software is easy to learn.
Efficiency	The software performs its functions quickly. The software does not overtax system resources during its operation.
Maintainability	The software can easily be modified to meet changing user requirements. The software can easily be adapted to a new technical or organizational environment. The software is easy to maintain.
Portability	The software can easily be operated on different operating systems. The software can easily be adapted to organizational environment.

Table 3. Software Quality

TESTQUAL and Software Quality

The focus of this paper is to propose the two key dimensions of TESTQUAL: testing service that is perceived by developers and testing service that is perceived by testers. The very next step is to verify whether TESTQUAL scale can demonstrate the prerequisite reliability and validity by conducting the questionnaire consisting of the items that were listed in table 1 and table 2. As prior research (eg., Devaraj, Matta, and Conlon, 2001) advocates that SERVQUAL is positively related to product quality, we argue that TESTQUAL is a significantly positive factor in determining software quality in the software development context. Linking TESTQUAL with other constructs is necessary to check the nomological validity of this newly adapted scale.

CONCLUSION

We posit a new perspective of viewing testers as the service providers, which is different from the paradigm where testing is a simple technical verification and validation function that does not tie business needs with technical implementations. This new "testing as a service" perspective, in our contention, will improve software development processes by allowing structured and implementable input from all development stakeholders. This new paradigm helps testers, developers and end users recognize that they are highly interdependent on one another and can be in a win-win situation. When the testers have a clear sense of the expectations and perceptions of other stakeholders including developers and end users, they are more likely to provide adequate and even superior services to their "customers," thus the project team as a whole is one step closer to successfully completing a software development project. Management needs not only to support the testing effort financially, but also to build and promote a "testers-as-service-providers" climate in software development project teams.

TESTQUAL is an implementable tool that will allow IT and business managers to quantify the level of service quality offered by their testing function. It can be administered to the receivers of testing services at multiple intervals in the development processes. Once the perceived level of service, gap, or ZOT, of service quality is quantified, managers can identify areas of deficiency. In particular, TESTQUAL might provide richer information than perception-only or expectation-minus-perception (ZOT) scores. Accordingly, managerial corrective measurements can be taken to address the testing service deficiencies or failures. TESTQUAL can also assist testers in identifying cost-effective ways of closing service quality gaps and in prioritizing which gaps to focus on, given the constraint of scare resources. It can also serve as a structured guideline for training newly hired testers in terms of the technical aspects of delivering quality testing service and also make them aware of the social aspects of interaction with other important stakeholders in software development projects.

REFERENCES

1. Barki, H. and Hartwick, J. (2001) Interpersonal conflict and its management in information system development, MIS Quarterly, 25, 2, 195-228.
2. Bentley, J. E. (2005) Software testing fundamentals – Concepts, roles, and terminology, *Proceedings of SAS Conference*, Philadelphia, Pennsylvania, 1-12.
3. Cohen, C. F., Birkin, S. J., Garfield, M. J., and Webb, H. W. (2004) Managing conflict in software testing, *Communications of the ACM,* 47, 1, 76-81.
4. Davis, J.M., Kettinger, W.J., and Kunev, D.G. (2009) When users are IT experts too: The effects of joint IT competence and partnership on satisfaction with enterprise-level systems implementation, *European Journal of Information Systems*, 18, 1, 26-37.
5. Devaraj, S., Matta, K., F., and Conlon, E.(2001) Product and service quality: The antecedents of customer loyalty in the automotive industry, Production and Operations Management, 2001,Winter.,
6. Gefen, D. (2002) Customer loyalty in E-commerce, *Journal of the Association for Information Systems,* 3, 1, 27-51.
7. Goth, G. (2008) "Googling" test practices? Web giant's culture encourages process improvement, *IEEE Software,* 25, 2, 92-94.
8. Grönroos, C. (1982) Strategic Management and Marketing in the service sector, Marketing Science Institute, Cambridge, MA.
9. Grönroos, C. (1984) A service quality model and its marketing implications, *European Journal of Marketing*, 18, 4, 36-44.
10. Hsu, J. S.C., Chan, C. L., Liu, J. Y.C., and Chen, H. G. (2008) The impact of user review on software responsitiveness: Modeling requirements uncertainty, *Information & Management*, 45, 4, 203-210.
11. Issac, G., Rajendran, C., and Anantharaman, R. N. (2003) Determinants of software quality: Customer's perspective, TQM and Business Excellence, 14, 9, 1053-1070.
12. Jia, R., Reich, B. H., and Pearson, J. M. (2008) IT service climate: An extension to IT service quality research, *Journal of the Association for Information Systems*, 9, 5, 294-320.
13. Kettinger, W. J. and Lee, C. C. (2005) Zones of tolerance: Alternative scales for measuring information systems service quality, *MIS Quarterly,* 29, 4, 607-623.

14. Liu, C. and Arnett, K. P. (2000) Exploring the factors associated with Web site success in the context of electronic commerce, *Information and Management,* 38, 1, 23-34.
15. Loiacono, E. T., Watson, R. T., and Goodhue, D. L. (2007) WebQual: An instrument for consumer evaluation of web sites, *International Journal of Electronic Commerce,* 11, 3, 51-87.
16. Mahmood, M. A. (1987) Systems development methods: A comparative investigation, *MIS Quarterly*, 11, 3, 293-311.
17. Nyeck, S., Morales, M., Ladhari, R., and Pons, F. (2002) 10 years of service quality measurement: Reviewing the use of the SERVQUAL instrument, *Cuadernos de Difusion,* 7, 13, 101-107.
18. Ortega, M., Pérez, M., and Rojas, T. (2003) Construction of a systemic quality model for evaluating a software product, *Software Quality Journal*, 11, 3, 219-242.
19. Parasuraman, A, Zeithaml, V. A., and Berry, L. L. (1988). SERVQUAL: A multiple item scale for measuring consumer perceptions of service quality. *Journal of Retailing, 64(1)*, 12-40.
20. Parasuraman, A., Zeithaml, V. A., and Berry, L. L. (1991) Refined and reassessment of the SERVQUAL scale, *Journal of Retailing,* 67, 4, 420-450.
21. Parasuraman, A., Zeithaml, V. A., and Berry, L. L. (1994) Reassessment of expectations as a comparison standard in measuring service quality: Implications for further research, *Journal of Marketing,* 58, 1, 111-124.
22. Pitt, L. F., Watson, R. T., and Kavan, C. B. (1995) Service quality: A measure of information systems effectiveness, *MIS Quarterly,* 19, 2, 173-187.
23. Pitt, L. F., Watson, R. T., and Kavan, C. B. (1997), Measuring information systems service quality: Concerns for a complete canvas, *MIS Quarterly,* 21, 6, 209- 221.
24. Pressman, R. (1988) Software engineering: A beginner's guide, McGraw Hill, New York.
25. Reichheld, F. and Sasser, W. (1990) Zero defects: Quality comes to services, *Harvard Business Review,* 68, 5, 105-111.
26. Van Dyke, T. P., Kappelman, L. A., and Prybutok, V. R. (1997) Measuring information systems service quality: Concerns on the use of the SERVQUAL questionnaire, *MIS Quarterly*, 21, 2, 195-208.
27. Whittaker, J. A. (2000) What is software testing? And why is it so hard? *IEEE Software*, 17, 1, 70-79.
28. Wolfinbarger, M. F. and Gilly, M. C. (2002) .comQ: dimensionalizing, measuring and predicting quality of E-tail experience, working paper no. 02-100, Marketing Science Institute, Cambridge, MA.
29. Yoo, B. and Donthu, N. (2001) Developing a scale to measure the perceived quality of an Internet shopping site (SITEQUAL), *Quarterly Journal of Electronic Commerce,* 2, 1, 31-45.
30. Zeithaml, V. A., Parasuraman, A., and Malhotra, A. (2002) Service quality delivery through web sites: A critical review of extant knowledge, *Journal of Academy of Marketing Science,* 30, 4, 362-375.