Published in: *Conference proceedings European Software Quality Week*, October 2000

Measuring software product quality during testing

Rob Hendriks, Robert van Vonderen and Erik van Veenendaal

Quality requirements of software products are often described in vague and broad terms. As a consequence it makes it difficult for software engineers to determine how quality influences their assignment and it is almost impossible for test engineers to evaluate the quality of the software product as no concrete and quantitative reference, of what quality in that context means, exists. This paper describes a possible way to define and measure software product quality. The authors have applied this method during the development of copier/printer controller software at Océ Technologies B.V., a Dutch developer and manufacturer of copying and printing equipment.

Context

In 1997, Océ Technologies started the development of a new line of copier/printer controllers, to be used in a new family of monochrome high-volume hybrid copier/printers. In April 1999 this development entered the engineering phase. The engineering of the software for this controller line takes place on three sites, of which two are in The Netherlands and one is in France. Approximately 60 software engineers are involved in the development of this software, hereafter referred to as the 'controller software'. The controller software is developed in an incremental fashion, with each development cycle conforming to the V-model. Each iteration of such a development cycle takes between 3 and 5 months.

At the start of the engineering phase a test team was formed with the assignment to verify correct functional behaviour and to determine product quality of the controller software. One of the main problems that occurred for the test team was the fact that the required quality level of the controller software was not specified. The only quality requirements available referred to the copier/printer product as a whole and not particular to it's software components. Those requirements were however easily measurable and easy to understand. Examples are the mean number of copies between failures (MCBF, which pertain to all system errors) and the average copies per repair (ACPR, which is the number of copies made between visits of a service technician).

The objective of the test team was to get a clear baseline of the quality requirements before the testing phase would actually start. Therefore a quality model had to be found that could help in defining and measuring software product quality. This model was found in ISO9126 (ISO/IEC 9126-

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

1:2000). In this ISO standard six quality characteristics are defined, which help in decomposing quality in manageable parts. The quality characteristics defined in the ISO9126 standard are functionality, reliability, usability, efficiency, maintainability and portability. To have a more detailed description these quality characteristics are divided into 27 sub-characteristics (Figure 1) in total.

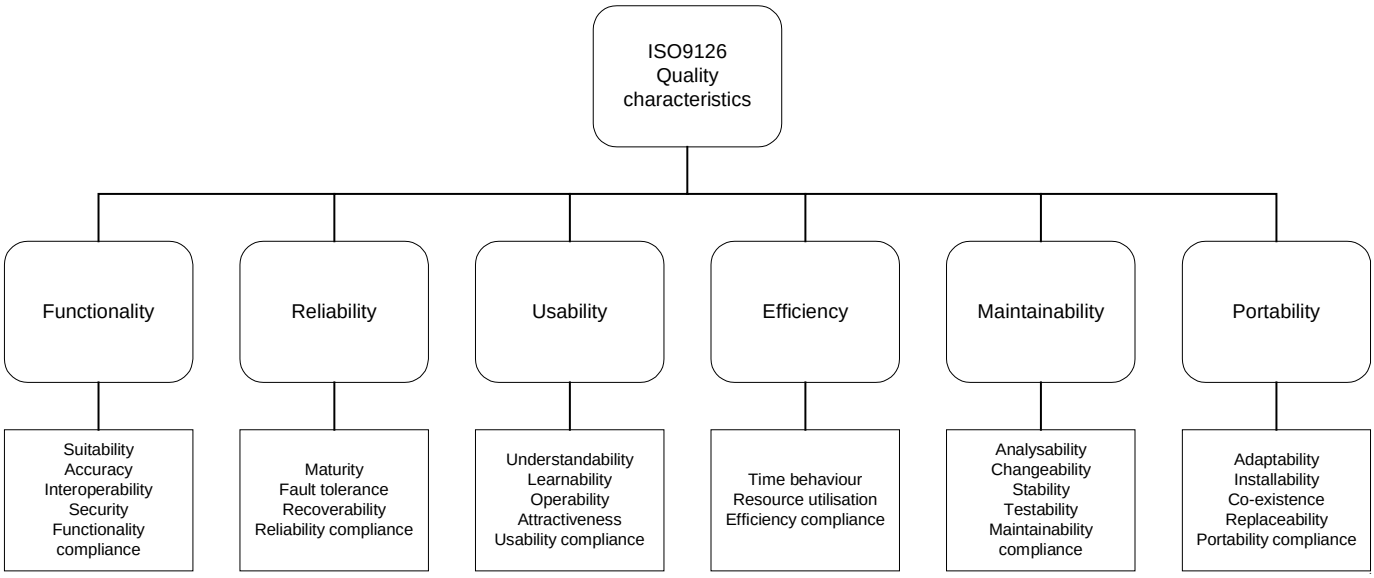

Figure 1 ISO9126 quality characteristics overview

Selection of product quality characteristics

Not all quality characteristics are of equal importance to the software product. Portability might be unimportant when the development aims at a dedicated platform and maintainability might not be an issue when it's a throwaway product. The important quality (sub-)characteristics need therefore be selected. This selection can be made by interviewing key persons in- and outside the project. Inside the project one can think of the product manager, the project manager or the software architect. Outside the project the various types of users are important. Copier users are not only limited to the person operating the copier, often forgotten are e.g. the service technician, the system administrator, etc.

The quality characteristics as defined by the ISO9126 standard are not always easy to interpret. What is meant by maintainability, or even worse usability? It's difficult to express these quality characteristics in an unambiguous way. As it will be hard to understand for IT professionals it will be even harder for users of the copier, who, in general, have no or limited knowledge of software. Most of the users don't even perceive that the product contains software. It will therefore be difficult to determine those quality characteristics that are important for the software component

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

of the product just by asking “Do you think usability is important?” The persons interviewed have their own definition of the characteristics and their own view of what the system should do and therefore what is important.

To overcome the problems mentioned above a questionnaire based method has been developed in the European SPACE-UFO project, funded by the European Union, and further elaborated by Improve Quality Services B.V.. This questionnaire consists of a number of questions about general product characteristics. These product characteristics are understandable for all interviewed persons. Instead of asking whether usability is important one asks questions about the characteristics of the users that influence the usability requirements, e.g. the number of different users, their experience with the product (or a similar one) and their educational level. Similar questions are asked for the other quality characteristics. The answers given are used to deduct the most relevant quality (sub-)characteristics. A fragment of the questionnaire is included in appendix A. This appendix also shows the related quality (sub-) characteristics.

Within the project 4 key-persons were selected and also 3 representatives of the different types of users were selected. Thus 7 persons were interviewed, each having their typical view on the copier/printer product. From the answers given a ranking for the product quality (sub-) characteristics could be deducted. A score from 1 to 5 was given, with 1 being unimportant and 5 most important. The results then were averaged for all respondents. This resulted in the score shown in table 1.

	Score = 1	Score = 2	Score = 3	Score = 4	Score = 5
Functionality				X	
Reliability				X	
Usability			X		
Efficiency		X			
Maintainability			X		
Portability	X				

Table 1 Score per quality characteristic

As can be seen from the results of the questionnaire, the quality characteristics functionality, reliability and maintainability were considered to be important for the controller software. This is explainable from the type of copier the project is developing: it is expected to run in a highly professional document production environment, where the uptime (to be expressed in ‘reliability’ and ‘maintainability’) is of utmost importance. Also, the copier/printer will be the successor product of an analogue high-

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

volume copier model, where a clear functionality demand (being compatible with existing products as well as providing an extension) is expressed. Usability also scores high but it has been decided not to take it into account for testing the controller software. The reason for this is that usability is considered to be a property of the user interface, which is not part of the controller software development.

Given this, the test team decided to focus on three quality characteristics (functionality, reliability and maintainability) primarily, and to develop quality metrics for these quality characteristics.

Definition of quality metrics

When the relevant quality (sub-)characteristics have been determined, one should think about how to measure quality. The ISO9126 standard defines, in the technical reports ISO 9126-2 (external metrics) and ISO 9126-3 (internal metrics), a number of metrics per quality characteristic that can be used for measurement.

A selection of metrics can be made and, when necessary, extended with self-defined metrics. The latter only on condition that the metrics are motivated by defining the goal and the attribute that will be measured.

For each quality sub-characteristic, a selection of metrics from the ISO9126 standard parts 2 and 3 (ISO/IEC 9126-2 and ISO/IEC 9126-3) and from the product requirements was made (e.g. Mean Copies Between Failures). The selection of metrics to be used was mainly based on the fact whether it was possible and easy to measure them, mainly because this was the first experiment with measuring quality characteristics.

Most of the requirements could be used directly from the ISO technical reports and some of them had to be fine-tuned to the project's definitions. An example is the ISO9126 metric Mean Time Between Failures (MTBF). This metric is often used to give an indication of the maturity of the system. For copier/printers the maturity is often indicated by means of the metric Mean Copies Between Failures (MCBF). Therefore a slight change in the metrics could be desirable. Furthermore the product requirements often include quality statements applicable to the product as a whole. As only the controller software was taken into account, these quality aspects needed to be translated for the controller software. E.g. the MCBF is higher for the controller software than for the product, because only failures caused by the controller software should be taken into account. Paper jams are not important when evaluating the maturity of the software. In total 16 metrics were defined, of which only 1 was defined additional to the internal and external metrics provided in the ISO technical reports. The

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

availability of design documentation was considered to be a good indication of the maintainability. Therefore a metric was added to verify the amount of design documentation that was available versus the amount of design documentation planned. Examples of the metrics defined can be found in table 2.

Quality characteristic	Sub-characteristic	Metric	Purpose
Functionality	Suitability	*Functional implementation completeness:* Number of missing functions detected during system testing / Number of functions described in requirement specifications.	How many functions have been implemented in relation to the number of functions specified in the requirement specifications?
Reliability	Maturity	*Mean Copies Between Failures:* Total number of copies during system testing / Number of defects, caused by controller software, detected during operation time.	How frequent are the defects of the controller software in operation?
Maintainability	Analysability	*Availability of design documentation:* Available (and approved) design documentation (i.e. SW architecture, top-level design, analysis views, design views and interface specifications) / Identified design documentation.	What's the proportion of design documentation available?

Table 2 Examples of metrics defined

Measurement

It was decided that a hypothetical baseline had to be defined before starting the actual measurement. Our goal was not to measure relative improvement with each test-cycle, but to compare the measured quality against an absolute goal, which was to be reached before the product could be released. Defining a baseline in advance forces people to start discussion when quality targets are not met. If no baseline is defined one is tempted to accept the quality as is, because no reference exists. The product is considered to be 'good enough'.

This baseline was defined by asking a number of experienced people within the project and comparable software projects for an estimate on each metric. These estimates were then averaged. Each estimate was requested to be the 'minimum but sufficient level' for release. E.g., the metric Mean Copies Between Failures was defined to be 100000. This holds that only 1 failure, attributable to the controller software, may occur every 100000 copies.

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

Decided was to measure all metrics (when applicable) during the system test phase of each incremental development cycle. The result of each development cycle could thus be scored against the defined baseline. In this way, both the improvements per development cycle, as well as the discrepancy with the 'minimum but sufficient level' could be depicted and reacted upon.

Collection of the metric data during the system test phases implied that some thorough administration was necessary during the test execution. Besides the defects found it was important to administer e.g. the number of copies made, the total down time of the copier/printer and the number of failures successfully restored by the system itself. It was important to clearly instruct the test engineers on what data should be recorded, otherwise important information to compute the metrics could be missing.

After each system test phase the measured values needed to be evaluated to see whether the baseline values defined at the start were still correct. Where necessary the baseline values could be modified, but then at least the discussion took place on the desired quality level. Project management should always have approved changing a baseline.

Real life experience

Until the moment of writing this paper, 3 development cycles have been tested and scored. For one increment, an extra system test cycle was added, leading to a fourth measurement (2 patch). Some results of this increment are presented in the table below.

Quality-characteristic	Sub-characteristic	Metric	Base-line	Value C1	Value C2	Value C2 patch	Value C3
Functionality	Suitability	Functional implementation completeness	0.90	0.91	1.0	1.0	0.82
		Functional implementation correctness	0.80	0.45	0.80	0.84	0.61
Reliability	Maturity	Defect detection	0.75	0.46	0.63	0.63	1.08
		Mean copies between failures	100000	n.a.	93	175	4880
		Test completeness	0.90	0.75	0.78	0.33	0.92
	Recoverability	Mean down time	10 min.	n.a.	5 min.	5 min.	5 min.

Table 3 Measurements results

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

From the table can be seen that not all metrics were measured for all increments. This is caused by the incremental development methodology used within the project, where it appeared to be impossible to measure e.g. 'Restorability' on release 1, since this functionality was not yet present.

The mean copies between failures is far below the baseline defined. Still this baseline is not adjusted. The low number for MCBF is caused mainly by the fact that the controller software was not yet robust for failures in the scanner or printer. E.g. a paper jam also resulted in a failure in the controller software. Release 3 was robust for paper jam and one can see that the MCBF increased tremendously.

Interesting to note are the measurements on release 2 patch compared with release 2. The actual use of the software resulting from release 2 was hindered by major instability and performance problems. These problems were resolved, after which measurement 2 patch took place. Although the difference between the metrics of release 2 and release 2 patch is only relatively small (only 9 major defects were solved, approx. 5% of the total amount of defects solved), the users perceived release 2 patch as a much 'better' system. This example shows that the metrics and values indicated in the previous table should be carefully interpreted. When the system only shows a few critical defects in the most used part of the system this system will be of an unacceptable quality level, but the metrics show otherwise. The severity of a defect and its location in the system is not taken into account. So besides the metrics also an evaluation based on common sense has to be made.

For release C3 one can see that the defect detection rate currently is higher than 1, which means that more defects have been found than initially expected. The calculation for the defects expected to be found might be incorrect, which has to be investigated. Literature and metrics of the own organisation were used to derive the number of defects expected to be found.

Conclusion and final remarks

We have now been measuring software quality metrics for less than a year. With more future releases planned (including more extended test periods), our metrics will improve, to the point where we can more clearly use them to set development priorities.

The metrics currently only are used to see what the quality level of the controller software is. The next step will be to submit changes in the development process to improve the quality of the software itself and the process. E.g. if the defect detection rate is lower than expected, as can be

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

seen for releases C1 to C2 patch, but the test completeness is quite high (75% - 78%) it might be necessary to evaluate the effectiveness of the test process. It's of course also possible that the engineering team makes fewer errors as expected.

Furthermore it's still unknown what will happen if the metrics show an insufficient quality level, but the release date has come. How will the organisation react on this and how much importance will it attach to the metrics? This is what still has to be found out.

Up till now the reporting on quality has been received well within the project, both for people involved in the quality model development and those only receiving the test results. The statements on product quality are now based on more than the number of defects and the feeling one has. Besides in the controller software development this method will now also be lifted to project level in order to make statements on the quality level of the copier/printer product as a whole.

As test and quality engineers we're very positive about the ISO9126 approach for defining quality and the questionnaire based method. It gave us a way of defining and reporting product quality in a clear and, quite, unambiguous manner. We've learned a lot and will continue the measurements for the remaining of the project and intend to also use it in future projects.

Authors information

Rob Hendriks
Improve Quality Services B.V.
Waalreseweg 17
5554 HA Valkenswaard
The Netherlands
E-mail: rhe@improveqs.nl

Robert van Vonderen
Océ Technologies B.V.
P.O. Box 101
5900 MA Venlo
The Netherlands
E-mail: lvvo@oce.nl

References

ISO/IEC 9126-1:2000, Information technology – Software product quality – Part 1: Quality model, International Organization for Standardization.

ISO/IEC 9126-2:1999, Information technology – Software product quality – Part 2: External metrics, International Organization for Standardization.

ISO/IEC 9126-3:2000, Information technology – Software product quality – Part 3: Internal metrics, International Organization for Standardization.

Solingen R. van & E. Berghout (1999), The goal/question/metric method, a practical method for quality improvement of software development, McGraw-Hill, UK, ISBN 007-709553-7.

Trienekens J.J.M. and E.P.W.M. van Veenendaal (1997), Software Quality from a business perspective, Kluwer Bedrijfsinformatie, Deventer, The Netherlands, ISBN 90-267-2631-7.

Veenendaal, E.P.W.M. van & J. McMullan (eds.) (1997), Achieving Software Product Quality, UTN Publishers, 's-Hertogenbosch, The Netherlands, ISBN 90-72194-52-7.

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

Appendix A Example product quality characteristics questionnaire

This appendix contains some of the questions of the product quality characteristics questionnaire. For each question the related quality sub-characteristics are indicated.

Process related questions:

What kind of market type is the product oriented towards?
1. Business market
2. Consumer market

(Related to suitability, interoperability, learnability, resource utilisation and time behaviour)

What is the geografic market target?
1. Local
2. Global

(Related to suitability, interoperability, learnability, resource utilisation and time behaviour)

What is the number of products to be sold in a certain market area?
1. 1-1000
2. 1000-10000
3. More than 10000

(Related to suitability and maturity)

User related questions:

What is the average experience of the recognized user groups with regard to the product?
1. More than one year of experience
2. Less than one year of experience
3. No experience

(Related to understandability)

What is the average age of the users?
1. Under 25
2. 25 – 40
3. Older than 40

(Related to learnability and attractiveness)

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl

Software product related questions:

Are there any alternatives to carry on with the activities when the software fails?
1. Yes
2. No

(Related to reliability)

Does the product perform actions without the user intervention?
1. Yes
2. No

(Related to understandability, learnability and operability)

Improve Quality Services BV, Waalreseweg 17, 5554 HA Valkenswaard, The Netherlands
Tel 040 – 2089283 • Fax 040 – 2021450 • E-mail info@improveqs.nl