INTERNATIONAL STANDARD

ISO/IEC 25040

First edition
2011-03-01

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process

Ingénierie des systèmes et du logiciel — Exigences de qualité et évaluation des systèmes et du logiciel (SQuaRE) — Modèle de référence d'évaluation et guide

Reference number
ISO/IEC 25040:2011(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electro technical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 25040 is part of the SQuaRE series of standards and was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering.*

Introduction

As the use of information technology grows, the number of critical computer systems also grows. Such systems include, for example, security critical, life critical, economically critical and safety critical systems. The quality of software in these systems is particularly important because software faults can lead to serious consequences.

Evaluation is the systematic determination of the extent to which an entity meets its specified criteria. The evaluation of software product quality is vital to both the acquisition and development of software. The relative importance of the various characteristics of software quality depends on the intended usage or objectives of the system of which the software is a part; software products need to be evaluated to decide whether relevant quality characteristics meet the requirements of the system.

This document is part of the SQuaRE series of standards and contains general requirements for software product quality evaluation as well as clarifies the associated general concepts.

The general goal of creating the SQuaRE set of standards is to move to a logically organized, enriched and unified series covering two main processes: software quality requirements specification and software quality evaluation, supported by a software quality measurement process. The purpose of the SQuaRE set of standards is to assist those developing and acquiring software products with the specification and evaluation of quality requirements. It establishes criteria for the specification of software product quality requirements, their measurement, and evaluation. It includes a quality model for aligning customer definitions of quality with attributes of the development process. In addition, the series provides recommended measures of software product quality attributes that can be used by developers, acquirers, and evaluators.

SQuaRE provides

- terms and definitions,
- reference models,
- general guide,
- individual division guides, and
- standards for requirements specification, planning and management, measurement and evaluation purposes.

SQuaRE includes International Standards on quality model and measures, as well as on quality requirements and evaluation.

SQuaRE replaces the current ISO/IEC 9126 series and the ISO/IEC 14598 series.

This International Standard is intended to be used in conjunction with the other parts of the SQuaRE series of standards, and with the ISO/IEC 14598 series and the ISO/IEC 9126 series until superseded by the ISO/IEC 25000 series of standards.

The SQuaRE series of standards consists of the following divisions under the general title *Systems and software product Quality Requirements and Evaluation:*

- ISO/IEC 2500n - *Quality Management Division,*
- ISO/IEC 2501n - *Quality Model Division*,

- ISO/IEC 2502n - *Quality Measurement Division*,
- ISO/IEC 2503n - *Quality Requirements Division*, and
- ISO/IEC 2504n - *Quality Evaluation Division*.

Annex A provides an explanation on levels of evaluation, aspects to be considered when defining evaluation levels and suggestions on evaluation techniques to be applied according to the rank of evaluation level.

Annex B provides examples of evaluation methods.

Annex C provides a table showing relationships between some evaluation methods, possible cost rank and effectiveness per software quality characteristics.

Annex D provides relationships between the software product quality evaluation process reference model and the software and system life cycle processes.

Annex E provides an example template of an evaluation report.

Annex F provides the diagrams of inputs, outcomes, constraints and resources for each evaluation activity.

Figure 1 illustrates the organization of the SQuaRE series representing families of standards, further called Divisions.

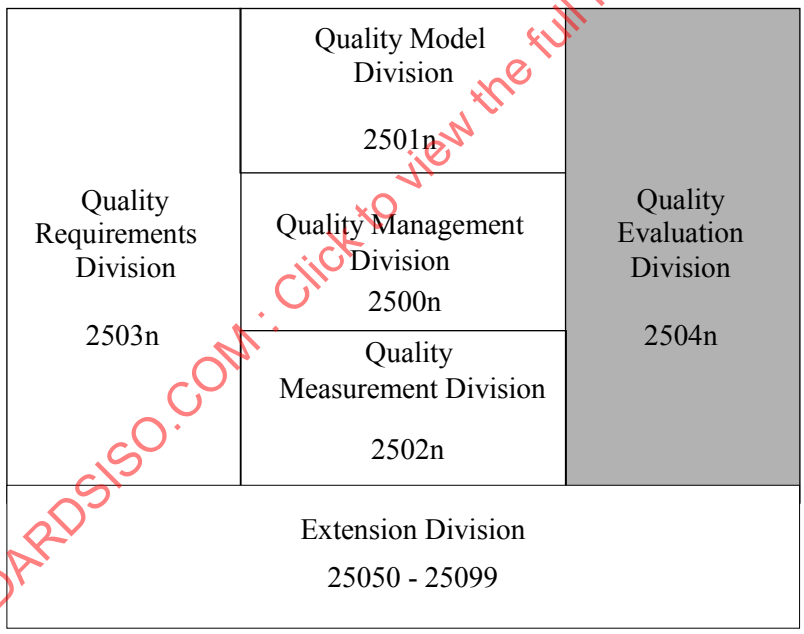

Figure 1 — Organization of the SQuaRE series of International Standards

The Divisions within the SQuaRE model are as follows.

- **ISO/IEC 2500n - Quality Management Division**. The International Standards that form this division define all common models, terms and definitions referred to by all other standards from the SQuaRE series. Referring paths (guidance through SQuaRE documents) and high-level practical suggestions in applying proper standards to specific application cases offer help to all types of users. The division also provides requirements and guidance for a supporting function which is responsible for the management of software product requirements, specification and evaluation.

- **ISO/IEC 2501n - Quality Model Division**. The International Standard that forms this division presents detailed quality models for software, quality in use and data. Practical guidance on the use of the quality model is also provided.

- **ISO/IEC 2502n - Quality Measurement Division**. The International Standards that form this division include a software product quality measurement reference model, mathematical definitions of quality measures, and practical guidance for their application. This division presents internal measures of software quality, external measures of software quality and quality in use measures. Quality measure elements (QME) forming foundations for the latter measures are defined and presented.

- **ISO/IEC 2503n - Quality Requirements Division**. The International Standard that forms this division helps specifying quality requirements. These quality requirements can be used in the process of quality requirements, elicitation for a software product to be developed or as inputs for an evaluation process. The requirements definition process is mapped to technical processes defined in ISO/IEC 15288.

- **ISO/IEC 2504n - Quality Evaluation Division**. The International Standards that form this division provide requirements, recommendations and guidelines for software product evaluation, whether performed by independent evaluators, acquirers or developers. The support for documenting a measure as an evaluation module is also presented.

ISO/IEC 25050 to ISO/IEC 25099 are reserved to be used for SQuaRE extension International Standards and/or Technical Reports.

This International Standard is part of the 2504n series on quality evaluation division that currently consists of the following International Standards:

- **ISO/IEC 25040 - Evaluation process**: contains general requirements for specification and evaluation of software quality and clarifies the general concepts. Provides a process description for evaluating quality of software product and states the requirements for the application of this process. The evaluation process is the basis for software product quality evaluation for different purposes and approaches. Therefore, the process can be used for the evaluation of quality in use, external measure of software quality and internal measure of software quality and can be applied to evaluate the quality of pre-developed software or custom software during its development process. The software product quality evaluation can be conducted, for instance, by an acquirer, a developer organization, or an independent evaluator.

- **ISO/IEC 25041 - Evaluation guides for developers, acquirers and evaluators**: contains specific requirements and recommendations for developers, acquirers and evaluators.

- **ISO/IEC 25042 - Evaluation modules**: defines the structure and content of the documentation to be used to describe an evaluation module. These evaluation modules contain the specification of the quality model (i.e. characteristics, subcharacteristics and corresponding internal, external or quality in use measures), the associated data and information about the planned application of the model and the information about its actual application. Appropriate evaluation modules are selected for each evaluation. In some cases it may be necessary to develop new evaluation modules. Guidance for developing new evaluation modules is found in ISO/IEC 25042. This International Standard can also be used by organizations producing new evaluation modules.

- **ISO/IEC 25045 - Evaluation module for recoverability**: provides the specification to evaluate the subcharacteristic of recoverability defined under the characteristic of reliability of the quality model. It determines the external measures of software quality of resiliency and autonomic recovery index when the information system composed of one or more software products' execution transactions is subjected to a series of disturbances. A disturbance could be an operational fault (e.g. an abrupt shutdown of an OS process that brings down a system) or an event (e.g. a significant increase of users to the system).

ISO/IEC 25040 is a revised version and replaces the current ISO/IEC 14598-1.

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process

1 Scope

This International Standard contains requirements and recommendations for the evaluation of software product quality and clarifies the general concepts. It provides a process description for evaluating software product quality and states the requirements for the application of this process. The evaluation process can be used for different purposes and approaches. The process can be used for the evaluation of the quality of pre-developed software, commercial-off-the-shelf software or custom software and can be used during or after the development process.

This International Standard establishes the relationship of the evaluation reference model to the SQuaRE documents as well as shows how each SQuaRE document should be used during the activities of the evaluation process.

It is intended for those responsible for software product evaluation and is appropriate for developers, acquirers and independent evaluators of software products. These three different approaches are detailed in ISO/IEC 14598-3, ISO/IEC 14598-4, and ISO/IEC 14598-5.

It is not intended for evaluation of other aspects of software products (such as functional requirements, process requirements, business requirements, etc.).

2 Conformance

Evaluation of software product quality conforms to this International Standard if it complies with the requirements of Clause 6.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

There are no normative references in this document.

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1
acquirer
individual or organization that acquires or procures a system, software product or software service from a supplier

NOTE Adapted from ISO/IEC 12207:2008.

4.2
analysis model
algorithm or calculation combining one or more base and/or derived measures with associated decision criteria

4.3
attribute
inherent property or characteristic of an entity that can be distinguished quantitatively or qualitatively by human or automated means

NOTE 1 Adapted from ISO/IEC 15939:2007.

NOTE 2 ISO 9000 distinguishes two types of attributes: a permanent characteristic existing inherently in something; and an assigned characteristic of a product, process or system (e.g. the price of a product, the owner of a product). The assigned characteristic is not an inherent quality characteristic of that product, process or system.

4.4
attribute for quality measure
attribute that relates to software product itself, to the use of the software product or to its development process

NOTE Attributes for quality measure are used in order to obtain quality measure elements.

4.5
base measure
measure defined in terms of an attribute and the method for quantifying it

NOTE 1 A base measure is functionally independent of other measures.

NOTE 2 Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

[ISO/IEC 15939:2007]

4.6
commercial-off-the-shelf software product
software product defined by a market-driven need, commercially available, and whose fitness for use has been demonstrated by a broad spectrum of commercial users

4.7
context of use
users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used

[ISO 9241-11:1998]

4.8
custom software
software product developed for a specific application from a user requirements specification

4.9
data
collection of values assigned to base measures, derived measures and/or indicators

[ISO/IEC 15939:2007]

4.10
decision criteria
thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result.

[ISO/IEC 15939:2007]

4.11
derived measure
measure that is defined as a function of two or more values of base measures

[ISO/IEC 15939:2007]

NOTE 1 Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

NOTE 2 A transformation of a base measure using a mathematical function can also be considered as a derived measure.

4.12
developer
individual or organization that performs development activities (including requirements analysis, design, testing through acceptance) during the software life-cycle process

NOTE Adapted from the definition in ISO/IEC 12207:2008.

4.13
division of standards
division forms a family of standards serving complementary purposes

4.14
end user
individual person who ultimately benefits from the outcomes of the system

NOTE The end user can be a regular operator of the software product or a casual user such as a member of the public.

4.15
entity
object that is to be characterized by measuring its attributes

EXAMPLE An object can be a process, product, project, or resource.

[ISO/IEC 15939:2007]

4.16
evaluation
systematic determination of the extent to which an entity meets its specified criteria

[ISO/IEC 12207:2008]

4.17
evaluation coverage
degree to which the evaluation covers the specified software product quality requirements

4.18
evaluation level
rigour to be applied during the evaluation that defines the depth or thoroughness of the evaluation in terms of evaluation techniques to be applied and evaluation results to be achieved

4.19
evaluation method
procedure describing actions to be performed by the evaluator in order to obtain results for the specified measurement applied to the specified product components or on the product as a whole

4.20
evaluation module
package of evaluation technology for measuring software quality characteristics, subcharacteristics or attributes

NOTE The package includes evaluation methods and techniques, input to be evaluated, data to be measured and collected and supporting procedures and tools.

4.21
evaluation records
documented objective evidence of all activities performed and of all results achieved within the evaluation process

4.22
evaluation requester
person or organization that requests an evaluation

4.23
evaluation tool
instrument that can be used during evaluation to collect data, to perform interpretation of data or to automate part of the evaluation

NOTE Examples of such tools are source code analysers to compute code metrics, CASE tools to produce formalized models, test environments to run the executable programs, checklists to collect inspection data or spreadsheets to produce syntheses of measures.

4.24
evaluation stringency
degree required for the software product quality characteristics and subcharacteristics to fulfil the expected use criticality of the software product

4.25
evaluator
individual or organization that performs an evaluation

4.26
failure
termination of the ability of a product to perform a required function or its inability to perform within previously specified limits

NOTE Adapted from IEEE 610.12-1990.

4.27
fault
incorrect step, process or data definition in a computer program

[IEEE 610.12-1990]

4.28
functional requirement
requirement that specifies a function that a system or system component must be able to perform

[IEEE 610.12-1990]

NOTE The software quality characteristic “functionality” can be used to specify or evaluate the suitability, accuracy, interoperability, security and compliance of a function.

4.29
implied needs
needs that may not have been stated but are actual needs

NOTE Some implied needs only become evident when the software product is used in particular conditions.

EXAMPLE Implied needs include: needs not stated but implied by other stated needs and needs not stated because they are considered to be evident or obvious.

4.30
independent evaluator
individual or organization that performs an evaluation independently from developers and acquirers

NOTE The individual or organization acting as developer or acquirer for the target system to be evaluated cannot become the independent evaluator for the system. The independent evaluator can be an organization. Independent evaluators can belong to the same organization as the developers as long as they are independent from developers and acquirers.

4.31
indicator
measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs

[ISO/IEC 15939:2007]

NOTE In ISO/IEC 14598-1 this definition was “a measure that can be used to estimate or predict another measure”.

4.32
information need
insight necessary to manage objectives, goals, risks, and problems

[ISO/IEC 15939:2007]

4.33
information product
one or more indicators and their associated interpretations that address an information need

EXAMPLE A comparison of a measured defect rate to planned defect rate along with an assessment of whether or not the difference indicates a problem.

[ISO/IEC 15939:2007]

4.34
information system needs
needs that can be specified as quality requirements by external measures and sometimes by internal measures

4.35
intermediate software product
product of the software development process that is used as input to another stage of the software development process

EXAMPLE Intermediate software products can include static and dynamic models, other documents and source code.

4.36
intermediate software product needs
needs that can be specified as quality requirements by internal measures

4.37
maintainer
individual or organization that performs maintenance activities

NOTE Adapted from ISO/IEC 12207:2008.

4.38
measure, noun
variable to which a value is assigned as the result of measurement

NOTE 1 The term “measures” is used to refer collectively to base measures, derived measures, and indicators.

NOTE 2 Adapted from ISO/IEC 14598-1:1999.

4.39
measure, verb
make a measurement

[ISO/IEC 14598-1:1999]

4.40
measurement
set of operations having the object of determining a value of a measure

[ISO/IEC 15939:2007]

NOTE 1 Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

NOTE 2 Measurement can include assigning a qualitative category such as the language of a source program (ADA, C, COBOL, etc.).

4.41
measurement function
algorithm or calculation performed to combine two or more base measures

[ISO/IEC 15939:2007]

4.42
measurement method
logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale

[ISO/IEC 15939:2007]

NOTE Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

4.43
measurement procedure
set of operations, described specifically, used in the performance of a particular measurement according to a given method

[ISO/IEC 15939:2007]

NOTE Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

4.44
measurement process
process for establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure

NOTE Adapted from ISO/IEC 15939:2007.

4.45
observation
instance of applying a measurement procedure to produce a value for a base measure

[ISO/IEC 15939:2007]

4.46
operator
individual or organization that operates the system

NOTE Adapted from ISO/IEC 12207:2008.

4.47
process
system of activities, which uses resources to transform inputs into outputs

NOTE Adapted from ISO 9000:2005.

4.48
quality in use (measure)
the extent to which a product used by specific users meets the users' needs to achieve specific goals with effectiveness, productivity, safety and satisfaction in specific contexts of use

4.49
quality measure elements
measure, which is either a base measure or a derived measure, that is used for constructing software quality measures

NOTE The software quality characteristic or subcharacteristic of the entity is derived afterwards by calculating a software quality measure.

4.50
quality model
defined set of characteristics, and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality

4.51
rating
action of mapping the measured value to the appropriate rating level and used to determine the rating level associated with the software product for a specific quality characteristic

4.52
rating level
scale point on an ordinal scale, which is used to categorize a measurement scale

NOTE 1 The rating level enables the software product to be classified (rated) in accordance with the stated or implied needs.

NOTE 2 Appropriate rating levels may be associated with the different views of quality, i.e. users', managers' or developers'.

4.53
requirements
expression of a perceived need that something be accomplished or realized

NOTE The requirements can be specified as part of a contract, or specified by the development organization, as when a product is developed for unspecified users, such as consumer software, or the requirements can be more general, as when a user evaluates products for comparison and selection purpose.

4.54
scale
ordered set of values, continuous or discrete, or a set of categories to which the attribute is mapped

[ISO/IEC 15939:2007]

NOTE Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

EXAMPLE Types of scales are: a nominal scale which corresponds to a set of categories; an ordinal scale which corresponds to an ordered set of scale points; an interval scale which corresponds to an ordered scale with equidistant scale points; and a ratio scale which not only has equidistant scale points, but also possesses an absolute zero. Measures using nominal or ordinal scales produce qualitative data, and measures using interval and ratio scales produce quantitative data.

4.55
software product
set of computer programs, procedures, and possibly associated documentation and data

[ISO/IEC 12207:2008]

NOTE 1 Products include intermediate products, and products intended for users such as developers and maintainers.

NOTE 2 In SQuaRE standards, software quality has the same meaning as software product quality.

4.56
software product evaluation
technical operation that consists of producing an assessment of one or more characteristics of a software product according to a specified procedure

4.57
software quality
degree to which the software product satisfies stated and implied needs when used under specified conditions

NOTE 1 This definition differs from the ISO 9000:2005 quality definition because the software quality definition refers to the satisfaction of stated and implied needs, while the ISO 9000 quality definition refers to the satisfaction of requirements.

NOTE 2 Adapted from the ISO/IEC 25000:2005 definition, rephrased as “degree to which”.

4.58
software quality characteristic
category of software quality attributes that bears on software quality

NOTE Software quality characteristics can be refined into multiple levels of subcharacteristics and finally into software quality attributes.

4.59
software quality evaluation
systematic examination of the extent to which a software product is capable of satisfying stated and implied needs

4.60
software quality in use
capability of the software product to enable specific users to achieve specific goals with effectiveness, productivity, safety and satisfaction in specific contexts of use

NOTE Before the product is released, quality in use can be specified and measured in a test environment for the intended users, goals and contexts of use. Once in use, it can be measured for actual users, goals and contexts of use. The actual needs of users might not be the same as those anticipated in requirements, so actual quality in use might be different from quality in use measured earlier in a test environment.

4.61
software quality measure
internal measure of software quality, external measure of software quality or software quality in use measure

NOTE Internal measure of software quality, external measure of software quality or software quality in use measure are described in the quality model in ISO/IEC 25010.

4.62
stakeholder
a party having a right, share or claim in a system or in its possession of characteristics that meet that party's needs and expectations

NOTE 1 Adapted from ISO/IEC 15288:2008.

NOTE 2 Stakeholders include, but are not limited to, end users, end user organizations, supporters, developers, producers, trainers, maintainers, disposers, acquirers, supplier organizations and regulatory bodies.

4.63
supplier
individual or organization that enters into a contract with the acquirer for the supply of a system, software product or software service under the terms of the contract

NOTE Adapted from ISO/IEC 12207:2008.

4.64
system
a combination of interacting elements organized to achieve one or more stated purposes

NOTE 1 A system can be considered as a product or as the services it provides.

NOTE 2 In practice, the interpretation of its meaning is frequently clarified by the use of an associative noun, e.g. aircraft system. Alternatively, the word system can be substituted simply by a context dependent synonym, e.g. aircraft, though this can then obscure a system principles' perspective.

[ISO/IEC 15288:2008]

4.65
target of process
software product or task executed by software product to which a measurement or evaluation process is applied

4.66
unit of measurement
particular quantity, defined and adopted by convention, with which a other quantities of the same kind are compared in order to express their magnitude relative to that quantity

[ISO/IEC 15939:2007]

NOTE Adapted from the *International Vocabulary of Basic and General Terms in Metrology*, 1993.

4.67
user
individual or group that interacts with a system or benefits from a system during its utilization

NOTE Users can include operators, recipients of the results of the software, or developers or maintainers of software.

4.68
validation
confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled

NOTE 1 “Validated” is used to designate the corresponding status.

[ISO 9000:2005]

NOTE 2 In design and development, validation concerns the process of examining a product to determine conformity with user needs.

NOTE 3 Validation is normally performed on the final product under defined operating conditions. It can be necessary in earlier stages.

NOTE 4 Multiple validations can be carried out if there are different intended uses.

4.69
value
number or category assigned to an attribute of an entity by making a measurement

4.70
verification
confirmation, through the provision of objective evidence, that specified requirements have been fulfilled

NOTE 1 “Verified” is used to designate the corresponding status.

[ISO 9000:2005]

NOTE 2 In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity.

5 Software product quality evaluation reference model

5.1 Reference model - general

Software product quality evaluation process includes the inputs and outcomes, constraints and resources for the software product quality evaluation process, as shown in Figure 2.

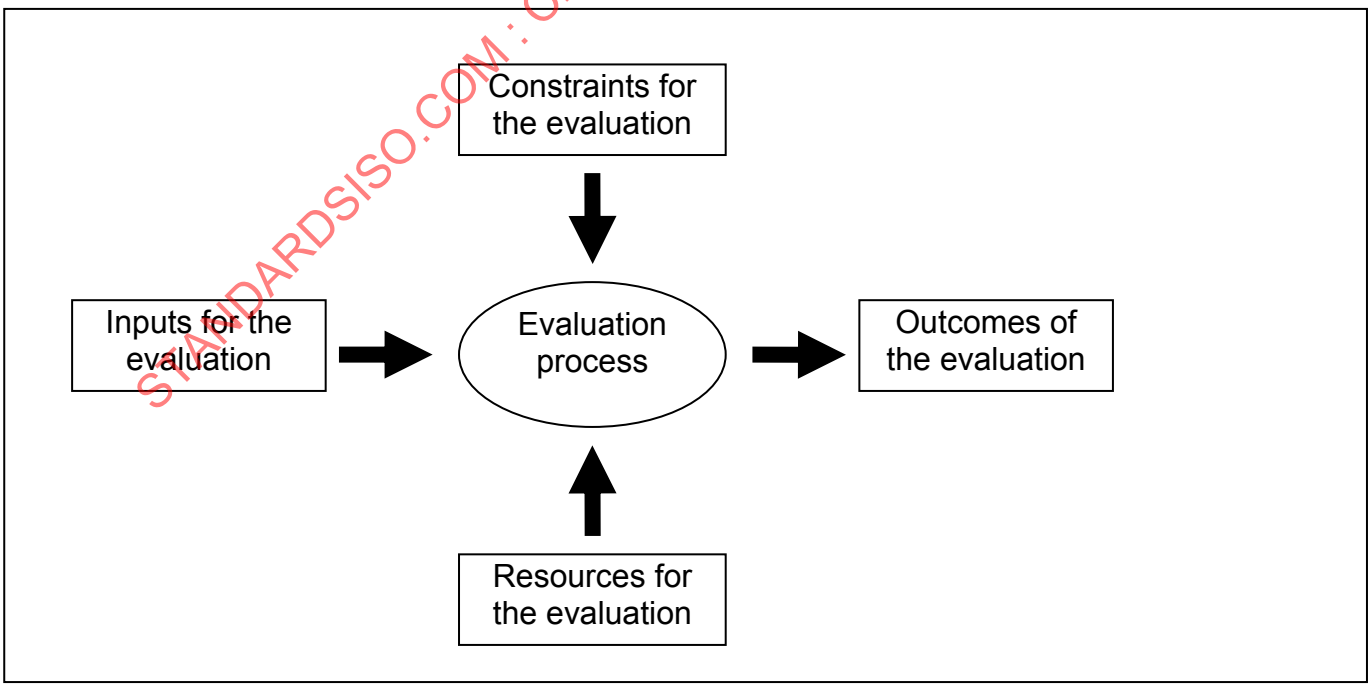

Figure 2 — Overview of software product quality evaluation

Inputs for the evaluation and outcomes for the evaluations are identified in clause 6.

Constraints for the software product quality evaluation process can include the following:

a) Specific user needs;

b) Resources;

c) Schedule;

d) Cost;

e) Environment;

f) Tools and methodology;

g) Reporting.

Resources for the software product quality evaluation process can include the following:

a) Applicable measurement tools and methodology, including evaluation modules;

b) Applicable SQuaRE documents (ISO/IEC 25001, 25010, 2502n, 25030, 25041, 25042);

c) Human resources for software product quality evaluation;

d) Economical resource for software product quality evaluation;

e) Information system for software product quality evaluation;

f) Knowledge data base for software product quality evaluation.

The software product quality evaluation reference model applies to those responsible for software product evaluation. It is appropriate for organizations in their role as acquirers, developers, or evaluators. It is intended but not limited to, developers, acquirers and independent evaluators of software products.

The software product quality evaluation reference model intends that the evaluation should be based on a software product quality requirement specification by using ISO/IEC 25030 before the evaluation and making clear the objectives and criteria of evaluations. ISO/IEC 25030 provides requirements and recommendations for software product quality requirements specification. It applies other SQuaRE document such as ISO/IEC 25010 and ISO/IEC 2502n.

Software product quality evaluation can be performed during or after the development process or acquisition process by the developer organization, the acquirer organization or an independent evaluator.

5.2 Reference model - evaluation processes

The software product quality evaluation process reference model describes the process and details the activities and tasks providing their purposes and complementary information that can be used to guide a software product quality evaluation (Figure 3).

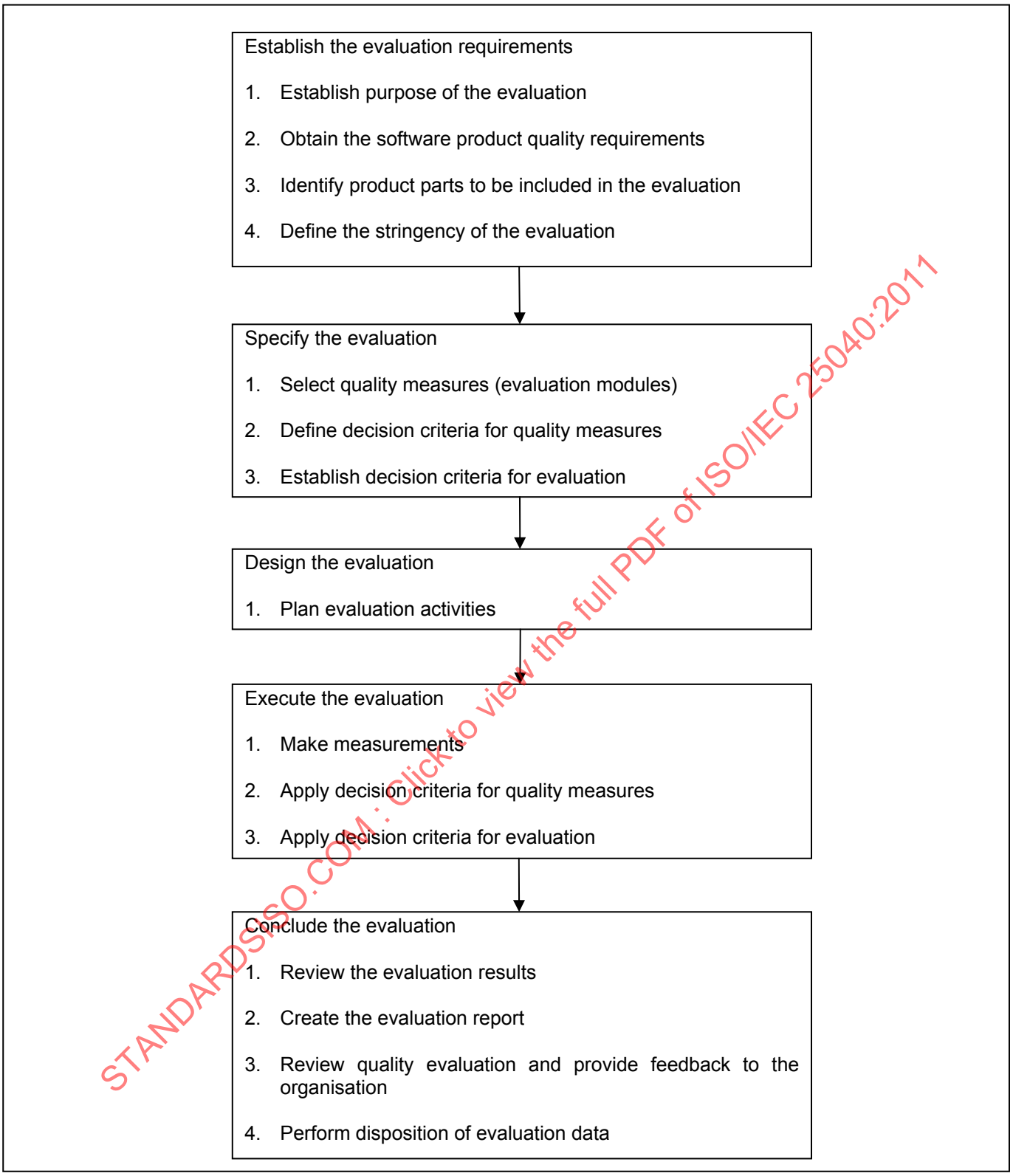

Figure 3 — Software Product Quality Evaluation Process

Clause 6 details the activities and tasks providing their inputs, outcomes and complementary information that can be used to guide a software product quality evaluation. Notice that clause 6 intends to present the general evaluation process.

It is essential that the implementation of the evaluation process have the flexibility to accommodate the uniqueness of each application, to avoid unnecessary work or work that adds no value, and to provide a practical means of establishing the requisite confidence in the software.

Requirements and recommendations for each activity and task are provided in clause 6.

5.3 Roles

Different roles including acquirers, developers, independent evaluator, suppliers, operators, and maintainers have different purposes.

- The acquirer: When acquiring a custom made software product, the acquirer can establish quality in use requirements and software quality requirements, can specify the requirements to the supplier, and can evaluate potential purchases against these requirements before acquisition. When acquiring a product to be developed, the objective of specifying quality requirements is to ensure that the product meets the stated and implied needs of the user. When purchasing a software product, evaluation can be used to compare alternative products and to ensure that the selected product meets the quality requirements.

- The developer: When implementing a custom made software product, the developer can evaluate the intermediate software products or final product in order to ensure the developed software quality. The developer can use the results of software product quality evaluation to ensure that products meet required quality criteria, which can be set by the acquirer, or by comparison with other products.

- The independent evaluators: When evaluating a target software product, the independent evaluator can evaluate the intermediate software products or final product in order to ensure the software quality. The evaluation process for independent evaluation approach provides requirements and recommendations for the practical implementation of software product evaluation, when several parties need to understand, accept and trust evaluation results. It is used by evaluators carrying out an independent evaluation of a software product. This evaluation could be performed at the request of a developer, acquirer or some other party.

- The supplier: The supplier can use the results of software product evaluation to ensure that products meet required quality criteria, which can be set by the acquirer, or by comparison with other products.

- The operator: The individual or organisation, which operates a system of which the software product is a part, can use software quality evaluation to validate that quality requirements are met under variable operating conditions, and to provide feedback on the need for any changes to those responsible for maintenance.

- The maintainer: The individual or organisation, which maintains a system of which the software product is a part, can use software evaluation to validate that quality requirements are still met, and requirements for maintainability and portability are achieved.

5.4 Quality in the life cycle

The software quality lifecycle and how software measures relate to the lifecycle is described in ISO/IEC 25020.

5.5 Support for the evaluation

These are activities for assisting evaluation by collecting information on software product quality evaluation methods and tools, developing and validating measures, and standardising evaluation process, and measures.

ISO/IEC 25001 contains requirements and guidance for supporting processes for software product quality evaluation as well as for improving the organizational level evaluation process by means of, for instance, assessing each evaluation project and the own evaluation process.

ISO/IEC 25041[1)] contains specific requirements and recommendations for developers, acquirers and evaluators.

ISO/IEC 25042 defines the structure and contents of evaluation modules. Evaluation modules document both evaluation technology and procedures for applying the technology.

6 Software product quality evaluation process

The purpose of this process is to conduct software product quality evaluation.

6.1 General requirements

The evaluator shall implement the activities and tasks described in clause 6 in accordance with applicable organization policies and procedures with respect to the software product quality evaluation process (see Figure 3).

NOTE 1 Figure 3 is a simplified view and does not show all iterations relating to specific activities or between activities.

There shall be an infrastructure including tools and technology in the evaluator's organization suitable for carrying out software product quality evaluation.

NOTE 2 ISO/IEC 25001 provides guidance on planning and management organizational aspects related to software product quality evaluation.

There shall be an infrastructure that allows for data collection and process modifications based on data analysis.

The personnel involved in the evaluation shall have the necessary skills and training.

In order to ensure repeatability, reproducibility, impartiality and objectivity of the evaluation results, the evaluator shall act in an organisational context that provides all necessary assurance to obtain sufficient quality for its activities.

6.2 Documentation

Each activity in the software product quality evaluation shall be recorded.

The records shall include a detailed account of actions performed by the evaluator while executing the software product quality evaluation plan.

The records shall contain sufficient information required for the management of the software product quality evaluation.

The evaluation records shall include any intermediate data on which any interpretation is based. The decisions made during the interpretation process shall also be included in the evaluation records as specified in the evaluation plan.

The records shall contain sufficient information for each activity for effective performance of subsequent activities of the software product quality evaluation.

The records shall be kept in order to document the software product quality evaluation and to allow re-processing of the evaluation results.

1) Under preparation.

A software product quality evaluation report shall be prepared documenting evaluation activities and results of the evaluation.

NOTE Annex E provides a template for an evaluation report. Alternatively an evaluation report format can be used from a standard such as the Common Industry Format (for instance ISO/IEC 25062), or it can be created by evaluators.

When a tool is used to perform an evaluation action, reference to the tool shall be included in the evaluation report. The reference shall consist of the identification of the tool and of its supplier and the version of the tool.

A more detailed reference to the tool used shall be included in the evaluation records. It shall include the detailed configuration of the tool and any pertinent information needed to be able to repeat the evaluation action in order to obtain the same intermediate result.

6.3 Establish the evaluation requirements

The following should be inputs for this activity:

a) Software product quality evaluation needs;

b) Software product quality requirements specification;

c) Software product to be evaluated including intermediate products.

The following should be outcomes of this activity:

a) Specification of software product quality evaluation purposes;

b) Specification of software product quality evaluation requirements;

c) Specification of high level software product quality evaluation plan.

This activity consists of the following tasks.

6.3.1 Establish the purpose of the evaluation

The purpose of the software product quality evaluation shall be documented as a basis for the further evaluation activities and tasks.

Software product quality evaluation directly supports both the development and acquisition of software, which meets user and customer needs. The ultimate objective is to ensure that the product provides the required quality - that it meets the stated and implied needs of the users (including operators, recipients of the results of the software, and maintainers of software).

Software product quality can be evaluated within a defined quality structure throughout the life cycle stages relating to software implementation process and acquisition process defined in ISO/IEC 12207 software life cycle processes and ISO/IEC 15288 system life cycle processes.

The software product quality can be evaluated as an intermediate product or as a final product.

The purpose of evaluation of intermediate software product quality may be to:

- assure quality for the product;
- decide on the acceptance of an intermediate software product from a subcontractor;
- access the ongoing feasibility of the development project;
- decide on the completion of a life cycle stage and when to send products to the next stage;
- predict or estimate final software product quality;
- collect information on intermediate software products in order to control and manage the process.

The purpose of evaluation of final software product quality may be to:

- decide on the acceptance of the product;
- decide when to release the product;
- compare the product with competitive products;
- select a product from among alternative products;
- assess both positive and negative effects of a product when it is used;
- determine the cause of a failure in an investigation;
- decide when to enhance or replace the product.

6.3.2 Obtain the software product quality requirements

The stakeholders of the software product shall be identified.

NOTE 1 Information from the requester of the evaluation may be needed for identifying all stakeholders.

NOTE 2 Stakeholders to be identified are a person, party or organisation and may be involved in the evaluation. Two kinds of stakeholders are to be identified. One is a stakeholder of software product, such as developer, acquirer, independent evaluator, user, operator, recipient of the results of the software, maintainer, or supplier. Another is an evaluation requester who needs information about software quality, sponsors the evaluation and requires an evaluation report.

The software product quality requirements specified using a quality model shall be provided.

NOTE 3 ISO/IEC 25010 defines a quality model in terms of quality characteristics and subcharacteristics.

NOTE 4 ISO/IEC 25030 provides requirements and recommendations for specifying software product quality requirements.

NOTE 5 If an existed software product quality requirements specification is available it may be reused, reviewed and refined.

The extent to which the quality evaluation covers the specified software quality requirements shall be defined, taking into account the software product quality requirements, in order to produce the software product quality evaluation requirements. This decision should be based on constraints such as evaluation budget, target date for the evaluation, purpose of the evaluation and use criticality of the software product.

NOTE 6 As the software product quality requirements specification may be improved during its development or acquisition process it may be necessary to refine the software product quality evaluation in order to make them compatible with the evaluation purpose.

6.3.3 Identify product parts to be included in the evaluation

All product parts to be included in the evaluation shall be identified and documented.

The type of intermediate or final software product to be evaluated (e.g. requirements specification, design diagrams and test documentation) depends on the stage in the life cycle and the purpose of the evaluation. For instance, if the purpose of the evaluation is the selection of a product among alternative products, the products to be evaluated are mainly final software products or components. At the initial stages of the evaluation process it is not possible to identify a detailed list of products to be evaluated, since it also depends on the measures to be applied and on the artefacts to be produced along the software development life cycle. Therefore at that moment the components of the software product that are likely to be evaluated should be identified so that the initial list can be refined as the evaluation activities evolve.

6.3.4 Define the stringency of the evaluation

The evaluation stringency shall be defined.

NOTE 1 Stringency is defined in order to provide confidence in the software product quality according to its intended use and purpose of the evaluation.

The evaluation stringency should be related to a set of characteristics and subcharacteristics that establish expected evaluation levels which define the evaluation techniques to be applied and evaluation results to be achieved.

NOTE 2 ISO/IEC 15026 defines system and software integrity levels, see Annex A. The required integrity level of the software largely determines the rigor and formality of the evaluation.

NOTE 3 The following is an example of evaluation techniques to be applied to the functionality characteristic according to different evaluation levels requirements, from less demanding levels to more demanding levels:

- Functional or black box testing;
- Inspection of development documentation guided by checklists;
- Unit testing with test coverage criteria.

6.4 Specify the evaluation

The following should be inputs for this activity:

a) Specification of software product quality evaluation purposes;

b) Specification of software product quality evaluation requirements;

c) Specification of high level software product quality evaluation plan.

The following should be outcomes of this activity:

a) Specification of selected quality measures

b) Specification of decision criteria for software product quality measures;

c) Specification of decision criteria for software product quality assessment;

d) Specification of revised high level software product quality evaluation plan.

This activity consists of the following tasks.

6.4.1 Select quality measures (evaluation modules)

The evaluator shall select quality measures (evaluation modules) to cover all software quality evaluation requirements.

Software product quality evaluation requirements should be allocated to the software product components to which they are related in such a way that it is possible to define each appropriate quality measure that are used to evaluate the software product quality.

The software product quality evaluation methods shall be documented, taking into account the actions to be performed in order to achieve the evaluation results. When the evaluation method described is based on the use of a software tool, this tool shall be identified in the evaluation plan. Such identification shall include at least the name of the tool, its version identification and its origin (e.g. the supplier). The description of the evaluation methods shall be completed by the identification of product components on which the method is to

be applied. When the evaluation specification is such that expert analysis of the measurements is required in order to interpret the results, the interpretation procedure shall be specified.

NOTE 1 ISO/IEC 25042 specifies how to package evaluation methods as an evaluation module, which also includes information on techniques, inputs to be evaluated, data to be measured and collected and supporting procedures and tools.

NOTE 2 At this stage, the evaluation methods are related to elements in the evaluation specification, which are themselves related to evaluation requirements. Each of the evaluation methods is planned to be applied on the various product components submitted for evaluation. It can happen that several evaluation methods are to be applied to the same product component or consist of common parts.

NOTE 3 When evaluation requirements refer to evaluation levels, informative annex A provides guidance on which evaluation technique to use as a function of the evaluation level and the quality characteristic considered.

NOTE 4 ISO/IEC 25020 provides a Software Product Quality Measurement Reference Model, which may be applied to addresses the selection and construction of software quality measures.

NOTE 5 During the software quality requirements specification process, as defined in ISO/IEC 25030, it may be necessary to use quality measures as a reference to specify quality requirements, mainly those that relate to quantitative objectives.

Rigorous measurements are required to make reliable comparisons, either between products or with criterion values. Measurement procedures should measure the software quality characteristic (or subcharacteristic) they claim to be measuring with sufficient accuracy to allow criteria to be set and comparisons to be made. Data from checklists and expert opinion may not be reliable when comparing products with different attributes. Allowance should be made for possible measurement errors caused by measurement tools or human error.

NOTE 6 The Technical Reports ISO/IEC 25022, ISO/IEC 25023 and ISO/IEC 25024 give examples of measures that can be used as they are or adapted to specific needs.

NOTE 7 The type of measurement required will depend on the purpose of evaluation. If the primary purpose is to understand and correct deficiencies, several measurements may be made on the software to monitor and control improvements. A wide range of measures can be useful for these purposes, including checklists and expert opinion. The primary requirement is that the measurements correctly identify the impact that any changes in the software have on quality.

The evaluation specification shall comprise:

- the scope of the evaluation referring to the product components as identified in the product description;
- a cross-reference between the information needed to perform the evaluation and the product components and other related documents listed in the product description;
- a specification of measurements and verifications to be performed and the references to product components on which they are to be performed;
- a mapping between the specification of measurements and verifications and the evaluation requirements together with the reference to standards or the justification for each measurement or verification listed.

6.4.2 Define decision criteria for quality measures

Decision criteria shall be defined for the selected individual measures.

NOTE Decision criteria are numerical thresholds or targets used to determine the need for action or further investigation, or to describe the level of confidence in a given result. These will often be set with respect to quality requirements and corresponding evaluation criteria. Users may also use benchmarks, statistical control limits, historical data, customer requirements or other techniques to set decision criteria. For example, if estimated defect that exceeds the

acceptable threshold, then perform additional defect detection and removal activities. This information is documented elsewhere; a reference to that location is adequate (see ISO/IEC 25020).

6.4.3 Define decision criteria for evaluation

The evaluator should prepare a procedure for further summarization, with separate criteria for different quality characteristics, each of which may be in terms of individual subcharacteristics and quality measures, or a weighted combination of subcharacteristics and quality measures. The summarization results should be used as a basis for the software product quality assessment.

NOTE 1 The assessment criteria can be further used to support managerial decision, as the summarised quality can be compared with other aspects such as time and cost.

NOTE 2 To assess the quality of the product, the results of the evaluation of the different characteristics need to be summarised.

NOTE 3 ISO/IEC 25020 shows the relationship between quality characteristics, subcharacteristics and quality measures as a software product quality measurement reference model.

6.5 Design the evaluation

The following should be inputs for this activity:

a) Specification of software product quality evaluation requirements;

b) Specification of software product quality requirements;

c) Specification of selected quality measures (evaluation modules);

d) Specification of decision criteria for software product quality measures;

e) Specification of decision criteria for software product quality assessment;

f) Specification of revised high level software product quality evaluation plan.

The following should be outcomes of this activity:

a) Specification of detailed software product quality evaluation plan;

b) Specification of software product quality evaluation methods.

This activity consists of the following tasks.

6.5.1 Plan evaluation activities

The identified software product quality evaluation activities shall be scheduled, taking into account the availability of resources such as personnel, software tools and computers.

NOTE 1 In scheduling the evaluation methods, it is important to recognize that a high degree of interdependence exists between the various evaluation methods, i.e. information obtained by one method may influence the focus of another method. As the nature of evaluation is iterative, issues may be revisited as information is gained. Therefore the evaluation plan will likely be altered as the evaluation is conducted. For example, it may be common for more detailed levels of evaluation to be considered either as unnecessary or as an additional requirement once the evaluation progresses.

NOTE 2 The evaluation of software products may be carried out in stages at different points in a development life cycle, or all at once, at one point in the life cycle. Different individuals or groups may be responsible for performing different parts of the evaluation. When the evaluation is done in stages, the steps of the evaluation activity are repeated in each stage until no further work is required.

The evaluation plan should have no duplicating tasks within the evaluation. The evaluation plan should define decision points in the evaluation process which determine when and why the evaluation is to be considered complete (i.e. acceptance or rejection criteria) and is to be stopped. This should be done in order to decrease the risk of errors and to reduce the planned evaluation effort, considering at least the following items:

- The evaluation budget;
- Evaluation methods and adapted standards;
- Evaluation tools;
- Evaluation activities, including the schedule and resources involved.

The evaluation plan shall include the evaluation purpose. The evaluation plan should consider the evaluation context within the organisation (see in ISO/IEC 25001 the role of an evaluation support group and in ISO/IEC 25001 Annex A: the Quality Evaluation Project Plan template). The evaluation plan should include the following:

- Purpose of the software product quality evaluation;
- The organisations involved in the evaluation, such as a independent evaluation organisation, software product developers and acquirers´ organizational units;
- The evaluation budget;
- Information products expected from the evaluation;
- Schedule for the evaluation milestones;
- Responsibilities for the parties involved in the evaluation;
- Environment for evaluation;
- Evaluation methods and tools;
- Decision criteria for software product quality measures;
- Decision criteria for software product quality assessment;
- Adopted standards;
- Evaluation activities.

During the early evaluation some of the items of the evaluation plan can only be defined at a high level. Therefore, the evaluation plan shall be revised as the evaluation activities evolve, providing additional information that allows the plan to be adjusted or detailed.

NOTE 3 The high level evaluation plan is revised to the detailed level plan step by step in the successive evaluation activities and tasks.

6.6 Execute the evaluation

The following should be inputs for this activity:

a) Specification of detailed software product quality evaluation plan;

b) Specification of software product quality evaluation requirements;

c) Specification of software product quality requirements;

d) Specification of selected quality measures;

e) Specification of decision criteria for software product quality measures;

f) Specification of decision criteria for software product quality assessment;

g) Specification of software product quality evaluation methods;

h) Software product to be evaluated including intermediate products.

The following should be outcomes of this activity:

a) Results of software product quality measures;

b) Results of evaluation.

This activity consists of the following tasks.

6.6.1 Make measurements

The selected software product quality measures shall be applied to the software product and components, according to the evaluation plan, resulting in values on the measurement scales.

6.6.2 Apply decision criteria for quality measures

The decision criteria for the software product quality measures shall be applied to the measured values.

6.6.3 Apply decision criteria for evaluation

The set of decision criteria shall be summarised into subcharacteristics and characteristics, producing the assess results as a statement of the extent to which the software product meets quality requirements. The evaluation results should:

a) establish an appropriate degree of confidence that the software product is able to meet the evaluation requirements;

b) identify any specific deficiencies with regard to the evaluation requirements and any additional evaluations needed to determine the scope of those deficiencies;

c) identify any special limitations or conditions placed on the use of the software product;

d) identify any weaknesses or omissions in the evaluation itself and any additional evaluation that is needed;

e) identify any options for the use of the software product uncovered by the evaluation.

NOTE The 'apply decision criteria for evaluation' task is a synthetic evaluation of software product quality and is not of software process assessment. The evaluation results can be used to support managerial decision, as the summarised quality is compared with other aspects such as time and cost. The managerial decision includes the acceptance or rejection, or on the release or no-release of the software product.

6.7 Conclude the evaluation

The following are required inputs for this activity:

a) Specification of software product quality evaluation requirements;

b) Specification of actual results of software product quality evaluation plan;

c) Specification of software product quality evaluation methods;

d) Results of evaluation.

The following are outcomes of this activity:

a) Software product quality evaluation report.

This activity consists of the following tasks.

6.7.1 Review the evaluation result

The evaluator and the requester shall carry out a joint review of the evaluation results.

6.7.2 Create the evaluation report

Depending on how the evaluation report is intended to be used, it should include the following items:

a) the software product quality evaluation requirements;

b) the software product quality requirements;

c) the software product quality evaluation plan;

d) results from the measurements and analyses performed;

e) intermediate results or interpretation decisions, when specified by the evaluation plan;

f) any limitations, constraints, deficiencies, or exclusions in an evaluation activity, including their impact on the use, configuration, modification, or general maintenance of the software product over time;

g) the evaluators and their qualifications;

h) any differences between the product versions assessed and the corresponding evaluation inputs; i.e., documentation or courses;

i) resolutions or workarounds in the event of a deficiency;

j) any other information necessary to be able to repeat or reproduce the evaluation;

k) result of the evaluation.

As a result of the analysis of the evaluation activities the evaluation report should identify:

a) each deficiency, any relevant analysis, and how each deficiency was resolved. Resolution of deficiencies may include the fact that:

- one of the evaluation methods has provided assurance that the deficiency is not major;
- a satisfactory “workaround” can be found to alleviate the impact of the deficiency; e.g., modification to the product, disable or remove unneeded functionality, regenerate missing design requirements using reverse engineering;
- the original requirement is not mandatory and the deficiency can be accepted;

- the deficiency is acceptable provided that the use of the software product will be controlled by specific conditions or limitations;
- additional evaluation work is required to resolve the deficiency or gaps in the evaluation;

b) any additional evaluations performed to resolve any identified deficiencies:

- to determine the scope or impact of a deficiency;
- to establish confidence that there is no deficiency;
- to verify that a workaround is technically feasible and/or suitable and acceptable;
- to verify the correct and acceptable performance of the software once a design change or changes have been made to correct deficiency;

c) in a case where it is necessary to limit or control the use of the software product, whether the limitation:

- interferes with the software product meeting the mandatory requirements of the application;
- impacts on the application's design, budget, and schedule;
- requires additional evaluation work;
- introduces any possibility of failure in the application;

d) any exclusions from scope of evaluation and/or restrictions on the results for each evaluation, such as:

- 'This evaluation does not include a detailed review of the functionality of the product.'; or
- 'This software product is deemed to be qualified to the required integrity level provided a full evaluation of the required functionality for the product is completed successfully.';

e) the integrated results of all the evaluation activities to allow an overall conclusion for the evaluation of the software product to be made.

NOTE Extensive operating history may make up for a deficient software engineering process

Comments to the evaluation report shall be disposed and included in the final version of the report.

6.7.3 Review quality evaluation and provide feedback to the organisation

The evaluator shall review the results of the evaluation and the validity of the evaluation process, indicators and measures applied. Feedback from the review should be used in order to improve the evaluation process and evaluation techniques (evaluation modules). When it is necessary to improve the evaluation modules, the data collection for extra indicators should be included, in order to validate them for later use.

NOTE Quality evaluation review and feedback is described in ISO/IEC 25001.

6.7.4 Perform disposition of evaluation data

When the evaluation is completed the data and evaluation items shall be disposed according to requirements of the requester.

This shall be done in the one of the following ways, depending on the type of data:

- the documents submitted to the evaluation shall be either returned to the requester or archived for a specified duration or destroyed in a secure way,

- the evaluation report and the evaluation records shall be archived for a specified duration,
- all other data shall be either archived for a specified duration or destroyed in a secure way.

When the specified archiving duration expires for some data, it shall be either archived again for a specified duration or destroyed in a secure way.

Annex A
(informative)

Evaluation levels

As it seems difficult to obtain consensus on the generic specification of quality characteristics, including relation to subcharacteristics and to measures for the case to be evaluated, evaluation requirements may specify evaluation levels for the quality characteristics selected.

Evaluation levels are to be related to software integrity levels as defined in ISO/IEC 15026. If a software integrity level has been assigned to a software product submitted for evaluation, this software integrity level may be used to select evaluation requirements. In particular, the degree of rigour associated to the software integrity level may be used as a guide to select evaluation techniques.

Evaluation levels are related, on the one hand, to the importance attached by the requester to a given characteristic. The chosen level should be meaningful with regard to the assumed usage and environment of the software product (e.g. safety conditions, security constraints, economic risk, application constraints).

On the other hand, an evaluation level defines the depth or thoroughness of the evaluation in terms of evaluation techniques to be applied and evaluation results to be achieved. As a consequence evaluation at different levels gives different level of confidence in the quality of the software product. The level can be chosen independently for each characteristic.

This annex proposes four levels named A, B, C, and D. The levels constitute a hierarchy with A as the highest level and D as the lowest. At level A the most stringent evaluation techniques (taking into account reasonable amount of effort and time scale) are applied giving the highest confidence. Going down to level D gradually less stringent methods are used and consequently less effort is usually devoted to the evaluation.

The evaluation level, for each software characteristic, may change for different components of a large product (for instance it is likely that critical components with high reliability requirements are kept separated from the other components of a system).

The first section of this annex proposes guidance for selecting evaluation levels as function of the context of use of the product. The second one helps to select evaluation techniques.

A.1 Selection of evaluation levels

The evaluation levels may be selected independently for each of the relevant quality characteristics. When selecting the levels, several aspects should be considered. For example, important aspects are those related to safety, to economy, to security, to the environment and to the marketing of the product when appropriate.

For a relevant quality characteristic, the risks and consequences involved by the non conformity of the product to requirements relating to this characteristic, as well as benefits from high quality, should be assessed for all the relevant aspects. For some of these aspects, the tables below provide the relationship between risks and levels to be selected. When several aspects need to be considered, the most stringent level should be selected.

For the issues of economic risks and marketing benefits, the cost of the evaluation should be considered.

Table A.1 Example of evaluation level for safety aspects	
Evaluation level	**Consequences**
level D	Small damage to property; no risk to people
level C	Damage to property; threat of injury to people
level B	Threat to human lives
level A	Many people killed

Table A.2 Example of evaluation level for economy aspects	
Evaluation level	**Consequences**
level D	Negligible economic loss
level C	Significant economic loss (company affected)
level B	Large economic loss (company endangered)
level A	Financial disaster (company will not survive)

Table A.3 Example of evaluation level for security aspects	
Evaluation level	**Consequences**
level D	No specific risk identified
level C	Protection against error risk
level B	Protection of critical data and services
level A	Protection of strategic data and services

Table A.4 Example of evaluation level for environment related aspects	
Evaluation level	**Consequences**
level D	No environmental risk
level C	Local pollution
level B	Recoverable environmental damage
level A	Unrecoverable environmental damage

A.2 Selecting evaluation techniques from evaluation levels

In order to elaborate an evaluation specification to satisfy some evaluation requirements, it is necessary to specify measures. Measures are based on evaluation techniques that may be chosen according to quality characteristics and evaluation levels. In the following is proposed, for each quality characteristics, a list of evaluation techniques ranked from less demanding levels to more demanding levels.

Functionality:

- functional or black box testing;
- inspection of development documentation guided by checklists;
- unit testing with test coverage criteria.

Reliability:

- verification of the use of specific programming language facilities;
- analysis of fault tolerance construct in the software design and code;
- reliability growth modeling.

Usability:

- user interface and documentation inspection;
- verification of the conformity to interface standards;
- performing usage experiments with real users.

Efficiency:

- execution time measurement;
- benchmark testing;
- analysis of the design to determine the algorithmic complexity.

Maintainability:

- inspection of development documentation guided by checklists;
- code measures and programming rules verification;
- analysis of traceability between elements of development documentation.

Portability:

- analysis of software installation procedures;
- programming rules verification;
- analysis of software design.

Annex B
(informative)

Evaluation methods

This Annex B provides examples of evaluation methods which may be used for software product quality evaluation.

B.1 Review of user and technical product documentation (including on-line documentation)

Product documentation may provide all the information necessary to make an evaluation of the functionality and usability requirements as well as other issues such as portability and maintainability. It may be possible to gain access to pertinent software product documentation without actually purchasing it, either by borrowing the documents or purchasing a documentation set. Although reviewing the software product documentation may not be as efficient as taking a course or training, it may prove to be the most economical, especially if the evaluator has the pertinent expertise.

B.2 Evaluation based on supplier courses and training

Product courses are offered for many software products either through the supplier, or a third party. In the case of software products where no courses exist, it may be possible to arrange for special training from experienced users or developers of the software product. The advantage that product courses or training offer is allowing the evaluator to focus on specific areas and gain specific information on the functionality and usability of the software product in a short period of time. It may be possible to gain the same information through review of the software product documentation but that may prove more time consuming. The additional cost of a course or training needs to be weighed against efficiency in gaining information and the generality of the course material.

B.3 Assessment of software engineering process

The software engineering process assessment is a means of determining the quality of the software product by examining the interim products of the process; i.e. product quality plan, requirements specifications, architecture descriptions, detailed design descriptions, code listings, verification and validation records, code inspection and testing records, etc. To achieve this, it is necessary to define what constitutes an acceptable documentation baseline for software engineering processes which will provide adequate assurance in the quality of the resulting software product.

An acceptable baseline can be defined by tailoring ISO/IEC 12207 requirements to the target integrity level in order to specify the required development and related support activities. This consists of determining:

a) required processes;

b) required process output documentation;

c) the requirements on process and process output documentation.

This assessment may be coupled with a supplier process capability level determination as defined in ISO/IEC 15504-2. ISO/IEC 12207 may be used to define process and expected outcomes to be applied according to the software product integrity level requirements.

NOTE Most of processes for software engineering process are described in Implementation processes, Software implementation processes and Software support processes of ISO/IEC 12207 Software life cycle processes.

For systems with high integrity requirements additional process and product requirements may be required by sector standards, e.g., IEC 880, IEC 1508, DOA-167A , MOD-55, etc.

Then the supplier quality/development plan and associated methodology procedures can be used to assess supplier compliance to this target baseline. The level of compliance may then be determined by identifying the major deficiencies and assessing their potential impact on the quality of the software product. Additional evaluation or workarounds can address the impact of the deficiencies.

It should be recognized that there are a variety of software engineering processes which are effective for particular organizations and different types of software products. The evaluation process should have the flexibility to accommodate a variety of reasonable software engineering processes and methods.

It is recommended that the software engineering review be done in a staged manner. When it is deemed that the integrity level of software does not require a full evaluation of the software engineering process, the review may be halted after stage I or II.

B.4 Review of operating history with the supplier

A review of the operating history with the supplier can provide a very effective means of indicating the quality of the software product. This is achieved by reviewing the sales figures of the software product and details of the industries and the applications in which it is used. This review also addresses the history of revisions to the software, the way in which revisions are maintained, the way in which deficiency reports from customers are dealt with, and details of known deficiencies. The most convenient way to conduct the review is by interviewing the supplier's engineering, sales, and customer support staff and by examining any supporting records.

B.4.1 Operating history requirements

a) The sales figures should be at least six months old; i.e., the number of sales used in the evaluation will only include those sold over six months before the evaluation takes place. This criteria is based on the fact that it may take up to six months for the software product to be delivered, installed, commissioned, and placed in service.

b) The software product should have gone through at least one major revision and there should be viable operating history data available on that revision. This is based on the assumption that the quality of the software product will depend on the amount of refinement it has gone through.

c) There is a means for users of the software product to feed back deficiency reports to the supplier, and that there is evidence of this happening, and of resulting dispositions being implemented.

B.4.2 Operating history review

Review of product operating history should determine:

a) Whether the software was produced by modifying another product and whether that product's operating history can be used. This would be contingent on the number of changes and the extent of the changes made to the software product;

b) The number of unit-years of operation for the software product. This is calculated by the following steps:

   1) Calculate the Sales Year = (initial sales [total sales in first year] + cumulative final total up to six months from present [assuming it typically takes 6 months delay before a unit is operational]) * (number of years the software product is in the market) / 2.

2) Calculate Unit-Years of operation = (Sales Year * duty cycle [percentage of the time the software is operational] * percentage of units actually in operation [this is pertinent, for example, to firmware in which a number of units of the host hardware may be kept as spares]).

c) If the operating experience provides evidence pertinent to the functionality required by the application. For example, is it used in similar applications by other customers?

d) If the operating experience provides evidence that the breadth of the software product's functionality has been exercised. For example, has it been used in a wide variety of applications and industries?

e) The number of unit-years of operation for each revision of the software and for each special option of the software product;

f) The differences between the revisions, the extent of the changes, and whether the changes are isolated;

g) Whether documented evidence exists to support the operating history data;

h) How revisions of the software product and revisions of any related hardware are controlled and tracked;

I) Whether it is possible to order a specific revision of the software product and the implications of ordering a revision which is not current;

j) The supplier process for accepting and disposing of deficiency reports received from customers;

k) Whether customers are kept abreast of reported deficiencies;

l) Any outstanding deficiencies in the software and their impact.

B.5 Review of operating history with customers

A review of the operating history with actual customers who use or have used the software product in one of their applications allows the evaluator to get relatively unbiased answers to specific questions based on realistic operating conditions. Depending on how similar the customer applications are to the proposed application, the assurance gained in overall quality or specific functionality may be as useful as that obtained from prototyping or even extensive trial usage. The most convenient way to conduct the review is by interviewing the customers at their site of operations and possibly even viewing demonstrations or supporting records.

The evaluator conducting an operating history review with customers should:

a) Establish the degree of similarity of the application(s) to the proposed application;

b) Attempt to view the software product in operation or obtain other supporting evidence;

c) Question the customer(s) on the form and quality of support provided by the supplier;

d) Determine the amount of error free operation.

B.6 Review of supplier capability, support, and quality system

Evaluation of the supplier's level of support capability and ability to maintain the software should address:

a) financial stability, experience, and capabilities;

b) product development environmental support, including assembly tools, operating system used, and maintenance and use of other component/library objects;

c) product interfaces to other products or product groupings, including interface standards;

d) contingencies for third party involvement;

e) sufficient resource commitment to product support;

f) sufficient customer base to justify continued support;

g) sufficient maintenance service for bug fixes, environmental support;

h) adequate references from installed base;

i) formalized and sufficient release and revision control procedures and evidence of practice;

j) formalized regression testing and formal evaluation of design changes;

k) documented and practiced problem reporting and resolution procedures;

l) quality system in place;

m) standards used for hardware and software;

n) plans for future development; i.e., strategy in place relative to current market position;

o) product warranty.

B.7 Prototyping and other evaluation methods

B.7.1 Prototyping

Prototyping is a method of evaluation that can be used to resolve or fine tune requirements, to determine the technical feasibility of using the software product, or to eliminate unknowns or technical risks associated with specific functionality or usability requirements and their implementation. Prototyping may or may not utilize the full suite of functionality of the software product or address the entire set of requirements of the application.

It should be noted that prototyping often requires the supplier to provide access to specialized equipment, personnel and documentation. Costs and schedule implications for these and other factors such as special environmental conditions or services should be considered in determining the feasibility of prototyping the software product.

In addition to the general requirements for an evaluation, prototyping should:

a) use examples which adequately address the requirements being assessed and provide realistic re-creation and simulation of key operating parameters;

b) be adequately documented so that it may be repeated by a third party;

c) make use of historical data pertinent to the proposed application where available.

B.7.2 Other evaluation methods

Evaluation techniques can be related to evaluation levels and software quality characteristics (see Annex D). The evaluation levels can correspond to integrity levels for the evaluation. Additional evaluation methods include:

a) analysis of software architecture design (maintainability);

b) fault tree analysis of software (reliability);

c) statistical random usage based testing of software product (reliability);

d) dynamic analysis of code to check syntax and semantics for correctness (reliability);

e) hazards analysis of software design (reliability);

f) review of software requirements specification (functionality);

g) code inspection (functionality);

h) black-box testing of software (functionality);

i) benchmark testing (efficiency);

j) analysis of requirements traceability (maintainability);

k) simulated faults at the interfaces between components (reliability).

For complex software of high integrity, fault tree analysis or hazards analysis of software design can be used to isolate “critical” software modules for evaluation. This may eliminate the need to perform rigorous evaluation on software that has no impact on the integrity of the application.

Annex C
(informative)

Example of Cost-Effectiveness Ranking of Evaluation Methods

This table presents a *qualitative* estimated ranking of the relative effectiveness and the gross relative cost of evaluation methods relative to specific product quality characteristics. This effectiveness rating assumes that the evaluation is conducted successfully and to the appropriate degree of rigor. This table can be used to select the target inputs that need to be evaluated in order to fully assess the adequacy of product quality characteristics against those specified in the software requirements specification. The evaluations required can be used to estimate total cost of product evaluation.

Table C.1 — Example of cost-effectiveness ranking of evaluation method

Target Input	Evaluation Method	Cost Ranking*	Effectiveness* Ranking of Evaluation Methods per Software Quality Characteristic					
			Functionality	Reliability	Usability	Efficiency	Maintainability	Portability
Final product documents	**Evaluation based on supplier Courses and Training**	Low	High	Low	High	Medium	Low	High
Intermediate product documents	**Assessment of software engineering process**	High	Low	Medium	Low	Medium	High	High
Operating History - Supplier	**Review of operating history with supplier**	Medium	Medium	High	Low	Low	Medium	Medium
Operating History – Customer	**Review of operating history with customer**	Medium	High	Medium	High	Medium	High	High
*NOTE – Cost and effectiveness rankings shown are relative to the extent of the evaluation conducted								

Annex D
(informative)

Relationships between software product quality evaluation process reference model and software and system life cycle processes

Activities and tasks of evaluation process reference model can be performed associatively as a part of activities and tasks of Software Life Cycle Processes (ISO/IEC 12207) and/or System Life Cycle Processes (ISO/IEC 15288), when an evaluation is performed concurrently with software product development.

Table D.1 — Typical relationships between software product evaluation reference model and software and system life cycle processes (ISO/IEC 12207 and 15288)

Activities and tasks of software product evaluation reference model in ISO/IEC 25040	Typical related processes in ISO/IEC 12207	Typical related processes in ISO/IEC 15288
6.3 Establish evaluation requirements		
6.3.1 Establish the purpose of evaluation		
6.3.2 Obtain the software product quality requirements	6.1.1 Acquisition process	6.1.1 Acquisition process
6.3.3 Identify product parts to be included in the evaluated	6.2.2 Supply process	6.2.2 Supply process
6.3.4 Define the stringency of the evaluation	6.2.5 Quality management process	6.2.5 Quality management process
6.4 Specify the evaluation		
6.4.1 Select quality measures	6.4.1 Stakeholder requirements definition process	6.4.1 Stakeholder requirements definition process
6.4.2 Define decision criteria for quality measures		
6.4.3 Establish decision criteria for evaluation	7.1.2 Software requirements analysis process	7.1.2 System requirements analysis process
6.5 Design the evaluation		
6.5.1 Plan evaluation activities		
6.6 Execute the evaluation		
6.6.1 Make measurements	6.3.7 Measurement process	6.3.7 Measurement process
6.6.2 Apply decision criteria for measures	6.4.5 Integration process	6.4.5 Integration process