



ISO/IEC JTC1/SC7
Software Engineering
Secretariat: CANADA (SCC)

ISO/IEC JTC1/SC7 N1949

1998-06-26

Document Type	FCD Ballot
Title	FCD 9126-1.2 Information Technology - Software product quality - Part 1: Quality model.
Source	SC7 Secretariat
Project	07.13.01.01
Status	2nd FCD Ballot
References	N1948
Action ID	ACT
Due Date	1998-10-26
Mailing Date	1998-06-26
Distribution	SC7_AG; P, O & L Members
Medium	Encoded Acrobat
No. of Pages	25
Disk	
Note	



ISO/IEC JTC1/SC7 FCD 9126-1.2	
Date 1998-06-26	Reference number ISO/JTC 1/SC 7 N1949
Supersedes document N1838	

THIS DOCUMENT IS STILL UNDER STUDY AND SUBJECT TO CHANGE. IT SHOULD NOT BE USED FOR REFERENCE PURPOSES.

ISO/JTC 1/SC 7 Committee Title Software Engineering Secretariat: Standards Council of Canada (SCC)	Circulated to P- and O-members, and to technical committees and organizations in liaison for: X voting by (P-members only) 1998-10-26 Please return all votes and comments in electronic form directly to the SC 7 Secretariat by the due date indicated.
---	---

ISO/IEC JTC1/SC7

Title: FCD 9126-1.2 Information Technology - Software product quality - Part 1: Quality model.

Project: 07.13.01.01

Introductory note: See page ii of the document

Medium: Encoded Acrobat

No. of pages: 25



Vote on FCD 9126-1.2	
Date of circulation 1998-06-26	Reference number ISO/JTC 1/SC 7 N1949
Closing date 1998-10-26	

ISO/JTC 1/SC 7 Committee Title Software Engineering Secretariat: Standards Council of Canada (SCC)	Circulated to P-members of the committee for voting Please return all votes and comments in electronic form directly to the SC 7 Secretariat by the due date indicated.
---	--

ISO/IEC JTC1/SC7

Title: FCD 9126-1.2 Information Technology - Software product quality - Part 1: Quality model.

Project: 07.13.01.01

Vote:

- ☐ APPROVAL OF THE DRAFT AS PRESENTED
- ☐ APPROVAL OF THE DRAFT WITH COMMENTS AS GIVEN ON THE ATTACHED
- ☐ general:
- ☐ technical:
- ☐ editorial:
- ☐ DISAPPROVAL OF THE DRAFT FOR REASONS ON THE ATTACHED
- ☐ Acceptance of these reasons and appropriate changes in the text will change our vote to approval
- ☐ ABSTENTION (FOR REASONS BELOW):

P-member voting:

[National Body \(Acronym\)](#)

Date:

[YYYY-MM-DD](#)

Submitted by:

[Your Name](#)

NOTE: do NOT submit this form for email voting. Simply type your vote (with comments where applicable) into an email message and send to sc7@qc.bell.ca.

5
TITLE: ISO/IEC FCD 9126-1.2:
Information Technology - Software product quality -
Part 1: Quality model

10
DATE: 19-Jun-98

SOURCE: JTC1/SC7/WG6

15
WORK ITEM: Project 7.13.01.1

STATUS: Version 9.5 agreed in Johannesburg

DOCUMENT
TYPE: second FCD

20
ACTION: For vote

25
PROJECT
EDITOR: Prof. Motoei AZUMA
Department of Industrial Eng. and Management
Waseda University
3-4-1, Okubo, Shinjuku-ku, Tokyo 169, Japan
FAX: +81-3-3200-2567
azuma@azuma.mgmt.waseda.ac.jp

30
DOCUMENT
EDITOR: Nigel BEVAN
National Physical Laboratory
NPL Usability Services
Teddington, Middx. TW11 0LW, United Kingdom
FAX: +44 181 943 6306

35
Nigel.Bevan@ npl.co.uk

40
CO-EDITOR: Anna GIANNETTI
SOGEI SpA
Software Quality Management
Via M.Carucci 99
I-00143 Rome, Italy
fax:+39-6-50957201
anna@sogei.it

45
REVIEWERS V Godamunne, T Komiyama

Reference number of working document: **ISO/JTC 1/SC 7 N**

Date: 19-Jun-98

Reference number of document: **ISO/FCD 9126-1.2**

Committee identification: ISO/JTC 1/SC 7/WG 6

Secretariat: ANSI

5

Information Technology — Software Product Quality — Part 1: Quality model

Document type: International standard

Document subtype:

Document stage:

Document language: E

Contents

	1 Scope	1
	2 Conformance	2
	3 Normative references	2
5	4 Terms and definitions	2
	5 Quality model framework	2
	5.1 Approaches to quality	2
	5.2 Product quality and the life-cycle	3
	5.3 Items to be evaluated	4
10	5.4 Evaluation using a quality model.....	4
	6 Software quality characteristics	5
	6.1 Functionality.....	5
	6.2 Reliability	6
	6.3 Usability	6
15	6.4 Efficiency.....	7
	6.5 Maintainability.....	7
	6.6 Portability.....	7
	7 Quality in use characteristics.....	8
	7.1 Quality in use	8
20	A.1 Software metrics.....	9
	A.2 Quality in use metrics	10
	A.3 Choice of metrics and measurement criteria.....	10
	A.4 Requirements for measurements used for comparison	11
	Annex B (informative) Definitions from other standards	17
25	Annex C (informative) History of the work.....	21
	Annex D (informative) Bibliography.....	23

Foreword

5 ISO (the International Organisation for Standardisation) and IEC (the International Electrotechnical Commission) form the specialised system for world-wide standardisation. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organisation to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organisations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

10 International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. 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.

15 International Standard ISO/IEC 9126-1 was prepared by Joint Technical Committee ISO/IEC JTC1 *Information Technology*.

ISO/IEC 9126 consists of the following parts under the general title *Information Technology - Software product quality*

Part 1: Quality model

20 *Part 2: External Metrics*

Part 3: Internal Metrics

Part 4: Quality in use metrics

Annex A forms a normative part of this part of ISO/IEC 9126. Annexes B, C and D are for information only.

25

Introduction

Computers are being used in an increasingly wide variety of application areas, and their correct operation is often critical for business success or human safety. Developing or selecting high quality software products is therefore of prime importance. Comprehensive specification and evaluation of software product quality is a key factor in ensuring adequate quality. This can be achieved by defining appropriate quality characteristics, taking account of the purpose of usage of the software product. It is important that the software product is evaluated for every relevant quality characteristic, using validated or widely accepted metrics.

ISO/IEC 9126 (1991): Software product evaluation - Quality characteristics and guidelines for their use, which was developed to support these needs, defined six quality characteristics and described a software products evaluation process model.

As quality characteristics, subcharacteristics and associated metrics can be useful not only for evaluating a software product but also for defining quality requirements and other usage, ISO/IEC 9126 (1991) has been replaced by two related multipart standards: ISO/IEC 9126 (Software product quality) and ISO/IEC 14598 (Software product evaluation). The software quality characteristics defined in this part of ISO/IEC 9126 can be used to specify customer and user needs for both functional and non-functional requirements.

This part of ISO/IEC 9126 is a revision of ISO/IEC 9126 (1991), and retains the same software quality characteristics. The major differences are:

- the introduction of normative subcharacteristics, most of which are based on the informative subcharacteristics in ISO/IEC 9126 (1991);
- the specification of a quality model;
- the introduction of quality in use;
- removal of the evaluation process (which is now contained in ISO/IEC 14598-1);
- co-ordination of the content with ISO/IEC 14598-1.

The relationship between the standards in the ISO/IEC 9126 and ISO/IEC 14598 series (see Annex D) is illustrated in Figure 1.

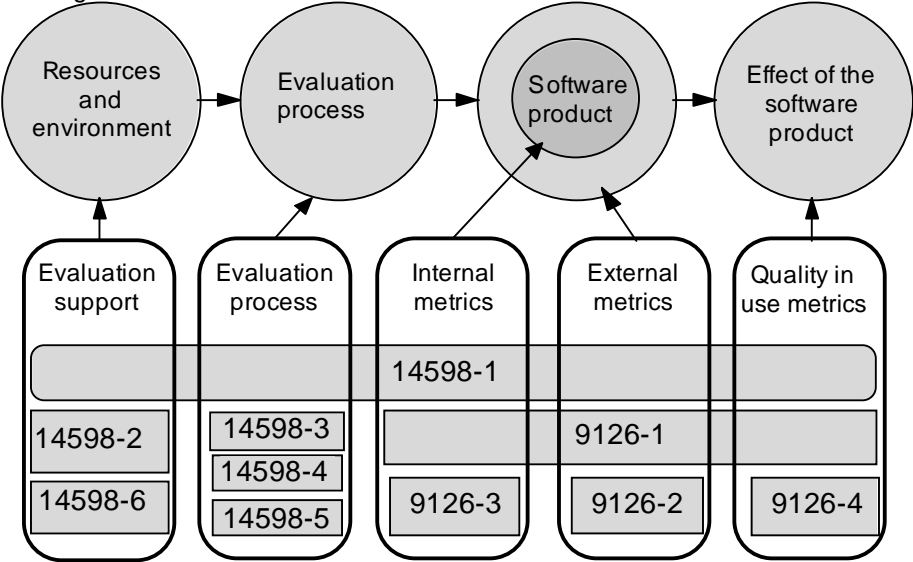


Figure 1 - Relationship between ISO/IEC 9126 and ISO/IEC 14598 standards

Information Technology Software product quality Part 1: Quality model

1 Scope

This part of ISO/IEC 9126 describes a model for software product quality which includes internal quality, external quality and quality in use. It specifies six characteristics for internal and external quality, which are further sub-divided into subcharacteristics. These subcharacteristics are manifested externally when the software is used as a part of a computer system, and are a result of internal software attributes. This part of ISO/IEC 9126 does not elaborate the model for internal and external quality below the level of subcharacteristics.

This part of ISO/IEC 9126 also specifies four quality in use characteristics which are the combined effect of the software quality characteristics for the user. This part of ISO/IEC 9126 does not elaborate the model for quality in use below the level of characteristics.

The characteristics defined are applicable to every kind of software, including computer programs and data contained in firmware. The characteristics and subcharacteristics provide consistent terminology for software quality. They also provide a framework for specifying quality requirements for software, and making trade-offs between software product capabilities.

Normative Annex A provides recommendations and requirements for software product metrics and quality in use metrics. Examples of these metrics are contained in other parts of ISO/IEC 9126. These metrics are applicable when specifying quality requirements and design goals for software products, including intermediate products. An explanation of how this quality model can be applied in software product evaluation is contained in ISO/IEC 14598-1.

This part of ISO/IEC 9126 is intended for use by developers, acquirers, quality assurance staff and independent evaluators, particularly those responsible for specifying and evaluating software product quality. It enables software product quality to be specified and evaluated from different perspectives by those associated with acquisition, requirements, development, use, evaluation, support, maintenance, quality assurance and audit of software. Examples of uses of the quality model defined in this part of ISO/IEC 9126 are to:

- validate the completeness of a requirements definition;
- identify software requirements;
- identify software design objectives;
- identify software testing objectives;
- identify quality assurance criteria;
- identify user acceptance criteria for a completed software product.

NOTE 1 This part of ISO/IEC 9126 can be used in conjunction with ISO/IEC 15504 (which is concerned with the software process assessment) to provide:

- a framework for software product quality definition in the customer-supplier process;
- support for review, verification and validation, and a framework for quantitative quality evaluation, in the support process;
- support for setting organisational quality goals in the management process.

NOTE 2 This part of ISO/IEC 9126 can be used in conjunction with ISO/IEC 12207 (which is concerned with the software lifecycle) to provide:

- a framework for software quality requirements definition in the primary lifecycle process;
- support for review, verification and validation in supporting life cycle processes.

NOTE 3 This part of ISO/IEC 9126 can be used in conjunction with ISO 9001 (which is concerned with quality assurance processes) to provide:

- support for setting quality goals;
- support for design review, verification and validation.

2 Conformance

Any statement of conformance to this part of ISO/IEC 9126 shall explain how the appropriate characteristics and subcharacteristics from clauses 6 and 7 were selected , and the reasons for any exclusions.

- 5 Requirements for metrics are contained in A.4.

3 Normative references

The following normative document contains provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9126. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 9126 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 14598-1:1998, *Information Technology - Software product evaluation - Part 1: General overview*

4 Terms and definitions

For the purposes of all parts of ISO/IEC 9126, the following definition and the definitions contained in ISO/IEC 14598-1 apply.

NOTE The relevant definitions are reproduced in informative annex B.

4.1

level of performance

the degree to which the needs are satisfied, represented by a specific set of values for the quality characteristics

5 Quality model framework

This clause describes a quality model which explains the relationship between different approaches to quality. A specific implementation of this quality model is given in clauses 6 and 7.

5.1 Approaches to quality

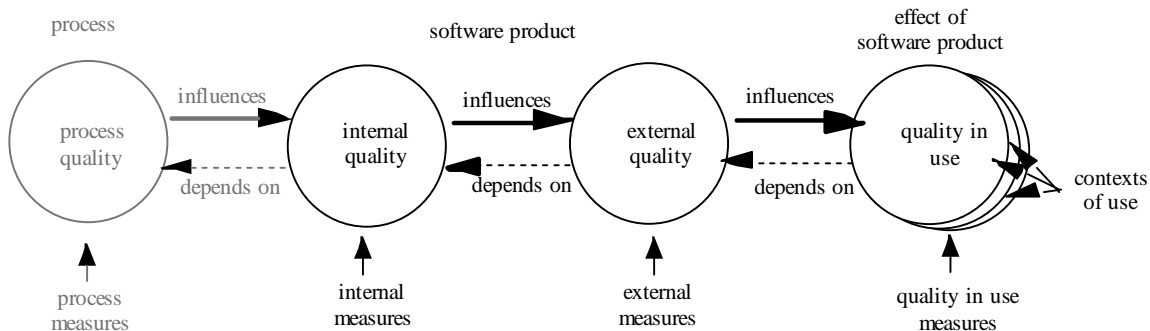


Figure 2 - Quality in the life-cycle

Evaluation of software products in order to achieve software product quality is one of the processes in the software development life-cycle. Software quality can be evaluated by measuring internal attributes (typically static measures of intermediate products), or by measuring external attributes (typically by measuring the behaviour of the code when executed). The objective is for the product to have the required effect in a particular context of use (Figure 2).

Process quality (the quality of any of the life cycle processes defined in ISO/IEC 12207) contributes to improving product quality, and product quality contributes to improving quality in use. Therefore, assessing and improving a process is a means to improve product quality, and evaluating and improving product quality is one means of improving quality in use. Similarly, evaluating quality in use

can provide feedback to improve a product, and evaluating a product can provide feedback to improve a process.

Appropriate quality evaluation processes are required to support the measurement of quality during development. Appropriate internal attributes of the software are a pre-requisite for achieving the required external behaviour, and appropriate external behaviour is a pre-requisite for achieving quality in use (Figure 2).

The requirements for software product quality will generally include criteria for internal quality, external quality and quality in use, to meet the needs of developers, maintainers, acquirers and end users.

5.2 Product quality and the life-cycle

Quality changes with the life-cycle of the software (e.g., required product quality at the start of the life-cycle differs from actual or delivered product quality). The assessment of quality is also a reflection of diverse points of view. It is necessary to define these diverse views of quality and changes in quality with the life-cycle, in order to manage quality properly at each stage of the life-cycle. The following are the different views of product quality at different stages in the software life-cycle (see ISO/IEC 14598-1:1998 Figure 4):

15

Goal Quality is the necessary and sufficient quality that reflects real user needs.

ISO 8402 defines quality in terms of the ability to satisfy stated and implied needs. However, needs stated by a user do not always reflect real user needs, because a user is often not aware of his real needs and needs may change after they are stated. So goal quality cannot be completely defined at the beginning of design. Yet, developers must keep this goal in mind and try to get closer to it. Goal quality does not necessarily mean perfect quality, but necessary and sufficient quality. Requirements for goal quality can be specified by quality in use metrics. It is often useful to document particular envisaged scenarios of use to more clearly understand the user needs and any potential conflicts. Goal quality can be evaluated by measuring quality in use when the product is complete, and requirements for quality in use should if possible be included in the quality requirements specification.

External Quality Requirements are what is actually stated in the quality requirements specification.

External quality requirements should be used as the target for initial validation. External quality requirements for all the quality characteristics defined in this part of ISO/IEC 9126 should be stated in the quality requirements specification using external metrics. Not only the optimal requirements, but also the minimum requirements, should be stated so that both the user and the developer can avoid unnecessary cost and schedule overruns.

Internal Quality Requirements specify the internal quality represented in the core parts or backbone of software design, including software architecture, program structure, and user interface design strategy. Specific internal quality requirements should be specified quantitatively using internal metrics.

Internal quality is a reflection of the design philosophy and strategy. It includes internal software properties which are mainly related to the programming environment, and also internal quality requirements for the quality characteristics defined in this part of ISO/IEC 9126. Details of software quality may be improved during code implementation and testing, but the fundamental nature of the software product quality represented by internal quality remains unchanged.

Estimated (or Predicted) External Quality is the external quality that is estimated or predicted for the end software product at each stage of development, and based on knowledge of the internal quality.

External quality may be estimated and predicted during development for each quality characteristic defined in this part of ISO/IEC 9126. For the purposes of prediction, technology should be developed to show the co-relation between external quality and internal quality.

External Quality is the quality of the delivered product, which is typically evaluated by testing in a simulated environment with simulated data using external metrics.

During testing, most faults should be discovered and eliminated. However, some faults may still remain after testing. As it is difficult to correct the software architecture or other fundamental design aspects of the software, the fundamental design usually remains unchanged throughout testing.

Quality in Use is the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself (quality in use is defined in clause 7).

NOTE 'Users' refers to any type of intended users, including both operators and maintainers, and their requirements can be different.

The quality in the users' environment may be different from that in the developers' environment, because some functions may not be visible to a user, or may not be used by a user. The user evaluates only those attributes of software, which are visible during actual use. Sometimes, software attributes specified by an end user during the requirements analysis phase, no longer meet the user requirements

when the product is in use, because of changing user requirements and the difficulty of specifying implied needs.

5.3 Items to be evaluated

- 5 Items can be evaluated by direct measurement, or indirectly by measuring their consequences. For example, a process may be assessed indirectly by measuring and evaluating its product, and a product may be evaluated indirectly by measuring the task performance of a user (using quality in use metrics). Software never runs alone, but always as part of a larger system consisting of other software products with which it has interfaces, hardware, human operators, and work flows. The completed software product can be evaluated by the levels of the chosen external metrics. These metrics describe its interaction with its environment, and are assessed by observing the software in operation. Quality in use (the capability of a product to meet stated and implied needs) can be measured by the extent to which a product used by specified users meets their needs to achieve specified goals with effectiveness, productivity, safety and satisfaction. This will normally be complemented by measures of more specific software quality characteristics, which is also possible earlier in the development process.
- 10 At the earliest stages of development, only resources and process can be measured. When intermediate products (specifications, source code, etc.) become available, these can be evaluated by the levels of the chosen internal metrics. These metrics can be used to predict values of the external metrics. They may also be measured in their own right, as essential pre-requisites for external quality. A further distinction can be made between the evaluation of a software product and the evaluation of the system in which it is executed. For example, the reliability of a system is assessed by observing all failures due to whatever cause (hardware, software, human error, etc.), whereas the reliability of the software product is assessed by extracting from the observed failures only those that are due to faults (originating from requirements, design or implementation) in the software. Also, where the boundary of the system is judged to be, depends upon the purpose of the evaluation, and upon who the users are.
- 15 NOTE For example, if the users of an aircraft with a computer-based flight control system are taken to be the passengers, then the system upon which they depend includes the flight crew, the airframe, and the hardware and software in the flight control system, whereas if the flight crew are taken to be the users, then the system upon which they depend consists only of the airframe and the flight control system.

5.4 Evaluation using a quality model

- 30 Software product quality should be evaluated using a defined quality model. The quality model should be used when setting quality goals for software products and intermediate products. Both software quality and quality in use should be hierarchically decomposed into a quality model composed of characteristics and subcharacteristics which can be used as a checklist of issues related to quality. Clauses 6 and 7 define a hierarchical quality model for software quality and quality in use (although other ways of categorising quality may be more appropriate in particular circumstances).
- 35 It is not practically possible to measure all internal and external subcharacteristics for all parts of a large software product. Similarly it is not usually practical to measure quality in use for all possible user-task scenarios. Resources for evaluation need to be allocated between the different types of measurement dependent on the business objectives and the nature of the product and design process.

6 Software quality characteristics

The quality model in this part of ISO/IEC 9126 categorises software quality attributes into six characteristics (functionality, reliability, usability, efficiency, maintainability and portability), which are further sub-divided into subcharacteristics (Figure 3). The subcharacteristics can be measured by internal or external metrics.

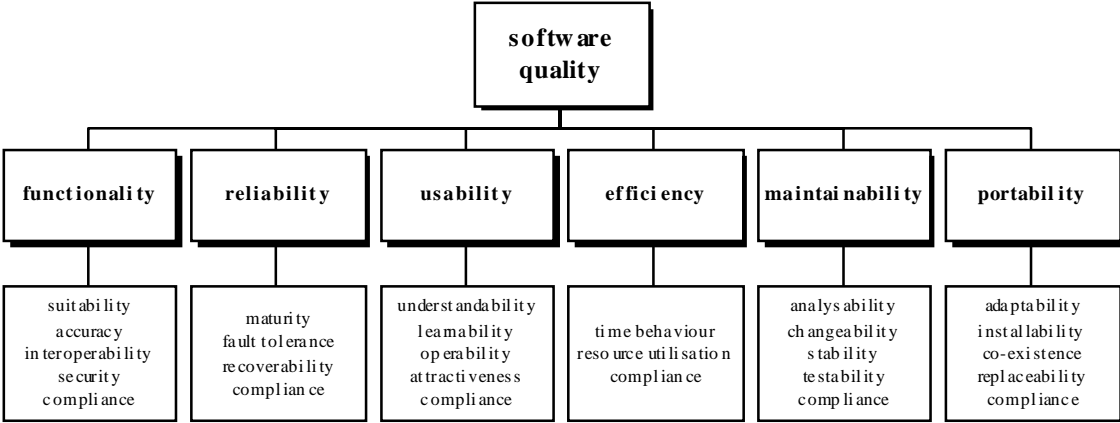


Figure 3 - Internal and external quality

Definitions are given for each quality characteristic and the subcharacteristics of the software which influence the quality characteristic. For each characteristic and subcharacteristic, the capability of the software is determined by a set of internal attributes which can be measured. Examples of internal metrics are given in ISO/IEC 9126-3. The characteristics and subcharacteristics can be measured externally by the extent to which the capability is provided by the system containing the software. Examples of external metrics are given in ISO/IEC 9126-2.

NOTE Some of the characteristics in this part of ISO/IEC 9126 relate to dependability. Dependability characteristics are defined for all types of systems in IEC 50(191), and where a term in this part of ISO/IEC 9126 is also defined in IEC 50(191), the definition given is broadly compatible.

6.1 Functionality

The capability of the software product to provide functions which meet stated and implied needs when the software is used under specified conditions.

NOTE 1 This characteristic is concerned with what the software does to fulfil needs, whereas the other characteristics are mainly concerned with when and how it fulfils needs.

NOTE 2 For the stated and implied needs in this characteristic, the note to the definition of quality applies, (see B.21).

NOTE 3 For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use (see clause 7).

6.1.1 Suitability

The capability of the software product to provide an appropriate set of functions for specified tasks and user objectives.

NOTE 1 Examples of appropriateness are task oriented composition of functions from constituent sub-functions, capacities of tables.

NOTE 2 Suitability corresponds to suitability for the task in ISO 9241-10

NOTE 3 Suitability also affects operability.

6.1.2 Accuracy

The capability of the software product to provide the right or agreed results or effects.

NOTE This includes the needed degree of precision of calculated values.

6.1.3 Interoperability

The capability of the software product to interact with one or more specified systems.

NOTE Interoperability is used in place of compatibility in order to avoid possible ambiguity with replaceability (see 6.6.4).

6.1.4 Security

The capability of the software product to protect information and data so that unauthorised persons or systems cannot read or modify them and authorised persons or systems are not denied access to them.

5 [ISO/IEC 12207: 1995]

NOTE 1 This also applies to data in transmission.

NOTE 2 **Safety** is defined as a characteristic of quality in use, as it does not relate to software alone, but to a whole system.

6.1.5 Compliance

10 The capability of the software product to adhere to standards, conventions or regulations in laws and similar prescriptions.

6.2 Reliability

The capability of the software product to maintain a specified level of performance when used under specified conditions

15 NOTE 1 Wear or ageing does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation. Failures due to these faults depend on the way the software product is used and the program options selected rather than on elapsed time.

20 NOTE 2 The definition of reliability in ISO/IEC DIS 2382-14:1994 is "The ability of functional unit to perform a required function...". In this document, functionality is only one of the characteristics of software quality. Therefore, the definition of reliability has been broadened to "maintain a specified level of performance..." instead of "...perform a required function"

6.2.1 Maturity

The capability of the software product to avoid failure as a result of faults in the software.

6.2.2 Fault tolerance

25 The capability of the software product to maintain a specified level of performance in cases of software faults or of infringement of its specified interface.

NOTE The specified level of performance may include fail safe capability.

6.2.3 Recoverability

30 The capability of the software product to re-establish a specified level of level of performance and recover the data directly affected in the case of a failure.

NOTE 1 Following a failure, a software product will sometimes be down for a certain period of time, the length of which is assessed by its recoverability.

35 NOTE 2 **Availability** is the capability of the software product to be in a state to perform a required function at a given point in time, under stated conditions of use. Externally, availability can be assessed by the proportion of total time during which the software product is in an up state. Availability is therefore a combination of maturity (which governs the frequency of failure), fault tolerance and recoverability (which governs the length of down time following each failure).

6.2.4 Compliance

40 The capability of the software product to adhere to standards, conventions or regulations relating to reliability.

6.3 Usability

The capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.

45 NOTE 1 Some aspects of functionality, reliability and efficiency will also affect usability, but for the purposes of ISO/IEC 9126 are not classified as usability.

NOTE 2 Users may include operators, end users and indirect users who are under the influence of or dependent on the use of the software. Usability should address all of the different user environments that the software may affect, which may include preparation for usage and evaluation of results.

6.3.1 Understandability

50 The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.

NOTE This will depend on the documentation and initial impressions given by the software.

6.3.2 Learnability

The capability of the software product to enable the user to learn its application.

55 NOTE The internal attributes correspond to suitability for learning as defined in ISO 9241-10.

6.3.3 Operability

The capability of the software product to enable the user to operate and control it.

NOTE 1 Aspects of suitability, changeability, adaptability and installability may affect operability.

60 NOTE 2 Operability corresponds to controllability, error tolerance and conformity with user expectations as defined in ISO 9241-10.

NOTE 3 For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use.

6.3.4 Attractiveness

The capability of the software product to be attractive to the user.

- 5 NOTE This refers to attributes of the software intended to make the software more attractive to the user, such as the use of colour and the nature of the graphical design.

6.3.5 Compliance

The capability of the software product to adhere to standards, conventions, style guides or regulations relating to usability.

6.4 Efficiency

The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions.

NOTE 1 Resources may include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, diskettes).

- 15 NOTE 2 For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use.

6.4.1 Time behaviour

The capability of the software product to provide appropriate response and processing times and throughput rates when performing its function, under stated conditions.

- 20 NOTE Human resources are included as part of productivity (7.1.2).

6.4.2 Resource utilisation

The capability of the software product to use appropriate amounts and types of resources when the software performs its function under stated conditions.

6.4.3 Compliance

- 25 The capability of the software product to adhere to standards or conventions relating to efficiency.

6.5 Maintainability

The capability of the software product to be modified. Modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications.

6.5.1 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.

6.5.2 Changeability

The capability of the software product to enable a specified modification to be implemented.

- 35 NOTE 1 Implementation includes coding, designing and documenting changes.

NOTE 2 If the software is to be modified by the end user, changeability may affect operability.

6.5.3 Stability

The capability of the software product to avoid unexpected effects from modifications of the software.

6.5.4 Testability

- 40 The capability of the software product to enable modified software to be validated.

6.5.5 Compliance

The capability of the software product to adhere to standards or conventions relating to maintainability.

6.6 Portability

The capability of the software product to be transferred from one environment to another.

- 45 NOTE The environment may include organisational, hardware or software environment.

6.6.1 Adaptability

The capability of the software product to be adapted for different specified environments without applying actions or means other than those provided for this purpose for the software considered.

- 50 NOTE 1 Adaptability includes the scalability of internal capacity (e.g. screen fields, tables, transaction volumes, report formats, etc.).

NOTE 2 If the software is to be adapted by the end user, adaptability corresponds to suitability for individualisation as defined in ISO 9241-10, and may affect operability.

6.6.2 Installability

The capability of the software product to be installed in a specified environment.

- 55 NOTE If the software is to be installed by an end user, installability can affect the resulting suitability and operability.

6.6.3 Co-existence

The capability of the software product to co-exist with other independent software in a common environment sharing common resources.

6.6.4 Replaceability

The capability of the software product to be used in place of another specified software product for the same purpose in the same environment.

NOTE 1 For example, the replaceability of a new version of a software product is important to the user when upgrading.

NOTE 2 Replaceability is used in place of **compatibility** in order to avoid possible ambiguity with interoperability (see 6.1.3).

NOTE 3 Replaceability may include attributes of both installability and adaptability. The concept has been introduced as a subcharacteristic of its own because of its importance.

6.6.5 Compliance

The capability of the software product to adhere to standards or conventions relating to portability.

7 Quality in use characteristics

The attributes of quality in use are categorised into four characteristics: effectiveness, productivity, safety and satisfaction (Figure 4).

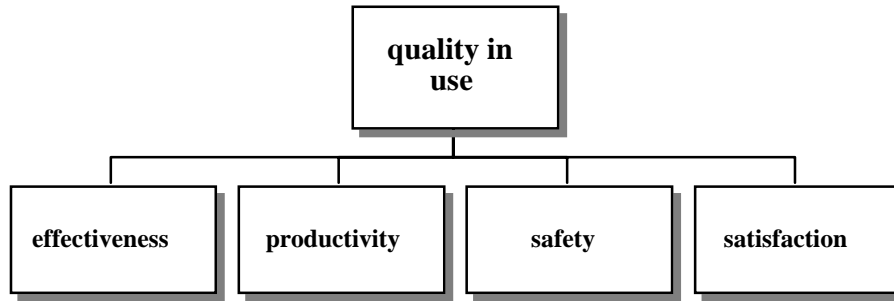


Figure 4 - Quality in use

Quality in use is the user's view of quality. Achieving quality in use is dependent on achieving the necessary external quality, which in turn is dependent on achieving the necessary internal quality (Figure 3). Measures are normally required at all three levels, as meeting criteria for internal measures is not usually sufficient to ensure achievement of criteria for external measures, and meeting criteria for external measures of subcharacteristics is not usually sufficient to ensure achieving criteria for quality in use. Examples of quality in use metrics are given in ISO/IEC 9126-4.

7.1 Quality in use

The capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use.

NOTE 1 Quality in use is the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself.

NOTE 2 Examples of metrics for quality in use are given in ISO/IEC 9126-4.

NOTE 3 The definition of quality in use in ISO/IEC 14598-1 (which is reproduced in Annex B) does not currently include the new characteristic of "safety".

NOTE 4 Usability is defined in ISO 9241-11 in a similar way to the definition of quality in use in this part of ISO/IEC 9126. Quality in use may be influenced by any of the quality characteristics, and is thus broader than usability, which is defined in this part of ISO/IEC 9126 in terms of understandability, learnability, operability, attractiveness and compliance.

7.1.1 Effectiveness

The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use.

7.1.2 Productivity

The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.

NOTE Relevant resources can include time, effort, materials or financial cost.

7.1.3 Safety

The capability of the software product to achieve acceptable levelsof risk of harm to people, software, equipment or the environment in a specified context of use.

NOTE 2 Risks to safety are usually a result of deficiencies in the functionality, reliability, usability or maintainability.

7.1.4 Satisfaction

The capability of the software product to satisfy users in a specified context of use.

NOTE Psychometrically-valid questionnaires can be used to obtain reliable measures of satisfaction.

Annex A (normative)

Metrics

5 A.1 Software metrics

A.1.1 Internal and external attributes

The levels of certain internal attributes have been found to influence the levels of some external attributes, so that there is both an external aspect and an internal aspect to most characteristics. For example, reliability may be measured externally by observing the number of failures in a given period of execution time during a trial of the software, and internally by inspecting the detailed specifications and source code to assess the level of fault tolerance. The internal attributes are said to be indicators of the external attributes. One internal attribute may influence one or more characteristics as well as one characteristic may be influenced by more than one attribute (Figure A.1). In this model the totality of software quality attributes are classified in a hierarchical tree structure of characteristics and subcharacteristics. The highest level of this structure consists of quality characteristics and the lowest level consists of software quality attributes. The hierarchy is not perfect, as some attributes may contribute to more than one subcharacteristic.

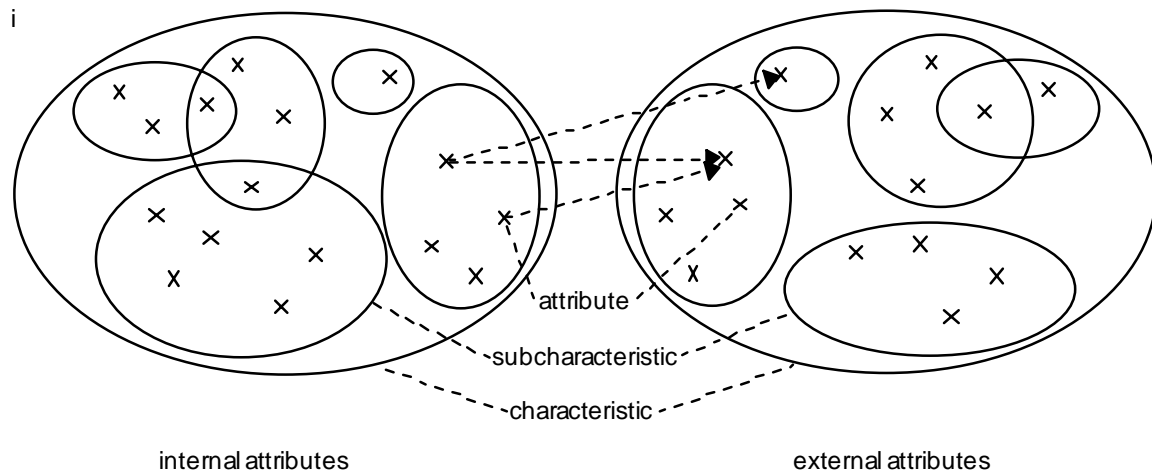


Figure A.1 - Quality characteristics, subcharacteristics and attributes

Subcharacteristics can either be measured by internal metrics or by external metrics. The correlation between internal attributes and external measures is never perfect, and the effect that a given internal attribute has upon an associated external measure will be determined by experience, and will depend on the particular context in which the software is used. In the same way, external properties (such as suitability, accuracy, fault tolerance or time behaviour) will influence the observed quality. A failure in quality in use (e.g. the user cannot complete the task) can be traced to external quality attributes (e.g. suitability or operability) and the associated internal attributes which have to be changed.

A.1.2 Internal metrics

Internal metrics can be applied to a non executable software product (such as a specification or source code) during designing and coding. When developing a software product the intermediate products should be evaluated using internal metrics which measure intrinsic properties, including those which can be derived from simulated behaviour. The primary purpose of these internal metrics is to ensure that the required external quality and quality in use is achieved: examples are given in ISO/IEC 9126-3. Internal metrics provide users, evaluators, testers, and developers with the benefit that they are able to evaluate software product quality and address quality issues early before the software product becomes executable.

Internal metrics measure internal attributes or indicate external attributes by analysis of the static properties of the intermediate or deliverable software products. The measurements of internal metrics use numbers or frequencies of software composition elements which appear for example on source code statements, the control graph, data flow and state transition representations.

5 NOTE Documentation can also be evaluated using internal metrics.

A.1.3 External metrics

10 External metrics use measures of a software product derived from measures of the behaviour of the system of which it is a part, by testing, operating and observing the executable software or system. Before acquiring or using a software product it should be evaluated using metrics based on business objectives related to the use, exploitation and management of the product in a specified organisational and technical environment. These are primarily external metrics: examples are given in ISO/IEC 9126-2. External metrics provide users, evaluators, testers, and developers with the benefit that they are able to evaluate software product quality during testing or operation.

A.1.4 Relationship between external and internal metrics

15 When the software quality requirements are defined, the software quality characteristics or subcharacteristics which contribute to the quality requirements are listed. Then, the appropriate external metrics and acceptable ranges are specified to quantify the quality criteria which validate that the software meets the user needs. The internal quality attributes of the software are then defined and specified to plan to finally achieve the required external quality and quality in use and to build them into the product during development. Appropriate internal metrics and acceptable ranges are specified to quantify the internal quality attributes so that they can be used for verifying that the intermediate software meets the internal quality specifications during the development.

20 It is recommended that the internal metrics are used which have as strong a relation as possible with the target external metrics, so that they can be used to predict the values of external metrics. However, it is generally difficult to design a rigorous theoretical model which provides a strong relationship between internal metrics and external metrics.

A.2 Quality in use metrics

30 Quality in use metrics measure the extent to which a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in a specified context of use. Evaluating quality in use validates software quality in specific user-task scenarios.

Quality in use is the user's view of the quality of a system containing software, and is measured in terms of the result of using the software, rather than properties of the software itself. Quality in use is the combined effect of the software quality characteristics for the user.

35 The relationship of quality in use to the other software quality characteristics depends on the type of user:

- the end user for whom quality in use is mainly a result of functionality, reliability, usability and efficiency;
- the person maintaining the software for whom quality in use is a result of maintainability;
- the person porting the software for whom quality in use is a result of portability.

A.3 Choice of metrics and measurement criteria

The basis on which the metrics are selected will depend on the business goals for the product and the needs of the evaluator. Needs are specified by criteria for measures. The model in this part of ISO/IEC 9126 supports a variety of evaluation requirements, for example:

- 45 • a user or a user's business unit could evaluate the suitability of a software product using metrics for quality in use;
- an acquirer could evaluate a software product against criterion values of external measures of functionality, reliability, usability and efficiency, or of quality in use;
- a maintainer could evaluate a software product using metrics for maintainability;
- 50 • a person responsible for implementing the software in different environments could evaluate a software product using metrics for portability;
- a developer could evaluate a software product against criterion values using internal measures of any of the quality characteristics.

NOTE ISO/IEC 14598-1 provides requirements and guidance for the choice of metrics and measurement criteria for software product evaluation.

A.4 Requirements for measurements used for comparison

Rigorous metrics are required to make reliable comparisons, either between products or with criterion values. Measurement procedures should measure the software product 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.

Metrics used for comparisons should be valid and accurate so that they can be used to make reliable comparisons. This means that measurements should be objective, empirical using an interval or better scale, and reproducible.

- To be objective, there shall be a written and agreed procedure for assigning the number or category to the attribute of the product.
- To be empirical, the data shall be obtained from observation or a psychometrically-valid questionnaire, and be measured on an interval, ratio or absolute scale.
- To be reproducible, the procedures for measurement shall result in the same measures (within appropriate tolerances) being obtained by different persons making the same measurement of the software product on different occasions.

When reporting the results of the use of quantitative metrics to make comparisons between products or with criterion values, the report shall state whether the metrics are objective, empirical using an interval or better scale, and reproducible.

Internal metrics should also have predictive validity, that is they should correlate with some desired external measures. For instance an internal measure of a particular software attribute should correlate with some measurable aspect of quality when the software is used. It is important that measurements assign values which coincide with normal expectations; for instance if the measurement suggests that the product is of high quality then this should be consistent with the product satisfying particular user needs.

Annex B (informative)

Definitions from other standards

5 Definitions are from ISO/IEC 14598-1: 1998 unless otherwise indicated.

B.1

acquirer

an organisation that acquires or procures a system, software product or software service from a supplier

10 [ISO/IEC 12207: 1995]

B.2

attribute

a measurable physical or abstract property of an entity

NOTE Attributes can be internal or external.

15

B.3

developer

an organisation that performs development activities (including requirements analysis, design, testing through acceptance) during the software lifecycle process

[ISO/IEC 12207: 1995]

20

B.4

direct measure

a measure of an attribute that does not depend upon a measure of any other attribute

B.5

evaluation module

25 a package of evaluation technology for a specific software quality characteristic or subcharacteristic

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

B.6

external measure

30 an indirect measure of a product derived from measures of the behaviour of the system of which it is a part

NOTE 1 The system includes any associated hardware, software (either custom software or off-the-shelf software) and users.

35 NOTE 2 The number of failures found during testing is an external measure of the number of faults in the program because the number of failures are counted during the operation of a computer system running the program.

NOTE 3 External measures can be used to evaluate quality attributes closer to the ultimate objectives of the design.

B.7

40

external quality

the extent to which a product satisfies stated and implied needs when used under specified conditions

B.8

failure

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

B.9

fault

an incorrect step, process or data definition in a computer program

NOTE This definition is taken from IEEE 610.12-1990.

50

B.10

implied needs

needs that may not have been stated but are actual needs when the entity is used in particular conditions

NOTE Implied needs are real needs which may not have been documented.

B.11**indicator**

a measure that can be used to estimate or predict another measure

NOTE 1 The predicted measure may be of the same or a different software quality characteristic.

- 5 NOTE 2 Indicators may be used both to estimate software quality attributes and to estimate attributes of the development process. They are imprecise indirect measures of the attributes.

B.12**indirect measure**

a measure of an attribute that is derived from measures of one or more other attributes

- 10 NOTE An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software.

B.13**intermediate software product**

- 15 a product of the software development process that is used as input to another stage of the software development process

NOTE In some cases an intermediate product may also be an end product.

B.14**internal measure**

- 20 a measure of the product itself, either direct or indirect

NOTE The number of lines of code, complexity measures, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself.

B.15**internal quality**

- 25 the totality of attributes of a product that determine its ability to satisfy stated and implied needs when used under specified conditions

NOTE 1 The term "internal quality", used in ISO/IEC 14598 to contrast with "external quality", has essentially the same meaning as "quality" in ISO 8402.

- 30 NOTE 2 The term "attribute" is used with the same meaning as the term "characteristic" used in 4.1.1, as the term "characteristic" is used in a more specific sense in ISO/IEC 9126.

B.16**maintainer**

an organisation that performs maintenance activities

[ISO/IEC 12207: 1995]

- 35 **B.17**

measure (verb)

make a measurement

B.18**measure (noun)**

- 40 the number or category assigned to an attribute of an entity by making a measurement

B.19**measurement**

the use of a metric to assign a value (which may be a number or category) from a scale to an attribute of an entity

- 45 NOTE Measurement can be qualitative when using categories. For example, some important attributes of software products, e.g. the language of a source program (ADA, C, COBOL, etc.) are qualitative categories.

B.20**metric**

the defined measurement method and the measurement scale

- 50 NOTE 1 Metrics can be internal or external, and direct or indirect
NOTE 2 Metrics include methods for categorising qualitative data.

B.21**quality**

the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs.

- 55 NOTE 1 In a contractual environment, or in a regulated environment, such as the nuclear safety field, needs are specified, whereas in other environments, implied needs should be identified and defined (ISO 8402 : 1994, note 1).

NOTE 2 In ISO/IEC 14598 the relevant entity is a software product.

[ISO 8402: 1994]

B.22**quality evaluation**

systematic examination of the extent to which an entity is capable of fulfilling specified requirements

- 5 **NOTE** The requirements may be formally specified, as when a product is developed for a specific user under a contract, or specified by the development organisation, as when a product is developed for unspecified users, such as consumer software, or the requirements may be more general, as when a user evaluates products for comparison and selection purpose.

[ISO 8402: 1994]

B.2310 **quality in use**

the extent to which a product used by specified users meets their needs to achieve specified goals with effectiveness, productivity and satisfaction in specified contexts of use

NOTE This definition of quality in use is similar to the definition of usability in ISO 9241-11. In ISO/IEC 14598 the term usability is used to refer to the software quality characteristic described in ISO/IEC 9126-1.

15 **B.24****quality model**

the set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality

B.2520 **rating**

the action of mapping the measured value to the appropriate rating level. Used to determine the rating level associated with the software for a specific quality characteristic

B.26**rating level**

25 a scale point on an ordinal scale which is used to categorise a measurement scale

NOTE 1 The rating level enables software to be classified (rated) in accordance with the stated or implied needs (see 10.2).

NOTE 2 Appropriate rating levels may be associated with the different views of quality i.e. Users', Managers' or 'Developers'.

30 **B.27****scale**

a set of values with defined properties

- 35 **NOTE** Examples of 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 point but also possess an absolute zero. Metrics using nominal or ordinal scales produce qualitative data, and metrics using interval and ratio scales produce quantitative data.

B.28**software**

40 all or part of the programs, procedures, rules, and associated documentation of an information processing system

NOTE Software is an intellectual creation that is independent of the medium on which it is recorded.

[ISO/IEC 2382-1: 1993]

B.2945 **software product**

the set of computer programs, procedures, and possibly associated documentation and data

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

[ISO/IEC 12207: 1995]

50 **B.30****supplier**

an organisation 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

[ISO/IEC 12207: 1995]

55 **B.31****system**

an integrated composite that consists of one or more of the processes, hardware, software, facilities and people, that provides a capability to satisfy a stated need or objective

[ISO/IEC 12207: 1995]

B.32**user**

an individual that uses the software product to perform a specific function

- 5 NOTE Users may include operators, recipients of the results of the software, or developers or maintainers of software.

B.33**validation**

confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled

- 10 NOTE 1 In design and development, validation concerns the process of examining a product to determine conformity with user needs.

NOTE 2 Validation is normally performed on the final product under defined operating conditions. It may be necessary in earlier stages.

NOTE 3 "Validated" is used to designate the corresponding status.

- 15 NOTE 4 Multiple validations may be carried out if there are different intended uses.

[ISO 8402: 1994]

B.34**verification**

confirmation by examination and provision of objective evidence that specified requirements have been fulfilled

- 20 NOTE 1 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.

NOTE 2 "Verified" is used to designate the corresponding status.

[ISO 8402:1994]

Annex C (informative)

History of the work

5 C.1 Background

The software industry is entering a period of some maturing, while at the same time software is becoming a crucial component of many of today's products. This pervasive aspect of software makes it a major new factor in trade. Furthermore, with new global demands for safety and quality, the need for international agreements on software quality assessment procedures is becoming important.

10 There are essentially two approaches that can be followed to ensure product quality, one being assurance of the process by which the product is developed, and the other being the evaluation of the quality of the end product. Both avenues are important and both require the presence of a system for managing quality. Such a system identifies the management commitment to quality, and states its policies, as well as the detailed steps that must be in place.

15 To evaluate the quality of a product through some quantitative means, a set of quality characteristics that describe the product and form the basis for the evaluation is required. This part of ISO/IEC 9126 defines these quality characteristics for software products.

C.2 History

20 State of the art in software technology does not yet present a well established and widely accepted description scheme for assessing the quality of software product. Much work has been done since about 1976 by a number of individuals to define a software quality framework. Models by McCall, Boehm, the US Air Force, and others have been adopted and enhanced over the years. However, today it is difficult for a user or consumer of software products to understand or compare the quality of software.

25 For a long time, reliability has been the only way to gauge quality. Other quality models have been proposed and submitted for use. While studies were useful, they also caused confusion because of the many quality aspects offered. Thus, the need for one standard model came about.

It is for this reason that the ISO/IEC JTC1 began to develop the required consensus and encourage standardisation world-wide.

30 First considerations originated in 1978, and in 1985 the development of ISO/IEC 9126 was started. The models proposed initially introduced properties of software that depend on application or implementation aspects (or both), to describe the quality of software.

35 The first step of the ISO technical committee to arrange these properties systematically failed for lack of definitions. Terms were interpreted in different ways by experts. All structures discussed were, therefore, of an arbitrary nature, without a common basis.

As a result it was decided that the best chance for establishing an International Standard was to stipulate a set of characteristics based on a definition of quality that was subsequently used in ISO 8402. This definition is accepted for all kinds of products and services. It starts with the user's needs.

C.3 Six ISO software quality characteristics

40 The requirements for choosing the characteristics described in ISO/IEC 9126 were as follows:

- To cover together all aspects of software quality resulting from the ISO quality definition.
- To describe the product quality with a minimum of overlap.
- To be as close as possible to the established terminology.
- To form a set of not more than six to eight characteristics for reason of clarity and handling.
- 45 • To identify areas of attributes of software products for further refinement.

The work of the technical committee resulted in the above set of characteristics.

However, a pure terminology standard, containing definitions of characteristics would not have provided sufficient support to users in assessing software quality. Therefore, a description on how to proceed with evaluating the quality of a software product was included.

- 5 Evaluating product quality in practice requires characteristics beyond the set at hand, and requires metrics for each of the characteristics. The state of art at present did not permit standardisation in this area. Waiting for enhancements would have delayed the publication of ISO/IEC 9126 substantially. For this reason, the technical committee issued the 1991 version of ISO/IEC 9126 to harmonise further development.

C.4 Revision of ISO/IEC 9126

- 10 In 1994 it was felt that other Standards being produced in the area of product quality evaluation necessitated the revision of ISO/IEC 9126. The revision retains the same 6 quality characteristics, but clarifies their relationship to internal and external metrics. The relationship between the characteristics and quality in use is also explained.
- 15 Quality is defined in ISO 8402 in terms of 'Totality of characteristics of an entity that bear on ...'. NOTE 4 in this definition states that 'The term quality should not be used as a single term to express a degree of excellence in a comparative sense'. For this reason the terms 'internal quality' and 'external quality' have been defined in ISO/IEC 14598-1 to refer to aspects of quality which can be measured. The wording of the definitions of the quality characteristics has been changed from: "A set of attributes that bear on" to: "The capability of the software to ..." so they can be interpreted in terms which enable both
- 20 internal and external quality to be measured.
- Subcharacteristics have been introduced, based on those in the informative annex of the previous version of ISO/IEC 9126. Compliance was made a subcharacteristic of all characteristics, as the principles are generally applicable to all the software characteristics.
- 25 The evaluation process model has been moved to ISO/IEC 14598-1. Two new technical reports are being prepared as parts 2 and 3 of ISO/IEC 9126, giving examples of external and internal metrics.

Annex D (informative)

Bibliography

- 5 [1] IEC 50(191) *International Electrotechnical vocabulary - Dependability and quality of service*
 [2] IEEE 610.12-1990 Standard Glossary of Software Engineering Terminology
 [3] ISO 8402: 1994, Quality -Vocabulary .
 [4] ISO 9001:1994, Model for quality assurance in design, development, production, installation and servicing;
- 10 [5] ISO/IEC PDTR 9126-2:new, Information Technology - Software product quality - Part 1: External metrics
 [6] ISO/IEC PDTR 9126-3:new, Information Technology - Software product quality - Part 1: Internal metrics
 [7] ISO/IEC PDTR 9126-4:new, Information Technology - Software product quality - Part 1: Quality in use metrics
- 15 [8] ISO 9241-10:1996, Ergonomic requirements for office work with visual display terminals (VDT)s - Part 10: Dialogue principles
 [9] ISO DIS 9241-11:1997, Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11: Guidance on usability.
- 20 [10] ISO/IEC 12207:1995, Information Technology - Software lifecycle processes.
 [11] ISO/IEC PDTR 15504:1996, Information Technology - Software Process Assessment
 [12] ISO/IEC 14598-2:new, Information Technology - Software product evaluation - Part 2: Planning and management
- 25 [13] ISO/IEC 14598-3:new, Information Technology - Software product evaluation - Part 3: Process for developers
 [14] ISO/IEC 14598-4:new, Information Technology - Software product evaluation - Part 4: Process for acquirers
 [15] ISO/IEC 14598-5:new, Information Technology - Software product evaluation - Part 5: Process for evaluators
- 30 [16] ISO/IEC 14598-6:new, Information Technology - Software product evaluation - Part 6: Documentation of evaluation modules
 [17] ISO/IEC 2382-14: *Reliability, maintainability and availability*
 [18] ISO/IEC 2382-1:1993 Data processing - Vocabulary - Part 1: Fundamental terms
 [19] ISO/IEC 2382-20 :1990, Information technology -- Vocabulary - Part 20 : Systems development.