Measuring Software Quality in Use: State-of-the-Art and Research Challenges

ISSA ATOUM AND CHIH HOW BONG
Universiti Malaysia Sarawak

Software quality in use comprises quality from the user's perspective. It has gained its importance in e-government applications, mobile-based applications, embedded systems, and even business process development. Users' decisions on software acquisitions are often ad hoc or based on preference due to difficulty in quantitatively measuring software quality in use. But, why is quality-in-use measurement difficult? Although there are many software quality models, to the authors' knowledge no works survey the challenges related to software quality-in-use measurement. This article has two main contributions: 1) it identifies and explains major issues and challenges in measuring software quality in use in the context of the ISO SQuaRE series and related software quality models and highlights open research areas; and 2) it sheds light on a research direction that can be used to predict software quality in use. In short, the quality-in-use measurement issues are related to the complexity of the current standard models and the limitations and incompleteness of the customized software quality models. A sentiment analysis of software reviews is proposed to deal with these issues.

Key words

ISO 25010, quality issues, sentiment analysis, software quality in use, SQuaRE series

INTRODUCTION

With a large amount of software published online it is essential for users to find the software that matches their stated or implied needs (quality in use). Users often seek adequate software quality. It is important to quantify software quality from a user's perspective in order to compare different types of software, thus allowing users to acquire the *best* software quality. To understand what is meant by inadequate or adequate software quality, it helps to define quality. Defining quality is not easy in that quality is based on many possible disciplines: philosophy, economics, marketing, and so on. Garvin (1984) identified five views/approaches of quality. The closest definition to this work is the user-based approach definition "meeting customer needs." As referred to by Deming (2000), Shewhart (1931) defines quality: "There are two common aspects of quality: One of them has to do with the consideration of the quality of a thing as an objective reality independent of the existence of man. The other has to do with what we think, feel or sense as a result of the objective reality. That is to say, there is a subjective side of quality." This side is yet another meaning of *quality in use*. If the customer is satisfied, then a product or service has adequate quality.

Quality in use or perceived quality by users is very important to many applications. Quality in use is gaining more attention, especially in applications where end users are the core of a successful project. It has been adopted in mobile-based applications (Alnanih, Ormandjieva, and Radhakrishnan 2014; La and Kim 2013; Osman and Osman 2013), Web applications (González et al. 2012; Orehovački, Granić, and Kermek 2013; Orehovački 2011), healthcare applications (Alnanih, Ormandjieva, and Radhakrishnan 2013; Alnanih, Ormandjieva, and Radhakrishnan 2014), project management tools (Oliveira, Tereso, and Machado 2014), and business process management (Heinrich, Kappe, and Paech 2011; Heinrich 2014).

There are many benefits that can be gained if quality is adopted early in the software development life cycle. Quality in use, a user-centered approach, will get users' feedback early; thus, different ways to access software, including content and user interface, are adopted early, hence software success. A good software design will let the user work effectively and efficiently, saving time; thus, it will increase user productivity. Moreover, if the user interface (one aspect of quality in use) takes into account user needs and expectations, it will reduce system errors. On the other hand, if the system interface is poorly designed, it will result in increased training time and malfunction errors. On one hand, given these benefits, it is more likely that the user will use the application; on the other hand, software vendors will get improved user acceptance. In fact, systematic evaluation of quality in use is important because it allows continuous software recommendations and improvements.

Moving from the general concept of quality of software, software quality can be conceptualized from three dimensions: software quality requirements, the quality model, and quality characteristics. Quality requirements are what the user needs in the software, such as performance, user interface, or security requirements. The quality model is how characteristics are related to each other and to final product quality. Measuring software quality will check if user requirements are met and determine the degree of quality. The final objective of software quality is to know whether the software is of adequate or inadequate quality. To identify the so-called adequate or inadequate quality, a software quality model is needed. The quality model categorizes quality into characteristics overseen by measurement methods. The quality model is a "defined set of characteristics and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality" (ISO/IEC 2005, 7). The quality model identifies the adequate or inadequate quality attributes. The quality characteristic is the "category of software quality attributes that bears on software quality" (ISO/IEC 2005, 9), simply the main factors or properties that determine whether quality is adequate or inadequate. Measurements are "sets of operations having the object of determining a value of a measure" (ISO/IEC 2011, 52). That is to say, the measurement is the actual score of adequate and inadequate quality attributes.

The ISO/IEC 25010:2010 standard (referred to as ISO 25010 hereafter), a part of a series known as software quality requirements and evaluation (SQuaRE), has two major dimensions: quality in use (QinU) and product quality. The former specifies characteristics related to the human interaction with the system and the latter specifies characteristics intrinsic to the product. QinU is defined as the "capability of a software product to influence users' effectiveness, productivity, safety, and satisfaction to satisfy their actual needs when using the software product to achieve their goals in a specified context of use" (ISO/IEC 2005, 17). The QinU model consists of five characteristics: effectiveness, efficiency, satisfaction, freedom from risk, and context coverage. Figure 1 illustrates the definition of these characteristics.

FIGURE 1 Definitions of quality-in-use characteristics as defined by the ISO 25010 standard

Characteristic	Definition
Effectiveness	Accuracy and completeness with which users achieve specified goals (ISO 9241-11).
Efficiency	Resources expended in relation to the accuracy and completeness with which users achieve goals (ISO 1998).
Freedom from risk	Degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment.
Satisfaction	Degree to which user needs are satisfied when a product or system is used in a specified context of use.
Context coverage	Degree to which a product or system can be used with effectiveness, efficiency, freedom from risk, and satisfaction in both specified contexts of use and in contexts beyond those initially explicitly identified.