

SIG/TÜViT Evaluation Criteria

Trusted Product Maintainability

with guidance for producers

Version 3.0

January 1, 2011
Dr. ir. Joost Visser
j.visser@sig.nl



Software Improvement Group

P.O. Box 94914
1090 GX Amsterdam
The Netherlands
t +31 20 314 0950
f +31 20 314 0955
info@sig.eu
www.sig.eu



1 Summary

Software product quality is a major concern for those that develop, maintain, enhance, acquire, or use software products or software-intensive systems. Technical quality of software products pertains to the ease and speed by which the software allows itself to be modified to keep pace with the changing needs of its users or other stakeholders.

This document specifies the *SIG/TÜViT Evaluation Criteria* for the quality mark:

TÜViT Trusted Product Maintainability



This quality mark certifies the technical quality of the source code of software products, where technical quality means the internal quality characteristic of *maintainability* and its sub-characteristics according to the ISO/IEC 9126 international standard for software product quality [ISO 9126-1]. These sub-characteristics are *analysability*, *changeability*, *stability*, and *testability*.

The assessment of the quality characteristic of maintainability and its sub-characteristics is based on the assessment of software product properties by source code analysis. These product properties are *volume*, *duplication*, *unit complexity*, *unit length*, *unit interfacing*, and *module coupling*. The numerical thresholds used in aggregation and rating are calibrated against a large benchmark repository of software product measurements.



2 Table of contents

1	SUMMARY	3
2	TABLE OF CONTENTS	4
3	INTRODUCTION	5
4	EVALUATION SCOPE	6
5	EVALUATION AREAS	7
5.1	Quality characteristics	7
5.2	Software product properties	7
5.3	Software product description	8
6	EVALUATION RESULTS	10
7	CALIBRATION OF THE EVALUATION METHOD	12
8	ISSUANCE OF CERTIFICATE AND CORRESPONDING QUALITY MARK	13
9	REFERENCES	14
A.	SUMMARY OF THE EVALUATION PROCEDURE	15
B.	CONTACT INFORMATION	16
C.	GUIDANCE FOR PRODUCERS	17
C.1.1	Volume	17
C.1.2	Duplication	17
C.1.3	Unit size	18
C.1.4	Unit complexity	18
C.1.5	Unit interfacing	18
C.1.6	Module coupling	18

3 Introduction

Software product quality is a major concern for those that develop, maintain, enhance, acquire, or use software products or software-intensive systems. The general notion of software quality embraces a variety of quality aspects of which a taxonomy is available in the ISO/IEC 9126 international standard on software product quality [ISO 9126-1].

Some quality aspects pertain to the current functions of the software and are primarily of interest to the users of software products. Other quality aspects pertain to the ease and speed by which the software allows itself to be modified to keep pace with the changing needs of its users. The latter aspects concern the *technical* rather than the *functional* quality of software products.

The *SIG/TÜViT Evaluation Criteria* for the quality mark *TÜViT Trusted Product Maintainability* are intended for the standardized evaluation and certification of the technical quality of the source code of software products. The purpose of such evaluation and certification is to provide an instrument to developers for guiding improvement of the products they create and enhance, and to acquirers for comparing, selecting, and accepting pre-developed software.

The scope of these *Evaluation Criteria* is limited to the internal quality characteristic of *maintainability* and its sub-characteristics of *analyzability*, *changeability*, *stability*, and *testability*. The evaluation concerns the source code of a software product, not the *behaviour* of the product in a test or production environment. The criteria can be used in combination with criteria for other notions of software quality, such as the quality of software processes, the quality of software professionals, or the quality of software installations.

The *Evaluation Criteria* define 5 increasing quality levels for maintainability represented by one to five stars, as outlined in Chapter 8. A certificate together with the quality mark *TÜViT Trusted Product Maintainability* can be issued for software products having successfully passed the evaluation based on these criteria and reaching an overall quality level of at least three stars.

4 Evaluation scope

The *SIG/TÜViT Evaluation Criteria Trusted Product Maintainability* are intended for the standardized evaluation and certification of the technical quality of the source code of software products. In general, the following kinds of software artefacts are considered for evaluation:

- Source code files (programs, scripts)
- Generated code files undergoing manual modification after generation
- Data definition files (data schemas)
- Data manipulation files (data queries)
- The active content of web pages

In addition to the software artefacts, a description of the product and its top-level components (their number, their names, their mapping onto software artefacts, and a description of their role in the product) shall be provided for evaluation.

Examples of artefacts that do not qualify as source code and are not considered for evaluation are the following:

- Data files
- Generated code files, other than those manually modified after generation
- Binary files (e.g. executables, object code, images)
- Configuration files (e.g. metadata files, property files)
- Web pages without active content (e.g. HTML files)
- End-user documentation (e.g. manual pages)
- Administrator or operator documentation (e.g. installation instructions)

NOTE: The scope of these evaluation criteria does not include any kind of software quality other than product quality. For example, quality of software processes (e.g. development process, maintenance process, testing process, acquisition process) or quality of software professionals (e.g. knowledge of staff about software technologies and or their practical skills) are not included in the evaluation scope.

5 Evaluation areas

The *SIG/TÜViT Evaluation Criteria Trusted Product Maintainability* follow the concept of the ISO/IEC 9126 international standard for software product quality [ISO 9126-1]. This standard elaborates the notion of software product quality into characteristics and sub-characteristics, and provides recommendations for association of metrics to these quality (sub-)characteristics.

5.1 Quality characteristics

The *SIG/TÜViT Evaluation Criteria* are targeted at the *technical quality* of the source code of software products, where technical quality means the *internal* quality characteristic of *maintainability* and its sub-characteristics according to the ISO/IEC 9126 international standard [ISO 9126-1]. These sub-characteristics are *analysability*, *changeability*, *stability*, and *testability*.

Maintainability: “The capability of the software product to be modified”, where “modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications” [ISO 9126-1, §6.5].

Analysability: “The capability of the software product to be diagnosed for deficiencies or causes of failures in the software, or for the parts to be modified to be identified” [ISO 9126-1, §6.5.1].

Changeability: “The capability of the software product to enable a specified modification to be implemented”, where “implementation includes coding, designing and documenting changes” [ISO 9126-1, §6.5.2].

Stability: “The capability of the software product to avoid unexpected effects from modifications of the software” [ISO 9126-1, §6.5.3].

Testability: “The capability of the software product to enable modified software to be validated” [ISO 9126-1, §6.5.4].

In terms of the quality perspectives distinguished by ISO/IEC 9126, the evaluation criteria include *internal* quality characteristics (i.e. concerning intrinsic properties), but exclude *external* quality characteristics (i.e. concerning behaviour in a test environment) or *quality-in-use* characteristics (i.e. concerning behaviour in a production environment). The evaluation concerns technical quality of the source code of a software product independent of its context of use or its role inside a system. In particular, the evaluation does not concern fulfilment of functional requirements or suitability for specified tasks.

5.2 Software product properties

The quality characteristics of maintainability and its sub-characteristics are determined by measuring a set of software product properties. These properties are *volume*, *duplication*, *unit complexity*, *unit size*, *unit interfacing*, and *module coupling*.

Volume: The overall size of the source code of the software product. Size is determined from the number of lines of code per programming language normalized with industry-average productivity factors for each programming language. Volume shall be rated on a scale that is independent of the type of the software product.

Duplication: The degree of duplication in the source code of the software product. Duplication concerns occurrence of identical fragments of source code in more than one place in the product.

Unit complexity: The degree of complexity in the units of the source code. The notion of *unit* corresponds to the smallest executable parts of source code, such as methods or functions.

Unit size: The size of the source code units in terms of the number of their source code lines.

Unit interfacing: The size of the interfaces of the units of the source code in terms of the number of interface parameter declarations.

Module coupling: The coupling between modules in terms of the number of incoming dependencies for the modules of the source code. The notion of *module* corresponds to a grouping of related units.

These software product properties have been chosen to be largely mutually independent. In particular, the total volume of a software product does not (statistically) predetermine the values for the other properties. The unit complexity and unit size of software products are known to be correlated, but only in a limited degree.

The determination of these software product properties is based on objective, repeatable, and accurate measurements at the level of source code lines, units, and modules. When the software product under evaluation consists of source code written in more than a single programming language, these measurements are conducted per programming language, after which the results per programming language are combined into whole-product results. The combination method shall take into account the relative volume of artefacts written in each programming language.

5.3 Software product description

In addition to the determination of the above product properties, a description of the product and its top-level components shall exist. This description shall satisfy the following minimal properties:

- The description identifies the product boundaries and its overall function.
- The description identifies all top-level components of the product.
- The description of the top-level components is such that any software artefact within the evaluation scope belongs to exactly one top-level component.



- The description identifies the role of each top-level component in the product.
- The description contains top-level components of appropriate number and size to facilitate maintenance of the product.

The high-level description of the software product, its top-level components, their functionality, interfaces, and interrelationships shall provide a global overview of the software product architecture and shall be appropriate for the understanding of the overall product to facilitate its maintenance.

6 Evaluation results

The evaluation leads to an assignment of evaluation results as a number of stars between one and five, where more stars represent better performance. The assignment is done for every product property, for every quality sub-characteristic, and for the main characteristic of maintainability.

The assignment of evaluation results to measurement areas proceeds in a number of steps. In the first step, the measurement results per programming language are used to assign overall evaluation results to the product properties. In the second step, the evaluation results for the product properties are used to assign evaluation results to the sub-characteristics of maintainability. In the final step, the evaluation results for the sub-characteristics are used to assign a single evaluation result to the technical quality or *maintainability* of the product as a whole.

The assignment of evaluation results at the level of quality sub-characteristics based on the evaluation results at the level of product properties takes the following relationships into account.

- The volume of source code negatively impacts its analyzability, since the diagnosis of faults or parts to be modified is more difficult or time-consuming when a larger volume of source code needs to be taken into consideration.
- The duplication of source code negatively impacts its analyzability, since the diagnosis of faults or parts to be modified is more difficult or time-consuming when the same or similar parts occur multiple times in different places of the source code.
- The duplication of source code negatively impacts its changeability, since the implementation of a modification is more difficult or time-consuming when it involves making changes to the source code in multiple locations.
- The unit size of source code negatively impacts its analyzability, since the diagnosis of faults or parts to be modified is more difficult or time-consuming when the source code is grouped into a low number of large units rather than into a high number of small units.
- The unit size of source code negatively impacts its testability, since validation of modified parts of source code is more difficult or time-consuming when the source code is grouped into a low number of large units rather than into a high number of small units.
- The unit complexity of source code negatively impacts its changeability, since the implementation of a modification in a specific part of the source code is more difficult or time-consuming when this part is complex.
- The unit complexity of source code negatively impacts its testability, since the validation of modifications to a software unit requires more test cases when the unit is complex.
- The unit interfacing of source code impacts its stability, since it is more difficult to avoid unexpected effects of modifications when the interfaces of units are large.



- The module coupling of source code impacts its changeability, since the modification of modules with many incoming dependencies is more difficult or time consuming.
- The module coupling of source code impacts its stability, since it is more difficult to avoid unexpected effects of modification for modules with many incoming dependencies.

These relationships between the product properties and quality sub-characteristics reflect expert opinion about the most significant influences of the various product properties on the quality sub-characteristics. They are summarized in the following table:

Maintainability sub-characteristics	Product property					
	Volume	Duplication	Unit size	Unit complexity	Unit interfacing	Module coupling
Analyzability	×	×	×			
Changeability		×		×		×
Stability					×	×
Testability			×	×		

A cross (X) indicates that a product property contributes to the maintainability sub-characteristic. All four sub-characteristics contribute to the main quality characteristics of *maintainability*. The weights of the different contributions are discussed in the following chapter.

7 Calibration of the evaluation method

Software quality measurements are meaningful when the measurement results reflect the relative position of a product within a large set of comparable products that have been evaluated recently. Thus, the numerical thresholds and weighting factors used in the evaluation shall be calibrated against a benchmark repository of source code analysis results of a large number of software products. The benchmark repository shall satisfy the following minimum requirements.

- The benchmark repository shall include products developed according to a range of different development methods, using a range of different programming languages, to the benefit of companies from a range of different industrial categories.
- The software products used for calibration shall include a substantial number of modern products, developed with modern programming languages.
- The total volume of the software products shall exceed 10 million lines of code.
- The total number of modules and units in the software products used for calibration shall exceed 1 million units and 100 thousand modules, respectively. Small, medium-sized, and large products shall all be represented.
- The information in the benchmark shall be curated by experts in software engineering quality and statistics.

At the level of product properties, calibration is carried out according to the following guidelines. The relative number of products in the repository to which a given number of stars is assigned shall approximately follow the following distribution:

- 5 stars: 5% of the products
- 4 stars: 30% of the products
- 3 stars: 30% of the products
- 2 stars: 30% of the products
- 1 star: 5% of the products

The best 5% of the products in the repository in terms of a given property (e.g. *unit complexity*) shall be assigned a 5 star rating; the next best 30% shall be assigned a 4 star rating; and so on. The last 5% of the products receive a single star.

At the level of quality sub-characteristics and at the level of maintainability, the rating assignments are done based on those at the level of product properties. The distribution of stars at those levels may be different. At the level of quality characteristics, the relative importance of product properties for each sub-characteristic and the relative importance of sub-characteristics for the final evaluation results for maintainability are established by knowledge elicitation from a panel of software quality experts. A structured method will be used for knowledge elicitation.



8 Issuance of certificate and corresponding quality mark

To be eligible for certification with the quality mark *TÜViT Trusted Product Maintainability*, the evaluation results for a software product shall satisfy a number of minimal conditions:

- The rating at the level of the main quality characteristics of *maintainability* is 3 stars or more.
- The rating at the level of each sub-characteristic of maintainability is 2 stars or more.
- The documentation requirements must be fulfilled.

When these three conditions are satisfied and documented in an evaluation report of a licensed evaluation facility, a certificate can be awarded to the software product with a mention of the rating of the product at the level of *maintainability*.

9 References

[ISO 14598-1] International Organization for Standardization. *ISO/IEC 14598-1: Information technology - software product evaluation - part 1: General overview*, 2001.

[ISO 9126-1] International Organization for Standardization. *ISO/IEC 9126-1: Software engineering - Product quality - Part 1: Quality model*, 2001.

[SIG 2007] Ilja Heitlager, Tobias Kuipers en Joost Visser, *A Practical Model for Measuring Maintainability*, In Proceedings of the 6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007), IEEE Computer Society Press, 2007.

[SIG 2008a] José Pedro Correia en Joost Visser, *Certification of Technical Quality of Software Products*, In Proceedings of the 2nd International Workshop on Foundations and Techniques for Open Source Software Certification (OpenCert 2008), satellite event of the 4th International Conference on Open Source Systems (OSS 2008).

[SIG 2008b] José Pedro Correia en Joost Visser, *Benchmarking Technical Quality of Software Products*, In Proceedings of the 15th Working Conference on Reverse Engineering (WCRE 2008), IEEE Computer Society, 2008.

[SIG 2011a] Joost Visser, *SIG Evaluation Guideline for Technical Quality of Software Products, Version 3.0*, Software Improvement Group, 2011.

A. Summary of the evaluation procedure

The procedure for evaluating a software product against the evaluation criteria of SIG/TÜViT Trusted Product Maintainability is documented in detail in the *SIG Evaluation Guideline for Technical Quality of Software Products* [SIG 2011a]. The procedure is structured according to the ISO/IEC 14598 International Standard for software product evaluation [ISO 14598-1].

The procedure involves the following steps:

- *Scoping*: In this step, the *scope* of the evaluation is established. This means that the source code files to be taken into consideration in the evaluation are identified. It may for instance be necessary to keep generated files out of scope. The scope is documented in a *scope definition*, which is presented to the evaluation client for confirmation.
- *Measuring*: In this step, the source code files within scope of the evaluation are measured. This results in a collection of measurement values at the level of source code lines, source code units, and source code modules.
- *Aggregation*: In this step, the measurements at the level of source code lines, units, and modules are aggregated to the level of properties of the product as a whole.
- *Rating*: In this step, the aggregated measurement values are used to assign ratings to the evaluation areas (product properties and quality sub-characteristics) listed in Chapter 5.

After these steps, the scope definition, measurement results, and ratings are recorded in an evaluation report.

B. Contact information

Software Improvement Group B.V.

Dr. Per John
Head of Evaluation Laboratory
Rembrandt Tower, 14th floor
Amstelplein 1
1096 HA Amsterdam
The Netherlands

T: +31 20 314 09 50
F: +31 20 314 09 55
E: p.john@sig.nl
W: <http://www.sig.eu>

TUV Informationstechnik GmbH

Member of TÜV NORD Group
Dr. Christoph Sutter
Certification
Division Manager
Langemarckstr. 20
45141 Essen
Germany

T: +49 201 8999-582
F: +49 201 8999-555
E: c.sutter@tuvit.de
W: <http://www.tuvit.net>

C. Guidance for producers

This appendix provides explanation to software producers about the measurement method of SIG applied for evaluation. For each measurement area, the threshold measurement values are provided that are required for eligibility of certification at the level of 4 stars. Note that these clarifications are meant as guidance for software producers, providing sufficient conditions for satisfying the measurement model that is used during evaluation by SIG.

C.1.1 Volume

Larger software products are deemed to be harder to maintain. To maximize the rating of a product with respect to volume, the producer should strive to keep its source code concise.

For the evaluation of the volume property, the software product's **rebuild value** is estimated on the basis of the number of lines of source code. To make the lines of code of software artefacts written in different programming languages comparable to each other, they are normalized on the basis of industry averages. To be eligible for certification at the level of 4 stars, the total rebuild value of the product should not exceed 30 man years. The following table lists the number of lines of code that is produced on average for some industry standard technologies.

Language	LoC in 30 man years
Java	263,000
C#	240 000
C	385,000

C.1.2 Duplication

Software products with less (textual) duplication are deemed to be easier to maintain. To maximize the rating of a product for the duplication property, the software producer should avoid multiple occurrences of the same fragments of code.

For the evaluation of the duplication property, an estimation is made of the **percentage of redundant lines of code**. A line of code is considered redundant if it is part of a code fragment (larger than 6 lines of code) that is repeated literally (modulo white-space) in at least one other location in the source code.

To be eligible for certification at the level of 4 stars, the percentage of redundant lines of code for each programming language used should not exceed 5%.

C.1.3 Unit size

Software products where more source code resides in large units are deemed to be harder to maintain. To maximize the rating of a product for the unit size property, the software producer should avoid large units.

The size of the units of a software product is determined by counting the number of lines of code within each unit.

To be eligible for certification at the level of 4 stars, for each programming language used the percentage of lines of code residing in units with more than 20 lines of code should not exceed 38%. The percentage in units with more than 50 lines of code should not exceed 11%. The percentage in units with more than 100 lines should not exceed 3%.

C.1.4 Unit complexity

Software products where more source code resides in units with high logical complexity are deemed to be harder to maintain. To maximize the rating of a product for the unit complexity property, the software producer should avoid units with high complexity.

The complexity of each unit is determined in terms of the McCabe cyclomatic complexity number. This number represents the number of non-cyclic paths through the control-flow graph of the unit and can be calculated by counting the number of decision points that are present in the source code.

To be eligible for certification at the level of 4 stars, for each programming language used the percentage of lines of code residing in units with McCabe complexity number higher than 10 should not exceed 11%. The percentage of lines of code in units with McCabe complexity number higher than 20 should not exceed 3%. There should be no code in units with McCabe complexity higher than 50.

C.1.5 Unit interfacing

Software products where more source code resides in units with large interfaces are deemed to be harder to maintain. To maximize the rating of a product for the unit interfacing property, the software producer should avoid units with large interfaces.

The size of the interface of a unit can be quantified as the number of parameters (also known as formal arguments) that are defined in the signature or declaration of a unit.

To be eligible for certification at the level of 4 stars, for each programming language used the percentage of lines of code residing in units with 3 or more parameters should not exceed 15%. The percentage in units with 5 or more parameters should not exceed 4%. The percentage in units with 7 or more parameters should not exceed 0.8%.

C.1.6 Module coupling

Software products where more source code resides in modules that are strongly coupled with other modules are deemed to be harder to maintain. To maximize the rating of a product for the module coupling property, the software producer should avoid having



strong coupling between modules. Modules are groupings of units, such as classes that group methods or files that group functions.

The coupling of the modules of a software product can be quantified as the number of incoming dependencies, such as invocations, per module.

To be eligible for certification at the level of 4 stars, for each programming language used the percentage of lines of code residing in modules with a number of incoming dependencies above 16 should not exceed 21%. The percentage in modules with a number of incoming dependencies above 32 should not exceed 14%. The percentage in modules with a number of incoming dependencies above 75 should not exceed 6