Journal of Innovation in Psychology, Education and Didactics	Vol. 16, No. 2 2012	81 - 88

THE INFLUENCE OF THE QUALITY OF SOFTWARE APPLICATIONS UPON THE KNOWLEDGE SOCIETY

Anamaria ŞICLOVAN [a] *

[a] "Vasile Alecsandri" University of Bacău, Mărăşeşti Street, 157, RO-600115, Bacău, Romania, Phd Student at THE School of Economics, "Al. I. Cuza" University, Iaşi

Abstract

Millions of people who use their software applications transmit through them information with a high degree of importance. Thus, it was noted that the use of software applications directly contributes to the formation of the knowledge society. The increasing quality of software use has led to the use of a greater frequency of these applications, which resulted in the transfer of a greater volume of information. We believe that high quality software applications facilitate interaction, a process leading to the development of knowledge society.

Key words*:* interaction, *knowledge society,* quality, society, software quality

1. Introduction

Knowledge society was the main source of information and development of mankind. Software quality is a topic of interest for both software developers and software users. The knowledge society underwent different processes, transformations and disseminations designed to build and apply knowledge for the human development. The importance of the knowledge society and its acceptance are important for the following reasons (UN, DESA, 2005): to recognize the importance of knowledge, to appreciate the work that knowledge influences the human development, to consider the size of the economy changes, to analyze changes in the society, to know what questions to ask so that we can develop the society, to know what to watch, what to accept so as the company is not flooded with the notion of progress, to be informed of the development that will take place.

* *E-mail address*: agapin_ana@yahoo.com

We must be aware of the importance of social development, which leads us to establish a knowledge society based on those individuals and citizens who face the new challenges and opportunities in attaining the knowledge-based economy based on a framework of sustainable development (*Lisbon European Council*, 2000). A special aspect of social media and social networking is the interaction through communication, collaboration and distribution of multimedia data (Doyle, 2010). The distribution of these data leads to the knowledge society. Freedom of expression developed by media, social media, social networks promote the development of knowledge society. The universal access to information and the tend of knowing whatever is new, is the fundamental foundation of the knowledge society development. In the past, information, knowledge and the way of exposing these were the features that differed the strong social and economic groups. Access and inclusion in the knowledge society is the process by which each person has or may have any information he needs. Moreover, the people included in the knowledge society have the required abilities of processing the information acquired in practical knowledge skills useful in their lives (UNESCO, 2010). The developments in social media, including community broadcasting and the World Wide Web have provided new opportunities for individuals and organizations to publish new information and ideas, to learn from others' opinion. We believe that the interaction is a powerful process favouring knowledge society, because in the recent years the level of interaction increased significantly, leading to the transmission of information from one individual to another, the exposing of new knowledge, the acquisition of unique information. In its turn, the interaction is supported and realized by software applications, and their quality facilitates more the connection.

2. Conceptual framework

A precise definition of quality could not be offered and it was considered important the following explanation to understand the concept of software: the quality is the presence of desirable characteristics and the absence of undesirable characteristics in the product or process. This is not a definition but a basic explanation to understand the terms and discuss of the software quality assurance. Features that indicate the presence or the absence of quality are completely dependent on the situation in which each product fits. Basically, the quality is relative. In practical terms, software quality is important because lack of quality translates to user frustration, financial losses, material damages, human injuries and even deaths.

The founders of modern quality assurance, namely Philip B. Crosby (1979) and Joseph M. Juran (1988) were the ones who have shaped the definitions of quality. Quality represents the compliance requirements (Crosby, 1979). That is, the quality relies on those features of the product that meet the customer needs and thereby, provide satisfaction with that product. Juran (1988) stated that quality is characterized by freedom of deficiencies. Crosby's definition of quality refers to the degree in which the writing software meets the specifications developed by the client and the professional team. This meant that software errors included in the specification are not taken into consideration and do not reduce the quality of the software, a feature that is considered poor if put into approach. Juran's definition aimed at achieving customer's satisfaction as well as gathering opinions regarding the real customers' satisfaction, this being in his opinion the true purpose of software quality. The acceptance and adoption of the second definition requires the developer to invest considerable professional effort in examining and correcting, if necessary a customer's requirements. According to the factors influencing the quality stated by McCall (1977), in the initial described factors, there are also the usability factors. Here, we can remind the browsing speed, friendly interface, software adaptation to intentional operations, quality standards. If these factors contribute to a favourable interaction between the software application and the user, then we can say that the interaction is driven by software applications that meet all those beneficial factors.

Another perspective is based on Garvin's (1984) general observation about different approaches to define product quality.

- *Process quality*: Software processes implement best practices of software in an organizational context. Process quality expresses the degree to which defined processes were followed and completed.
- *Product quality*: Software products are the output of software processes. Product quality is determined by the degree to which the developed software meets the defined requirements.
- *Quality in use*: A product that perfectly matches defined requirements does not guarantee to be useful in the hands of a user when the implemented requirements do not reflect the intended use. Quality in use addresses the degree to which a product is fit for a purpose when exposed to a particular context of use.

The Department of Defense (DOD, 1985) in the USA defines software quality as "the degree to which the attributes of the software enable it to perform its intended end use".

Kitchen (1989) refers to software quality “fitness for needs” and claims quality involves matching expectations. There are two features of a piece of quality software: conformance to its specification; fitness for its intended purpose.

According to Standard: IEEE Std 610 (1990), software quality is:

- The degree to which a system, component, or process meets specified requirements.
- The degree to which a system, component, or process meets customer or user needs or expectations.

Software quality assurance is a planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements and a set of activities designed to evaluate the process by which the products are developed or manufactured.

Skogstad (1999) appreciates that over the years, the focus of software quality has shifted. At least three stages can be identified: “the first ages”, “How do we do it?”, “How do we measure quality?”.

“The first ages” – During the first uses of the software, the problems of software quality were felt by developers, and professional users (the batch age).

“How do we do it?” – The problems of software quality very soon focused down to the quest for methods for developing quality software.

“How do we measure quality?” – With the casual software users we have today, a need emerged to have a method for controlled quality in the dissemination of new products. This need made objective software quality measurements necessary.

Software quality (ISO/IEC 1999) refers to the totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs. As a consequence, the ISO/IEC Standard 9126 (2001) and its successor ISO/IEC Standard 25000 (2005) decompose software quality into process quality, product quality, and quality in use. Some approaches are focused on a direct specification of the quality of software product, while the others are focused on assuring high quality of the process by which the product is developed.

Pressman (2003) considers that software quality is conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software. Akingbehin (2005) proposes a new quantitative definition for software quality. The definition is based on the Taguchi philosophy for assessing and improving the quality of manufacturing processes. The proposed definition of software quality shows good correlation to other popular qualitative and quantitative definitions for software quality.

Jones (2012) offers a complex approach to defining the concept of software quality. The main criteria of highlightghing the definitions are structured in Table 1. There are identified two types of definitons: basic and economic definitions (Jones and Bonsignour, 2011).

Table 1. *Directions of defining software quality*

Criteria	Categories	Definitions
Basic definitions of software quality	Functional Software Quality	Software that combines low defect rates and high levels of user satisfaction. The software should also meet all user requirements and adhere to international standards.
	Structural Software Quality	Software that exhibits a robust architecture and can operate in a multi-tier environment without failures or degraded performance. Software has low cyclomatic complexity levels.
	Aesthetic Software Quality	Software with elegant and easy to use commands and interfaces, attractive screens, and well formatted outputs.
Economic definitions	"Technical debt"	The assertion (by Ward Cunningham in 1992) that quick and careless development with poor quality leads to many years of expensive maintenance and enhancements.
	Cost of Quality (COQ)	The overall costs of prevention, appraisal, internal failures, and external failures. For software these mean defect prevention, pre-test defect removal, testing, and post-release defect repairs. (Consequential damages are usually not counted.)
	Total Cost of Ownership (TCO)	The sum of development + enhancement + maintenance + support from day 1 until application is retired. (Recalculation at 5 year intervals is recommended.)

It results that the concept of software quality is more complex after analyzing the definitions of concept. Within the software quality area, the need to provide a solution that matches user needs is often considered as "design quality", whilst ensuring a match to the specification is considered as "manufacturing quality". In the context of the present study, there is the need to create a new perspective upon the definition of software quality in connection with the interaction.

3. Software quality versus interaction

The modern theory of the quality states this concept in its dynamics and links the quality of the economic-informatics application to quality interaction. If the modern society is a society of quality, then it is equally a society of computerization and information technology, i.e. of the interaction information (Ivan, 1999). As the importance and size of the knowledge society increases, the study of the quality of the information applications and their innovation is increasingly becoming influential. The performing services rely on the quality of information applications and are distributed regionally, nationally and globally, representing a real income for many organizations. Their role is to improve performance. Like these services, the interaction has an increased software quality. In Figure 1 we can see how the interaction and software quality directly contributes to the formation and the development of the knowledge society.

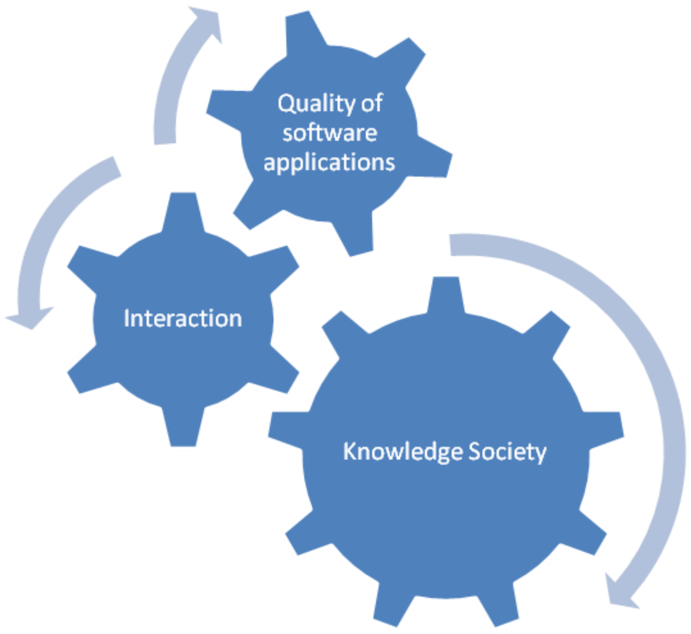

Figure 1. *The influence brought to knowledge society by interaction and software quality*

If achievement of software applications meet the requirements developed by the user and if he uses these applications for information, communication, we can say that a proper software quality influences the interaction. And, as interaction influences the development of the knowledge society, we can deduce that a high quality of software applications leads to a better development of knowledge society.

Conclusion

We all think that we are heading towards a society dominated by interaction, but we do not have the concrete measurement tools of this level. We observed that the software applications, through their characteristics, support software interaction and develop the knowledge society on software applications as technologies and as well as information and communication process. As arguments for this theoretical article, we explained how the interaction facilitates information development in the knowledge society, and then the way in which such interaction is supported by quality software. Thus, the assessment value of the software quality applications, along with the interaction process developed in the virtual environment contributes to the forming of capable individuals of knowledge society. If we discuss interaction in the knowledge society, we propose software developers to consider standards that ensure the quality of these applications, and so the development of global information society will be a key feature of the society we live in.

References

Akingbehin, K. (2005) . A quantitative supplement to the definition of software quality. In *Software Engineering Research, Management and Applications, Proceeding of Third ACIS International Conference*, 11-13 August 2005 (pp. 348-352).

Crosby, P. (1979). *Quality is Free*. New York: McGraw-Hill.

DESA, U. (2005). *Understanding Knowledge Societies*. Department of Economic and Social Affairs.

Garvin, David (1984). What does 'product quality' really mean?. *Sloan Management Review*, Fall, pp. 25–45.

ISO/IEC (1999). *ISO/IEC 9000:2000 Quality management systems - Fundamentals and vocabulary* . Geneva, Switzerland: International Organization for Standardization.

ISO (2001). *ISO/IEC 9126 -1, Software Engineering – Product Quality – Part 1: Quality model*, Geneva: International Organization for Standardization.

ISO/IEC 25000 (2005). Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) - Guide to SquaRE.

Ivan, I. (1999). Managementul calităţii software. Revista *Informatică Economică*, No. 12, 37-42.

Jones, C., & Bonsignour, O. (2011). *The Economics of Software Quality*. Addison Wesley.

Jones, C. (2012). *Software Quality in 2012: A Survey of the State of the Art*. Namcook Analytics LLC.

Juran, J. (1988). Juran's Quality Control Handbook, 4th edn, J.M. Juran, Editor in Chief; I.M. Gryne, Associate Editor. McGraw-Hill, New York.

Kitchen ham, B., Kitchen ham, A., and Fellows, J. (1986). The Effects of Inspections on Software Quality and Productivity. *ICL Technical Journal*.

Layton, Edwin T. (1974). Technology as Knowledge. *Technology and Culture* 15, pp. 31–41.

Lisbon European Council (2000).

Mokyr, J. (2003). The Knowledge Society: Theoretical and Historical Underpinnings, Presented to the Ad Hoc Expert Group on Knowledge Systems, United Nations, New York, Sept. 4-5.

Mccall, J.A., Richards, P.K. & Walters, G.F. (1977). Factors in software quality, Vol. I-III, Rome Air Development Centre, Italy MURINE, G. AND CARPENTER, C. (1984)

Moran, T. P. (2010). *Introduction to the history of communication: evolutions and revolutions*. Peter Lang Publishing, N.Y.

Skogstad, Ø. (1999). Software quality. *Telektronikk*, Vol. 95, No. 1, 3-11.

UNESCO (2010). *Towards Inclusive Knowledge Societies*.

*** (1990). *Standard: IEEE Std 610 – IEEE*. Standard Glossary of Software Engineering Terminology.