December 1998

A Theoretical Model for Software Quality Management

Paul Ambrose
Southern Illinois University at Carbondale

Jim Eynon
Southern Illinois University at Carbondale

A Theoretical Model for Software Quality Management

Paul J. Ambrose
James W. Eynon
Southern Illinois University at Carbondale

Abstract

The focus of software quality management is on systems development quality. This paper takes a view that such an approach is short sighted and that at it best can lead to customer satisfaction in the short term. To develop quality software with negligible information systems outage on account of software quality problems, two other important factors - system-in-use quality and environmental quality are to be concurrently considered with systems development quality. This paper proposes that software quality should be measured in terms of customer satisfaction, and that the key factors influencing customer satisfaction are 'delivered system quality' 'system-in-use quality' and 'environmental quality'. A theoretical model is built around these three factors to help software quality management.

Introduction

Software manufacturers are reluctant to launch new products unless they are fully satisfied with the quality characteristics of the product. With no comprehensive framework to guide software quality management, the software manufacturers find it difficult to reduce software development time. Almost every major software manufacturer releases a free 'beta' version for customer evaluation before a product is launched. The existence of numerous 'beta' versions is a strong indicator that there is no comprehensive framework to guide software quality management practices, and that the manufacturers have to look outside their organizations to customers for help in developing quality products.

Traditionally software quality was equated with product quality and this did not help much in reducing the cycle time in the introduction of new products. With the emergence of new quality practices promoting the view that process quality was an antecedent of product quality, (Deming, 1986; Shingo, 1986; Taguchi, 1986) the focus of software quality management shifted to systems development quality. Systems development process was identified as the critical path in getting new products of good quality quickly into the market (Rockart and Hofman, 1992).

However, this paper takes the view that a software quality management approach focusing just on system development quality is short sighted and that it can at best lead to customer satisfaction in the short term. To develop quality software with negligible information systems outage on account of software quality problems, two other important factors - system-in-use quality and environmental quality are to be concurrently considered with systems development quality. This paper proposes that software quality should be measured in terms of customer satisfaction, and that the key factors influencing customer satisfaction are delivered systems quality, systems-in-use quality, and environmental quality. A theoretical model is built around these three factors to help software quality management. The developed model should be useful, not just to commercial software manufacturers, but also to information systems departments that develop in-house software to support organizational informational needs.

he paper is organized as follows. First, software quality, the dependent variable of the study and the three factors influencing software quality are defined. This section presents the model developed in the paper. Second, the limitations of the focus on just systems development quality are highlighted by comparing and contrasting key systems development variables from a system-in-use perspective. The final section concludes with implications of the model developed.

Model for Software Quality Management

This paper defines software quality from the customer satisfaction perspective in line with the Total Quality Management philosophy. Accordingly, customer satisfaction is used as an indicator of software quality. Dr. Noriaki Kano, had identified three components collectively contributing to customer satisfaction - dissatisfiers, satisfiers and exciters (Evans and Lindsay, 1996). Dissatisfiers are quality characteristics that by their presence do not add to the satisfaction of the consumer, but by their absence cause dissatisfaction. Satisfiers are those quality characteristics that help induce satisfaction among users. Exciters are those quality characteristics that the customer has not visualized and whose appearance in the software will truly delight the customer. Quality characteristic that are exciters at a period in time need not continue to remain as exciters. They usually become satisfiers and then end up as dissatisfiers. Thus the composition of satisfiers, dissatisfiers and exciters are constantly changing.

A customer is said to be satisfied if the software both satisfies and excites him or her, and if there are no missing quality characteristics that will induce dissatisfaction. Three factors that are crucial in ensuring the presence of dissatisfiers, satisfiers and exciters, are 'delivered system' quality, system-in-use quality and environmental quality. Figure 1 presents the theoretical model proposed based on these three factors.

Delivered system quality refers to product quality that was present at the time the customer acquired the product for use. It consists of inherent and perceived product qualities. Inherent product quality refers to the value of the product in terms of its attributes, and perceived product quality, the users' perceptions about the capabilities of the product (Rai and Ravichandran, 1997). A major antecedent to delivered system quality is system development process quality.

System-in-use quality is the ability of the software to meet and exceed customer expectations for the full useful life of the software. Given the high rate of technological obsolescence in hardware and software, the useful life of a product might reduce due to the introduction of other products. It is the duty of the manufacturer to proactively manage the situation by free upgrades and replacements. System development process quality and system environment quality are antecedents to system-in-use quality.

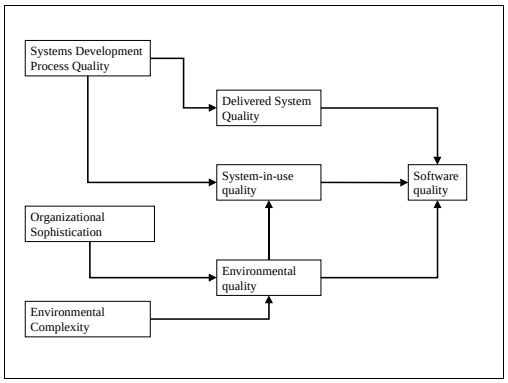

Figure 1. Theoretical Model for Software Quality Management

System environment quality consists of internal and external environmental qualities. Internal environmental quality pertains to the conducive conditions within the organization to enable the smooth functioning of the software. External environment quality refers to the conducive conditions outside the organizational boundary to enable the smooth functioning of the organization and hence the software. Stable organizational and societal structures are antecedents of environmental quality.

Next, the limitations of a systems development process quality approach is highlighted so that the importance of looking at system-in-use and 'environmental quality' is brought out.

Limitations of Systems Development Process Quality Approach to Software Quality

Systems development process quality management ensures that quality products are sent out for use. Typically, systems development process quality consists of inherent and perceived process qualities. Inherent process quality refers to the ability of the systems development process to consistently and efficiently deliver quality systems, and perceived process quality, the users' beliefs about the ability of the process to develop high quality systems (Rai and Ravichandran, 1997). These definitions clearly focus on delivering quality systems for use by customers. But they do not take into account how the system will function after installation. Though the system development process involves user participation in systems development, this does not ensure that quality issues that are essential for a system in use have been adequately addressed.

Quality priorities during systems development process can be different when focusing on just delivering quality products for use as compared to the priorities set with the focus on system-in-use quality. As a result while the system might initially satisfy the customer, dissatisfaction might set in on use with quality characteristics pertaining to system-in-use not taken into account. The difference that might occur on same variable when approached from two different perspectives can be best illustrated using the following systems development process quality variables - user participation, reusability, and reward systems.

User Participation

Users' representatives are usually included in system development teams to ensure that user requirements are captured adequately. The user's typical concerns at this stage are systems conformance to specification and ease of use. However in actual use, the user priorities change to service concerns like maintainability and reparability. Thus from a system-in-use perspective, the systems development process should have considered factors like fail-safing which do not get addressed in detail from a systems development perspective.

Reusability

Program code and design model reusability reduce system development time and thereby increase the efficiency of a software development project. Productivity and quality gains are said to occur from such practices (Biggerstaff and Richer, 1987; Lanergan and Grasso, 1984). Software manufacturers thus approach reusability from the systems development perspective, where efficiency and productivity are prime movers, largely ignoring the dynamics of system-in-use perspective. With the business environment constantly changing, software design model and codes can become obsolete. The process focus will become impervious to such changes and will continue to manufacture software based on modular design models and code. However if reusability was approached from a 'system-in-use' perspective, the emphasis might be on adaptability rather than reusability.

Rewards

The reward structure for software developers is based on productivity, efficiency and 'delivered system' quality. These practices can generate quality products, but are not oriented to ensuring that the product continues to perform to the full customer satisfaction for the entire useful life of the product. It is imperative that the reward structure be designed from the system-in-use perspective where the software developer's rewards are linked more firmly to the quality performance of the software throughout its useful life.

The above arguments should suffice to point out that using systems development processes quality as a surrogate of software quality is inadequate. We now conclude the paper by reiterating the usefulness of looking at software quality management from a much broader perspective.

Conclusion

The paper developed a theoretical model to guide software quality management so that information systems developed are of high quality. With software an important component of information systems, and information systems an important component in an organization's arsenal for survival and growth, software quality takes a high priority in information systems development. System outage on account of software can cause corporate losses. It is imperative that a framework guide software quality to have uniformly high quality software in information systems. The framework that takes into account 'delivered system quality' 'system-in-use quality' and 'environmental quality' as suggested in this paper should help software manufacturers and developers put their products into the market with a reduced development cycle time.

References

References available upon request from first author (paulamb@siu.edu).