Software Quality Journal, 13, 17–30, 2005

The Evolution Path for Industrial Software Quality Evaluation Methods Applying ISO/IEC 9126:2001 Quality Model: Example of MITRE's SQAE Method

MARC-ALEXIS CÔTÉ, WITOLD SURYN and CLAUDE Y. LAPORTE
marcalexis_cote@yahoo.ca, {wsuryn, claporte}@ele.etsmtl.ca
IQUAL, École de technologie supérieure, 1100, rue Notre Dame Ouest, Montréal, Canada

ROBERT A. MARTIN ramartin@mitre.org
MITRE Corporation, 202 Burlington Road, Bedford, MA, USA

Abstract. This paper examines how the industrial applicability of both ISO/IEC 9126:2001 and MITRE Corporation's Software Quality Assessment Exercise (SQAE) can be bolstered by migrating SQAE's quality model to ISO/IEC 9126:2001. The migration of the quality model is accomplished through the definition of an abstraction layer. The consolidated quality model is examined and further improvements to enrich the assessment of quality are enumerated.

Keywords: assessment, measurement, quality factors, quality models, risk management, quality standards

1. Introduction

The software engineering industry has long been diagnosed with a "*quality* problem." This often leads to heated and interesting debates, because what exactly constitutes the *quality* of a product is often the subject of hot debate. Throughout the years, many researchers have proposed their own categorization of software quality, from the early work of McCall (McCall et al., 1977) to the more recent work of Dromey (1995). ISO/IEC 9126:2001 (ISO/IEC, 2001a), the latest revision to the international software product quality standard, attempts to bring this debate a step further towards its conclusion by proposing a quality model issued from an international consensus.

While not everybody agrees about the definition of a quality model, there is no doubt as to the importance of measuring software quality in a systematic and repeatable fashion. Assessing quality is important because it is not quality that is expensive, but rather the lack of quality (Crosby, 1979). The lack of quality can thus be perceived as a risk to the development of software, a risk that should be identified and contained as early as possible in the development life cycle. Tools and models, like MITRE Corporation's (MITRE) *Software Quality Assessment Exercise* (SQAE) have been developed with such goals in mind and have helped a number of development teams over the last decade (Martin, 2003; Martin and Shaffer, 1996).

The assumption supporting this migration experiment is that the industrial applicability of SQAE and the quality of its assessment can be improved by having it rely on the latest version of ISO/IEC 9126. This paper demonstrates approaches for how the SQAE can be migrated to take full advantage of the internationally recognized quality model defined by ISO/IEC 9126:2001. ISO/IEC 9126 and SQAE will both be briefly introduced. These descriptions will be followed by a discussion of how the SQAE can be migrated to ISO/IEC 9126 and how this migration improves the industrial applicability of SQAE.

2. Overview of SQAE

The maintenance of software can account for over 60 percent of all effort expended by a development organization (Pressman, 2001; Manna, 1993). This is in part due to the fact that much of the software we depend on today is more than 15 years old and had to be migrated to different hardware platforms (Osbourne and Chikofsky, 1990). However, most software organizations have traditionally focused on resolving present risks rather than future (and more expensive) risks (Martin, 2003). Short-term risks that are usually the immediate focus of a development or maintenance team include:

- Managing the initial development schedule.
- Containing the development costs.
- Providing desired functionality.

This often results in software that is hard to maintain and entails unforeseen long-term costs. However, as software vendors come and go, IT organizations must make management choices now, choices that they will have to live with for the next 15 years. How are those organizations supposed to assess the risk associated with such an important choice?

In order to provide a satisfying answer to this question, MITRE has created a Software Quality Assessment Exercise (SQAE) providing a set of tools and evaluation methods that give a repeatable and consistent measure of quality of the software and its associated risks (Martin and Shaffer, 1996). The assessment of the quality provided by SQAE focuses on the risk associated with different quality areas and produces a list of risk drivers and mitigating elements that can help software developers and managers reorient their development effort and assist IT organizations in making judicious choices when selecting a software developer and/or maintainer. The SQAE is primarily aimed for third-party evaluations, where an independent group is assessing and evaluating the quality of the software products being developed. By design, the SQAE is very rapid, economical, and the results are independent of the individuals involved in any particular assessment.

The quality model behind the SQAE method is based on the earlier work of Boehm (1978), McCall (McCall et al., 1977) and Dromey (1995) and not on the internationally recognized quality model proposed by ISO/IEC 9126, since the two efforts (SQAE and ISO/IEC 9126) were developed in parallel. This three-layer quality model is composed of 4 quality areas, 7 quality factors, and 76 quality attributes. Figure 1 illustrates how each layer in the model is constructed while Table 1 shows how each quality area is composed of quality factors.

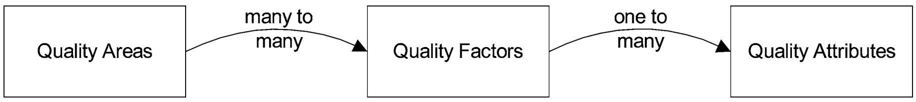

Figure 1. SQAE's three-layer quality model.

Table 1. Relationships between the quality areas and the quality factors in SQAE.

Quality area	Quality factor						
	Modularity	Self-descriptiveness	Design simplicity	Consistency	Documentation	Anomaly control	Independence
Maintainability	×	×	×	×	×	×	
Evolvability	×	×	×		×	×	
Portability	×	×			×		×
Descriptiveness		×			×		

As every attribute attached to a quality factor is measured, a risk profile can be built for each quality factor. An assessment of the quality at the level of the "quality areas" can be constructed from the results obtained from the quality factors.

3. Overview of ISO/IEC 9126

In 1991, the International Organization for Standardization introduced a standard named ISO/IEC 9126 (1991): Software product evaluation—Quality characteristics and guidelines for their use. This standard aimed to define a quality model for software and a set of guidelines for measuring the characteristics associated with it. ISO/IEC 9126 quickly gained notoriety with IT specialists in Europe as the best way to interpret and measure quality (Bazzana et al., 1993). However, Pfleeger (2001) reports some important problems associated with the first release of ISO/IEC 9126:

- There are no guidelines on how to provide an overall assessment of quality.
- There are no indications on how to perform the measurements of the quality characteristics.
- Rather than focusing on the user view of software, the model's characteristics reflect a developer view of software.

In order to address these concerns, an ISO committee began working on a revision of the standard. The results of this effort are the introduction of a revised version of ISO/IEC 9126 focusing on the quality model, and a new standard, ISO/IEC 14598 (ISO/IEC, 1999) focusing on software product evaluation. ISO/IEC 14598 addresses Pfleeger's first concern while the revision to ISO/IEC 9126 aims to resolve the second and third issues. ISO/IEC 9126 is now a four part standard. The first part presents the quality model that addresses the different aspects of quality while the 3 other parts attach measures to the external and internal quality model, as well as to the quality in use model.

Figure 2 illustrates the relationships between quality in use, external quality and internal quality as defined by ISO/IEC 9126.

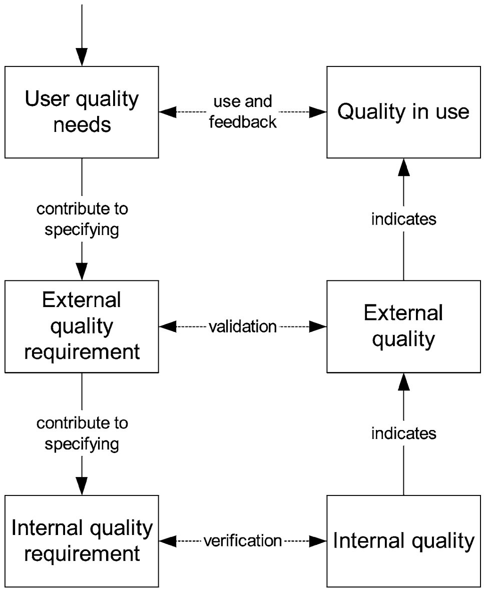

Figure 2. Relationships between the different aspects of quality.

The left-hand side of Figure 2 indicates that requirements discovery and definition should begin by defining user quality needs. In order to define such requirements, the stakeholders may use ISO/IEC 9126-4 as support. Those quality in use requirements can then be used to help in the discovery of external quality needs. This does not mean that *all* external quality requirements should be drawn from the quality in use requirements. Stakeholders may use ISO/IEC 9126-2 as support in defining external quality requirements. The same reasoning applies to the definition of internal quality requirements. The right-hand side of the figure indicates that the measured internal quality can be used to predict external quality, while the measured external quality can be used as a prediction of quality in use. Unfortunately, ISO/IEC 9126-1 stops short of defining what quality attributes are predictive of which.

What is important to remember from the above discussion is that ISO/IEC 9126 is not only a model for use in the *evaluation* of quality, but also a model for use in the *specification* of quality needs.

The aspects for internal and external quality are quite similar. They both rely on a three-layer model composed of characteristics, subcharacteristics and metrics (Figure 3). Of course, the associated metrics are different for internal and external quality. The major difference with the first incarnation of ISO/IEC 9126 is the inclusion of suggested metrics (ISO/IEC, 2003a, 2003b) for measuring each subcharacteristic. It is important to note that these metrics are not normative (i.e., a custom set of metrics can be defined, as long as they conform to annex A of ISO/IEC 9126-1).

Another important addition is a quality in use aspect (ISO/IEC, 2001b). This part of the model aims at defining the quality attributes that are important for the end user and therefore addresses Pfleeger's third concern about ISO/IEC 9126. The quality in use aspect is illustrated in Figure 4.

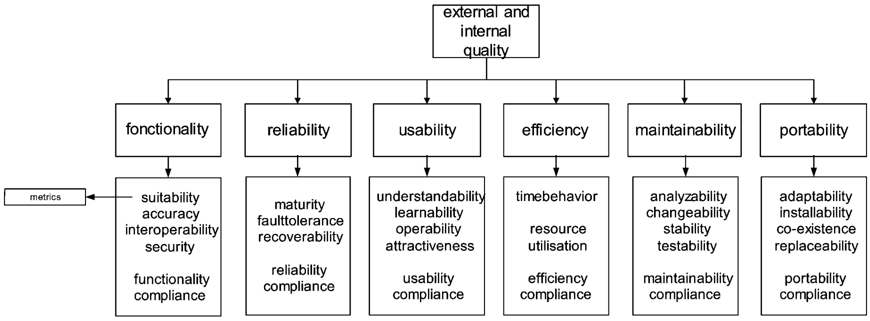

Figure 3. 3-layer model for internal and external quality.

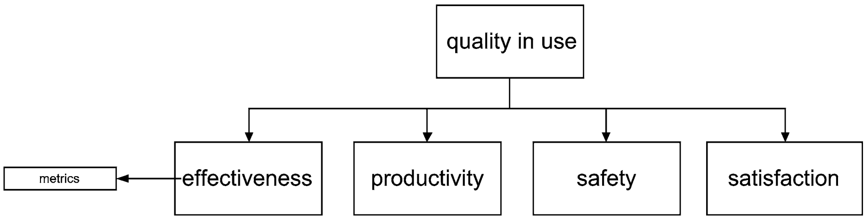

Figure 4. 2-layer model for quality in use.

As is the case with the internal and external quality parts, a set of informative metrics is associated with each quality in use characteristic (ISO/IEC, 2001b). This model is very appropriate for giving an appreciation of the quality as seen from a user's perspective. It may also serve as a starting point for the discovery of quality requirements.

4. Industrial applicability of SQAE and ISO/IEC 9126

SQAE has been used to analyze more than 100 systems. This represents more than 50 million lines of code written in a large number of programming languages, from Assembler to 4GL. It has also been used to assess the quality of systems ranging from 4 thousand lines of code, to more than 6 million lines of code. SQAE has proven through time that it can provide a useful assessment of the quality of a great variety of software packages (Martin, 2003; Martin and Shaffer, 1996). The Department of Defense and other U.S. Government agencies have used SQAE to analyze software quality.

On the other hand, ISO/IEC 9126 is the international standard for software quality that has been agreed on by a majority of the international community and which some countries, like Japan, have adopted as a national standard. It defines a common language relating to software product quality and is widely recognized as such, at least in Europe, where a survey indicates that it is known by at least 70% of the IT community (Bazzana et al., 1993). However, as noted by Pfleeger (Pfleeger, 2001), ISO/IEC 9126 has been confined to usage by the academia and has only seen sparse industrial applications. It is believed that this situation will change with the advent of the revision to the standard.

SQAE and ISO/IEC 9126 can both benefit from a close relationship. In order to gain a wider applicability, SQAE should grow out of its software acquisition risk analysis mold into a full blown assessment of software quality. From the description of both SQAE and ISO/IEC 9126 in preceding sections, it is possible to conclude that SQAE focuses on a partial analysis of *internal quality*, as seen from the perspective of ISO/IEC 9126. While such a model might be sufficient to evaluate software with respect to its original goals, SQAE's analysis of quality may be greatly enriched by relying on ISO/IEC 9126's quality model.

Such a migration of SQAE to the internationally recognized quality model would also be beneficial for ISO/IEC 9126, as it would demonstrate that in can be used in the industry.

5. Defining an abstraction layer between SQAE and ISO/IEC 9126

Rather than completely replacing SQAE's quality model by the one proposed in ISO/IEC 9126, it has been elected to attempt to merge the two models in order to preserve as much of SQAE as possible. Two paths may be envisioned for this unification of the two models:

- Enrich the quality model behind SQAE with new quality attributes in order to make it compliant to ISO/IEC 9126.
- Express SQAE's quality factors as a composition of ISO/IEC 9126's subcharacteristics and borrow its measurement model (see Figure 5).

The first path is clearly the brute force one and not the best way to proceed. Simply adding quality attributes, factors and areas could result in a model that is unwieldy and hard to maintain. Further modifications to ISO/IEC 9126 would inevitably results in changes to the new proposed model. The second path is akin to adding an abstraction layer between SQAE and ISO/IEC 9126 that would insulate SQAE from minor changes to ISO/IEC 9126. There are also other factors that point out the second path as the best solution:

- It will be possible to express the 4 quality areas, composed of 7 quality factors, in the clear and unambiguous language of ISO/IEC 9126.
- By expressing the quality factors as a composition of subcharacteristics, SQAE automatically inherits from an internationally recognized set of metrics.
- It will be possible to apply the measurement methodology both statically (source code, documents, etc.) and dynamically (on an executable image). This is an important advantage that will considerably enrich SQAE. This subject will be explored further in a subsequent section.

In order to construct the abstraction layer, the links between ISO/IEC 9126 and SQAE must be thoroughly understood. A "translation" attempt between the two models has provided the necessary insights to build the abstraction layer.

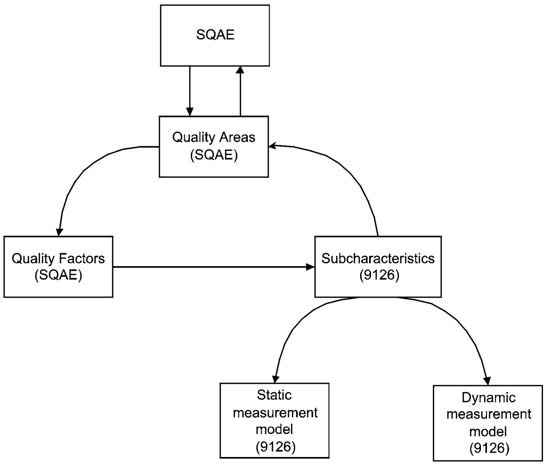

Figure 5. Expressing the quality factors in terms of ISO/IEC 9126's subcharacteristics.

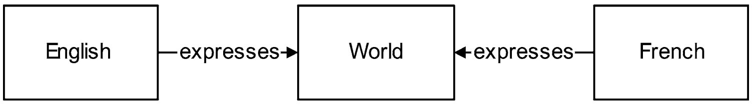

Figure 6. French and English are used to express a common concept.

Figure 7. SQAE and ISO/IEC 9126 are also used to express a common concept.

5.1. Discovering the relationship between SQAE and ISO/IEC 9126 through translation

The ISO/IEC 9126 standard and MITRE's SQAE have one common goal: expressing software quality, an intangible concept, in a language that is understood by all. This context is strikingly similar to one we are more familiar with: the context where two languages, let's say English and French, try to express a common concept (Figures 6 and 7).

Since French and English both express a common concept, it is possible to *translate* from one to the other. The assumption that the same can be done with ISO/IEC 9126 and SQAE, as they also both express a common concept, is the basis of this work. As with the linguistic metaphor, it is quite likely that some concepts will not be easily translatable from SQAE to ISO/IEC 9126 and vice-versa.

The translation attempt that was made has given us essential insight for the adaptation of SQAE to ISO/IEC 9126. The attempt revealed three kinds of issues:

- The first possibility, and the one which is the most desirable, is when there is a

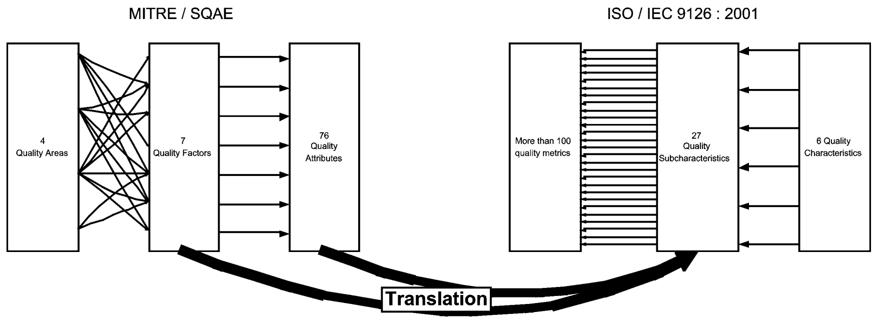

Figure 8. The translation activity.

perfect correspondence (translation) between a concept expressed in SQAE and ISO/IEC 9126.

- A second possibility is when a concept is not easily translatable from one language to the other. However, it might be possible to express the concept using several other concepts or by using more general ones. In such a case, a lost of precision is almost inevitable.
- A last possibility is when no translation is possible between the two languages because there are simply no common grounds or because a notion is totally lacking from the target language.

The first possibility is quite probable, since both SQAE and ISO/IEC 9126 emerged from a common line of thought. The second and third possibilities are also likely, since SQAE and ISO/IEC 9126 have diverging goals. An overabundance of issues that fall in these two categories would justify a *migration* activity, which is defined as a set of modifications that would allow for more clarity and simplicity in expressing a given concept.

As is shown in Figure 8, the translation has been made from SQAE's quality attributes and factors to ISO/IEC 9126's subcharacteristics. It is possible to justify the choice of this level by referring to the linguistic metaphor: the results of a *word* by *word* translation (in this case attributes to metrics) are almost always unsatisfying. It makes more sense to take the quality attributes and factors, which respectively represent the *words* and the *sentences* of SQAE and formulate with them the concepts embodied by ISO/IEC 9126's 27 subcharacteristics.

The results of the translation activity are based on information collected in the ISO/IEC 9126 (ISO/IEC, 2001a, 2003a, 2003b) standard and MITRE's description of SQAE (Martin and Shaffer, 1996). They are presented as a graphic showing how each quality factor and attribute contributes to the expression of a subcharacteristic. The following conclusions can be drawn from this work:

- All of SQAE's attributes contribute to the expression of at least one ISO/IEC 9126 subcharacteristic.
- A large number (18/27 = 66%) of ISO/IEC 9126's subcharacteristics are covered by SQAE's quality attributes.

Those two elements clearly paint SQAE as a language that is not as rich as ISO/IEC 9126. Although most quality characteristics are somehow evaluated by SQAE, 9 are

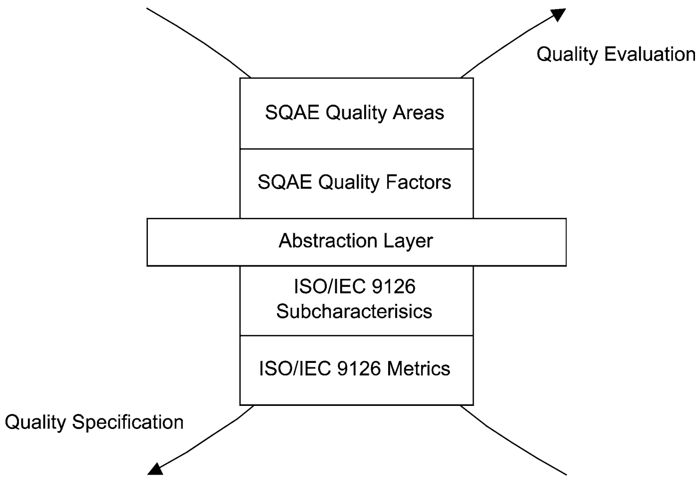

Figure 9. Consolidated quality model.

not covered and for some others the link with SQAE's quality attributes is weak. The quality attributes are thus insufficient to translate completely SQAE to the clear and unambiguous language defined by ISO/IEC 9126.

However, the results of the translation activity present in a clear way the relations that exist between the two quality models and lays down the path for a migration of SQAE to ISO/IEC 9126.

5.2. *The abstraction layer*

Table 2 presents a possible abstraction layer between SQAE and ISO/IEC 9126. The given numerical values are based on the strength of the correlation between the two models that emerged from the translation attempt. For example, the Independence quality factor was found to have a strong correlation to Interoperability, Changeability and Adaptability and a weak correlation to Installability. Twice as much importance was given to the stronger correlations than to the weaker one, resulting in the values below.

From this table, it may be observed that each quality attribute defined in SQAE, with the exception of the ones with the crosshatch pattern, is composed of multiple subcharacteristics as defined in ISO/IEC 9126.

This abstraction layer may be used as the basis for enhancing SQAE. These enhancements are presented below.

5.3. *Consolidated quality model*

From the definition of the abstraction layer, a consolidated quality model on which to base further improvements of SQAE can be defined. The new quality model is illustrated in Figure 9 and explained below.

Table 2. Correlation between SQAE's quality factors and ISO/IEC 9126's subcharacteristics.

ISO 9126		SQAE						
		Consistency (%)	Independence (%)	Modularity (%)	Documentation (%)	Self-descriptiveness (%)	Anomaly control (%)	Design simplicity (%)
Functionality	Suitability	8						
	Accuracy	18						
	Interoperability		30					
	Security							12.5
	Functionality compliance	8						
Reliability	Maturity						20	
	Fault tolerance						20	
	Recoverability						20	
	Reliability compliance	8					20	
Usability	Understandability	8				20		
	Learnability				33			
	Operability					40		
	Attractiveness							
	Usability compliance	8						
Efficiency	Time behaviour							
	Resource utilisation							
	Efficiency compliance	8						
Maintainability	Analysability	18			33	40		25
	Changeability		30	50				25
	Stability						20	12.5
	Testability			50				25
	Maintainability compliance	8						
Portability	Adaptability		30					
	Installability		10					
	Co-existence				17			
	Replaceability				17			
	Portability compliance	8						

This new model relies completely on ISO/IEC 9126 for the measurement of quality, while SQAE provides the evaluation and interpretation of the measurements. The proposed migration of SQAE to ISO/IEC 9126 would transform this method from one that directly measures quality to one that evaluates quality as measured by an international standard. If such a change were to be carried out, it would benefit SQAE in the following ways:

- The quality model and the measurements would be based on an international standard.
- The risk assessment part of SQAE would retain all its value.
- If new aspects of quality are proposed, they could be integrated in the model.

This new quality model is fully compliant with ISO/IEC 9126, because quality is now measured in a standard compliant way.

6. Further improvements

This new model can serve as the basis for further research into the enhancement of SQAE.

6.1. Broader coverage of the different aspects of quality

As was suggested by the definition of the abstraction layer, a number of subcharacteristics from ISO/IEC 9126 are not covered by SQAE. This implies that SQAE does not measure some elements of quality as defined by ISO/IEC 9126, since *Attractiveness*, *Time behavior* and *Resource Utilization* are covered neither by the quality areas or the quality factors. Such a lack of coverage is due to the fact that SQAE was designed as a method to analyze static artifacts (source code, documentation, etc.) and these subcharacteristics are naturally more prone to a static evaluation. However, ISO/IEC 9126 shows us that these three subcharacteristics can indeed be measured statically. It would therefore be desirable to modify SQAE by introducing a new quality area and a few quality factors that would measure these aspects of quality. With respect to the consolidated quality model proposed in Figure 9, this means that the coverage of the upper part of the model must be broadened.

6.2. Evaluation of external quality

SQAE was originally conceived to measure quality statically (internal quality in the terms of ISO/IEC 9126). One of the advantages of the method that was used to accomplish the migration is that the new model inherits the ability to measure quality dynamically (external quality in the terms of ISO/IEC 9126). This is due to the fact that to each subcharacteristic is attached a set of internal and external metrics. Since each quality factor is now composed of subcharacteristics, it follows that SQAE can now measure quality both statically and dynamically.

By using external metrics to measure the subcharacteristics, SQAE can now be used to give an interpretation of the external quality of the software being evaluated. It must be kept in mind that the quality model behind SQAE was created to measure internal quality. Therefore, the following question must be asked: “Do the quality areas and quality factors make sense when evaluating the dynamic aspect of quality?” A full answer to this question is a subject for the next phase of this research program.

6.3. Evaluation of quality in use

In modern history, countless accidents could have been avoided if the interface to a system had been better thought out. In his book on the design of everyday things, Norman (2002) gives a good number of accidents (most notably the Three Mile Island incident) that could have been averted if the quality in use of some systems had been better. One of the primary failings of the first version of ISO/IEC 9126, as well as many other quality models, is the focus on the developer's view of quality at the expense of evaluating the quality from the user's point of view (Pfleeger, 2001). Putting too much focus on internal quality can result in systems that fail at the user interface level, as Norman brilliantly points out in his book with countless examples of interface design failures.

It should be recalled, as is shown in Figure 2, that quality requirements for a system should originate in most cases from the end-user. For any given software, if the user's requirements for quality in use are not met, it poses both a short-term and a long-term risk. The short-term risk emanates from a lack of acceptance by the end-user of the software. It would therefore be useful to evaluate the quality in use of early prototypes in order to shape future development efforts and predict end-user acceptance. The long-term risk comes from maintenance related problems (rework due to poor quality in use), legal liability in accidents caused by poor quality in use, and high training cost (it takes longer to train user's to a nonintuitive system).

One of the most important aspects of the newest version of ISO/IEC 9126 is the integration of a quality in use model (ISO/IEC, 2001a, 2001b). As one of the main goals of SQAE is to assess the risk associated to software, it would be interesting to improve the model by including an evaluation of the risks associated with the quality in use. Further research is needed in order to integrate the evaluation of quality in use intelligently and effectively with SQAE.

7. Reusing the translation metaphor

It is certainly possible to reuse the translation metaphor and the resulting abstraction layer in the migration of other quality models and tools towards ISO/IEC 9126. As per IEEE recommendations (IEEE, 1998), quality models are generally decomposed hierarchically. The current experiment has shown that attempting to join two models at the bottom of such a hierarchy, generally the metrics level, fails to give satisfactory results. Therefore, migration attempts should be carried higher up in the hierarchy. It is important to note that the higher the migration is carried, the more the original model looses of its originality. In the case of MITRE's SQAE, the metrics level was removed and an abstraction layer was defined so that ISO/IEC 9126's metrics could be use instead. This reliance on ISO/IEC 9126 allows SQAE to gain the credibility and notoriety associated with an international standard while allowing it to retain its original interpretation of quality. The same gains could be obtained from the migration of other models. Another benefit of attempting to join two quality models is that it could show deficiencies in the two models. For example, in the case of MITRE's SQAE, the migration attempt has shown that while the coverage of internal quality was considerable, the model failed to acknowledge external quality and quality in use as important aspects of overall quality.

8. Conclusion

Through the usage of a linguistic metaphor, it has been shown that the quality model behind MITRE's Software Quality Assessment does not measure all the aspects of quality as defined by ISO/IEC 9126. A new consolidated quality model that shows greater compliance with ISO/IEC 9126 has been defined with the help of an abstraction layer. This layer helps close the gap between SQAE's quality model and ISO/IEC 9126 and clearly shows areas needing improvements. The new consolidated model allows SQAE to retain its unique evaluation of the risk associated with software while relying on an internationally recognized standard for the measurement of quality.

Based on the consolidated quality model, new research subjects have been proposed to further enhance SQAE. Namely, the coverage of the different aspects of quality should be broadened, the possibility of evaluating quality in a dynamic fashion with the consolidated model should be tested and validated, and finally integration of the evaluation of quality in use should be considered in order to provide a more thorough assessment of risk. Such enhancements can only better the industrial applicability of SQAE.

References

Bazzana, G., Andersen, O., and Jokela, T. 1993. ISO 9126 and ISO 9000: Friends or foes? *Presented at Software Engineering Standards Symposium*.

Boehm, B.W. 1978. *Characteristics of Software Quality*. New York, American Elsevier.

Crosby, P.B. 1979. *Quality Is Free: The Art of Making Quality Certain*. New York, McGraw-Hill.

Dromey, R.G. 1995. A model for software product quality, *IEEE Transactions on Software Engineering* 21: 146–162.

IEEE Std. 1061-1998. 1998. IEEE Standard for a Software Quality Metrics Methodology.

ISO/IEC. 1999. Software product evaluation. Part 1: General overview, ISO/IEC 14598-1, Geneva, Switzerland, International Organization for Standardization.

ISO/IEC. 2001a. Software engineering—Software product quality. Part 1: Quality model, ISO/IEC 9126-1, Geneva, Switzerland, International Organization for Standardization.

ISO/IEC. 2001b. Software engineering—Software product quality. Part 4: Quality in use metrics, ISO/IEC DTR 9126-4, Geneva, Switzerland, International Organization for Standardization.

ISO/IEC. 2003a. Software engineering—Software product quality. Part 2: External metrics, ISO/IEC TR 9126-2, Geneva, Switzerland, International Organization for Standardization.

ISO/IEC. 2003b. Software engineering—Software product quality. Part 3: Internal metrics, ISO/IEC TR 9126-3, Geneva, Switzerland, International Organization for Standardization.

Manna, M. 1993. Maintenance burden begging for a remedy, *Datamation* (April): 53–63.

Martin, R.A. 2003. Managing software risks with metrics and measures. *Proceedings of the Second Annual Conference on the Acquisition of Software-Intensive Systems*.

Martin, R.A. and Shaffer, L. 1996. *Providing a Framework for Effective Software Quality Assessment*. Bedford, MA, MITRE Corporation.

McCall, J.A., Richards, P.K., and Walters, G.F. 1977. Factors in software quality, Griffiths Air Force Base, NY, Rome Air Development Center Air Force Systems Command.

Norman, D.A. 2002. *The Design of Everyday Things, First Basic Paperback*. New York, Basic Books.

Osbourne, W.M. and Chikofsky, E.J. 1990. Fitting pieces to the maintenance puzzle, *IEEE Software* (January): 10–11.

Pfleeger, S.L. 2001. *Software Engineering: Theory and Practice*, 2nd ed. Upper Saddle River, NJ, Prentice Hall.

Pressman, R.S. 2001. *Software Engineering: A Practitioner's Approach*, 5th ed. Boston, McGraw-Hill.

Marc-Alexis Côté is a graduate student at the École de technologie supérieure in Montreal, Canada. His research interests include software quality engineering, software engineering tools, as well as programming languages. Marc-Alexis Côté graduated Suma Cum Laude in computer engineering from the University of Ottawa in 2003. He was awarded a Lucent Global Science Scholarship in 2001.

Witold Suryn is a Professor at the École de technologie supérieure, Montreal, Canada (Engineering School of the Université du Québec network of institutions) where he teaches graduate and undergraduate software engineering courses and conducts research in the domain of software quality engineering, software engineering body of knowledge and software engineering fundamental principles. Dr. Suryn is also the principal researcher and the director of GELOG: IQUAL, the Software Quality Engineering Research Group at École de technologie supérieure.

Claude Y. Laporte is a Software Engineering Professor at the École de technologie supérieure since 2000. After retiring, in 1992, from the Department of National Defense at the rank of major, he joined Oerlikon Contraves to coordinate the development and deployment of engineering and management processes. In 1989, he instigated the development of a software engineering center modeled on the Software Engineering Institute. As a professor, at the Military College of Saint-Jean, he was also tasked to lead the development of a graduate program in software engineering for the Department of National Defense. Professor Laporte is a member of IEEE, INCOSE and PMI.

Robert A. Martin has been with MITRE for 21 years and is a Principal Engineer in its Information Technologies Directorate. In the early 90's Robert and his MITRE team developed a standardized software quality assessment process to help their customers improve their software acquisition and over-sight methods as well as the quality, cost, and timeliness of their delivered software products. Robert's efforts are currently focused on the interplay of cyber security, critical infrastructure protection, and software quality measurement technologies and methods. Robert has a bachelor's degree and a master's degree in electrical engineering from Rensselaer Polytechnic Institute and a master's of business degree from Babson College. He is a member of the ACM, AFCEA, NDIA, and the IEEE.