TECHNICAL REPORT

ISO/IEC TR 9126-2

First edition
2003-07-01

Software engineering — Product quality —

Part 2: External metrics

Génie du logiciel — Qualité des produits —

Partie 2: Métrologie externe

Reference number
ISO/IEC TR 9126-2:2003(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 Electrotechnical 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.

In exceptional circumstances, the joint technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when the joint technical committee has collected data of a different kind from that which is normally published as an International Standard (“state of the art”, for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

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 TR 9126-2:2003, which is a Technical Report of type 2, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and system engineering*.

This document is being issued in the Technical Report (type 2) series of publications (according to the Procedures for the technical work of ISO/IEC JTC 1) as a “prospective standard for provisional application” in the field of external metrics for quantitatively measuring external software because there is an urgent need for guidance on how standards in this field should be used to meet an identified need.

This document is not to be regarded as an “International Standard”. It is proposed for provisional application so that information and experience of its use in practice may be gathered. Comments on the content of this document should be sent to the ISO Central Secretariat.

A review of this Technical Report (type 2) will be carried out not later than three years after its publication with the options of: extension for another three years; conversion into an International Standard; or withdrawal.

ISO/IEC 9126 consists of the following parts, under the general title *Software engineering — Product quality*:

— *Part 1: Quality model*

— *Part 2: External metrics*

— *Part 3: Internal metrics*

— *Part 4: Quality in use metrics*

Introduction

This Technical Report provides external metrics for measuring attributes of six external quality characteristics defined in ISO/IEC 9126-1. The metrics listed in this Technical Report are not intended to be an exhaustive set. Developers, evaluators, quality managers and acquirers may select metrics from this Technical Report for defining requirements, evaluating software products, measuring quality aspects and other purposes. They may also modify the metrics or use metrics which are not included here. This Technical Report is applicable to any kind of software product, although each of the metrics is not always applicable to every kind of software product.

ISO/IEC 9126-1 defines terms for the software quality characteristics and how these characteristics are decomposed into subcharacteristics. ISO/IEC 9126-1, however, does not describe how any of these subcharacteristics could be measured. ISO/IEC TR 9126-2 defines external metrics, ISO/IEC TR 9126-3 defines internal metrics and ISO/IEC 9126-4 defines quality in use metrics, for measurement of the characteristics or the subcharacteristics. Internal metrics measure the software itself, external metrics measure the behaviour of the computer-based system that includes the software, and quality in use metrics measure the effects of using the software in a specific context of use.

This Technical Report is intended to be used together with ISO/IEC 9126-1. It is strongly recommended to read ISO/IEC 14598-1 and ISO/IEC 9126-1, prior to using this Technical Report, particularly if the reader is not familiar with the use of software metrics for product specification and evaluation.

Clauses 1 to 7 and Annexes A to D are common to ISO/IEC TR 9126-2, ISO/IEC TR 9126-3, and ISO/IEC 9126-4.

Software engineering — Product quality —

Part 2: External metrics

1 Scope

This Technical Report defines external metrics for quantitatively measuring external software quality in terms of characteristics and subcharacteristics defined in ISO/IEC 9126-1, and is intended to be used together with ISO/IEC 9126-1.

This Technical Report contains:

I. an explanation of how to apply software quality metrics

II. a basic set of metrics for each subcharacteristic

III. an example of how to apply metrics during the software product life cycle

This Technical Report does not assign ranges of values of these metrics to rated levels or to grades of compliance, because these values are defined for each software product or a part of the software product, by its nature, depending on such factors as category of the software, integrity level and users' needs. Some attributes may have a desirable range of values, which does not depend on specific user needs but depends on generic factors; for example, human cognitive factors.

This Technical Report can be applied to any kind of software for any application. Users of this Technical Report can select or modify and apply metrics and measures from this Technical Report or may define application-specific metrics for their individual application domain. For example, the specific measurement of quality characteristics such as safety or security may be found in International Standards or Technical Reports provided by IEC 65 and ISO/IEC JTC 1/SC 27.

Intended users of this Technical Report include:

— Acquirer (an individual or organization that acquires or procures a system, software product or software service from a supplier);

— Evaluator (an individual or organization that performs an evaluation. An evaluator may, for example, be a testing laboratory, the quality department of a software development organization, a government organization or a user);

— Developer (an individual or organization that performs development activities, including requirements analysis, design, and testing through acceptance during the software life cycle process);

— Maintainer (an individual or organization that performs maintenance activities);

— Supplier (an 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) when validating software quality at qualification test;

— User (an individual or organization that uses the software product to perform a specific function) when evaluating quality of software product at acceptance test;

— Quality manager (an individual or organization that performs a systematic examination of the software product or software services) when evaluating software quality as part of quality assurance and quality control.

2 Conformance

There are no conformance requirements in this Technical Report.

NOTE General conformance requirements for metrics are in ISO/IEC 9126-1 Quality model.

3 Normative references

ISO/IEC 9126-1:2001, *Software engineering — Product quality — Part 1: Quality model*

ISO/IEC TR 9126-3[1)], *Software engineering — Product quality — Part 3: Internal metrics*

ISO/IEC 9126-4[1)], *Software engineering — Product quality — Part 4: Quality in use metrics*

ISO/IEC 14598-1:1999, *Information technology — Software product evaluation — Part 1: General overview*

ISO/IEC 14598-2:2000, *Software engineering — Product evaluation — Part 2: Planning and management*

ISO/IEC 14598-3:2000, *Software engineering — Product evaluation — Part 3: Process for developers*

ISO/IEC 14598-4:1999, *Software engineering — Product evaluation — Part 4: Process for acquirers*

ISO/IEC 14598-5:1998, *Information technology — Software product evaluation — Part 5: Process for evaluators*

ISO/IEC 14598-6:2001, *Software engineering — Product evaluation — Part 6: Documentation of evaluation modules*

ISO/IEC 12207:1995, *Information technology — Software life cycle processes*

ISO/IEC 14143-1:1998, *Information technology — Software measurement — Functional size measurement — Part 1: Definition of concepts*

ISO/IEC 2382-20:1990, *Information technology — Vocabulary — Part 20: System development*

ISO 9241-10:1996, *Ergonomic requirements for office work with visual display terminals (VDTs) — Part 10: Dialogue principles*

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14598-1:1999 and ISO/IEC 9126-1:2001 apply. They are also listed in Annex D.

5 Abbreviated terms

The following abbreviations are used in this Technical Report:

SQA — Software Quality Assurance (Group)

SLCP — Software Life Cycle Processes

1) To be published.

6 Use of software quality metrics

These Technical Reports (ISO/IEC TR 9126-2 External metrics, ISO/IEC TR 9126-3 Internal metrics and ISO/IEC 9126-4 Quality in use metrics) provide a suggested set of software quality metrics (external, internal and quality in use metrics) to be used with the ISO/IEC 9126-1 Quality model. The user of these Technical Reports may modify the metrics defined, and/or may also use metrics not listed. When using a modified or a new metric not identified in these Technical Reports, the user should specify how the metrics relate to the ISO/IEC 9126-1 quality model or any other substitute quality model that is being used.

The user of these Technical Reports should select the quality characteristics and subcharacteristics to be evaluated, from ISO/IEC 9126-1; identify the appropriate direct and indirect measures, identify the relevant metrics and then interpret the measurement result in an objective manner. The user of these Technical Reports also may select product quality evaluation processes during the software life cycle from the ISO/IEC 14598 series of standards. These give methods for measurement, assessment and evaluation of software product quality. They are intended for use by developers, acquirers and independent evaluators, particularly those responsible for software product evaluation (see Figure 1).

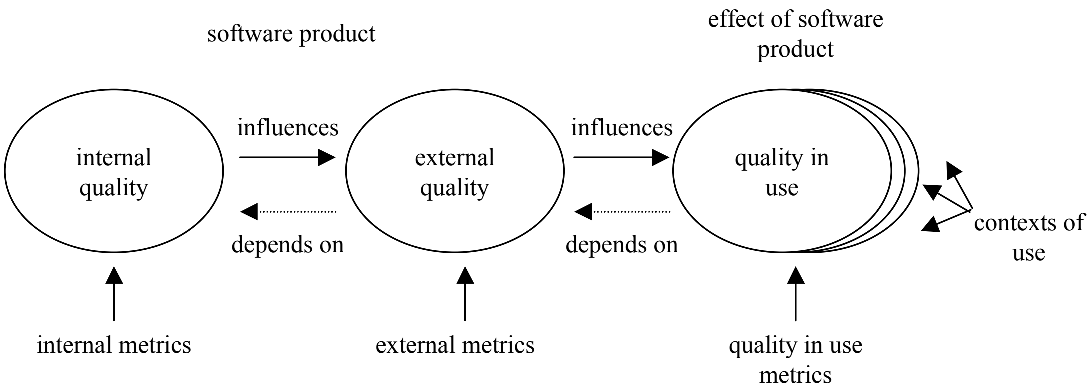

Figure 1 – Relationship between types of metrics

The internal metrics may be applied to a non-executable software product during its development stages (such as request for proposal, requirements definition, design specification or source code). Internal metrics provide the users with the ability to measure the quality of the intermediate deliverables and thereby predict the quality of the final product. This allows the user to identify quality issues and initiate corrective action as early as possible in the development life cycle.

The external metrics may be used to measure the quality of the software product by measuring the behaviour of the system of which it is a part. The external metrics can only be used during the testing stages of the life cycle process and during any operational stages. The measurement is performed when executing the software product in the system environment in which it is intended to operate.

The quality in use metrics measure whether a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in a specified context of use. This can be only achieved in a realistic system environment.

User quality needs can be specified as quality requirements by quality in use metrics, by external metrics, and sometimes by internal metrics. These requirements specified by metrics should be used as criteria when a product is evaluated.

It is recommended to use internal metrics having a relationship as strong as possible with the target external metrics so that they can be used to predict the values of external metrics. However, it is often difficult to design a rigorous theoretical model that provides a strong relationship between internal metrics and external metrics. Therefore, a hypothetical model that may contain ambiguity may be designed and the extent of the relationship may be modelled statistically during the use of metrics.

Recommendations and requirements related to validity and reliability are given in ISO/IEC 9126-1, Clause A.4. Additional detailed considerations when using metrics are given in Annex A of this Technical Report.

7 How to read and use the metrics tables

The metrics listed in Clause 8 are categorized by the characteristics and subcharacteristics in ISO/IEC 9126-1. The following information is given for each metric in the table:

a) **Metric name:** Corresponding metrics in the internal metrics table and external metrics table have similar names.

b) **Purpose of the metric:** This is expressed as the question to be answered by the application of the metric.

c) **Method of application:** Provides an outline of the application.

d) **Measurement, formula and data element computations:** Provides the measurement formula and explains the meanings of the used data elements.

NOTE In some situations more than one formula is proposed for a metric.

e) **Interpretation of measured value:** Provides the range and preferred values.

f) **Metric scale type:** Type of scale used by the metric. Scale types used are; Nominal scale, Ordinal scale, Interval scale, Ratio scale and Absolute scale.

NOTE A more detailed explanation is given in Annex C.

g) **Measure type:** Types used are; Size type (e.g. Function size, Source size), Time type (e.g. Elapsed time, User time), Count type (e.g. Number of changes, Number of failures).

NOTE A more detailed explanation is given in Annex C.

h) **Input to measurement:** Source of data used in the measurement.

i) **ISO/IEC 12207 SLCP Reference:** Identifies software life cycle process(es) where the metric is applicable.

j) **Target audience:** Identifies the user(s) of the measurement results.

8 Metrics tables

The metrics listed in this clause are not intended to be an exhaustive set and may not have been validated. They are listed by software quality characteristics and subcharacteristics, in the order introduced in ISO/IEC 9126-1.

Metrics, which may be applicable, are not limited to these listed here. Additional specific metrics for particular purposes are provided in other related documents, such as functional size measurement or precise time efficiency measurement.

NOTE 1 It is recommended to refer a specific metric or measurement form from specific standards, technical reports or guidelines. Functional size measurement is defined in ISO/IEC 14143. An example of precise time efficiency measurement can be referred from ISO/IEC 14756.

Metrics should be validated before application in a specific environment (see Annex A).

NOTE 2 This list of metrics is not finalized, and may be revised in future versions of this Technical Report. Readers of this Technical Report are invited to provide feedback.

8.1 Functionality metrics

An external functionality metric should be able to measure an attribute such as the functional behaviour of a system containing the software. The behaviour of the system may be observed from the following perspectives:

a) Differences between the actual executed results and the quality requirements specification;

NOTE 1 The quality requirements specification for functionality is usually described as the functional requirements specification.

b) Functional inadequacy detected during real user operation which is not stated but is implied as a requirement in the specification.

NOTE 2 When implied operations or functions are detected, they should be reviewed, approved and stated in the specifications. Their extent to be fulfilled should be agreed.

8.1.1 Suitability metrics

An external suitability metric should be able to measure an attribute such as the occurrence of an unsatisfying function or the occurrence of an unsatisfying operation during testing and user operation of the system.

An unsatisfying function or operation may be:

a) Functions and operations that do not perform as specified in user manuals or requirement specification.

b) Functions and operations that do not provide a reasonable and acceptable outcome to achieve the intended specific objective of the user task.

8.1.2 Accuracy metrics

An external accuracy metric should be able to measure an attribute such as the frequency of users encountering the occurrence of inaccurate matters which includes:

a) Incorrect or imprecise result caused by inadequate data; for example, data with too few significant digits for accurate calculation;

b) Inconsistency between actual operation procedures and described ones in the operation manual;

c) Differences between the actual and reasonable expected results of tasks performed during operation.

8.1.3 Interoperability metrics

An external interoperability metric should be able to measure an attribute such as the number of functions or occurrences of less communicativeness involving data and commands, which are transferred easily between the software product and other systems, other software products, or equipment which are connected.

8.1.4 Security metrics

An external security metric should be able to measure an attribute such as the number of functions with, or occurrences of security problems, which are:

a) Failing to prevent leak of secure output information or data;

b) Failing to prevent loss of important data;

c) Failing to defend against illegal access or illegal operation.

NOTE 1 It is recommended that penetration tests be performed to simulate attack, because such a security attack does not normally occur in the usual testing. Real security metrics may only be taken in “real life system environment”, that is “quality in use”.

NOTE 2 Security protection requirements vary widely from the case of a stand-alone-system to the case of a system connected to the Internet. The determination of the required functionality and the assurance of their effectiveness have been addressed extensively in related standards. The user of this standard should determine security functions using appropriate methods and standards in those cases where the impact of any damage caused is important or critical. In the other case the user may limit his scope to generally accepted “Information Technology (IT)” protection measures such as virus protection backup methods and access control.

8.1.5 Functionality compliance metrics

An external functionality compliance metric should be able to measure an attribute such as the number of functions with, or occurrences of compliance problems, which are the software product failing to adhere to standards, conventions, contracts or other regulatory requirements.

Table 8.1.1 Suitability metrics

External suitability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Functional adequacy	How adequate are the evaluated functions?	Number of functions that are suitable for performing the specified tasks comparing to the number of function evaluated.	X=1-A/B A= Number of functions in which problems are detected in evaluation B= Number of functions evaluated	0 <= X <= 1 The closer to 1.0, the more adequate.	Absolute	X= Count/ Count A= Count B= Count	Requirement specification (Req. Spec.) Evaluation report	6.5 Validation, 6.3 Quality Assurance, 5.3 Qualification testing	Developer, SQA
Functional implementation completeness	How complete is the implementation according to requirement specifications?	Do functional tests (black box test) of the system according to the requirement specifications. Count the number of missing functions detected in evaluation and compare with the number of function described in the requirement specifications.	X = 1 - A / B A = Number of missing functions detected in evaluation B = Number of functions described in requirement specifications	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Req. spec. Evaluation report	6.5 Validation, 6.3 Quality Assurance, 5.3 Qualification testing	Developer, SQA

FOOTNOTES

1 *Input to the measurement process is the updated requirement specification. Any changes identified during life cycle must be applied to the requirement specifications before using in measurement process.*

2 *This metric is suggested as experimental use.*

NOTE Any missing function cannot be examined by testing because it is not implemented. For detecting missing functions, it is suggested that each function stated in a requirement specification be tested one by one during functional testing. Such results become input to “Functional implementation completeness” metric. For detecting functions which are implemented but inadequate, it is suggested that each function be tested for multiple specified tasks. Such results become input to the “Functional adequacy” metric. Therefore, users of metrics are suggested to use both these metrics during functional testing.

Table 8.1.1 (*continued*)

External suitability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Functional implementation coverage	How correct is the functional implementation?	Do functional tests (black box test) of the system according to the requirement specifications. Count the number of incorrectly implemented or missing functions detected in evaluation and compare with the total number of functions described in the requirement specifications Count the number of functions that are complete versus the ones that are not.	X=1- A / B A= Number of incorrectly implemented or missing functions detected in evaluation B= Number of functions described in requirement specifications	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Req. spec. Evaluation report	6.5 Validation, 6.3 Quality Assurance, 5.3 Qualification testing	Developer, SQA

FOOTNOTES

1 *Input to the measurement process is the updated requirement specification. Any changes identified during life cycle must be applied to the requirement specifications before using in measurement process.*

2 *This measure represents a binary gate checking of determining the presence of a feature.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Functional specification stability (volatility)	How stable is the functional specification after entering operation?	Count the number of functions described in functional specifications that had to be changed after the system is put into operation and compare with the total number of functions described in the requirement specifications.	X = 1- A / B A= Number of functions changed after entering operation starting from entering operation B= Number of functions described in requirement specifications	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Req. spec. Evaluation report	6.8 Problem Resolution 5.4 Operation	Maintainer SQA

FOOTNOTE

This metric is suggested as experimental use.

Table 8.1.2 Accuracy metrics

External accuracy metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Accuracy to expectation	Are differences between the actual and reasonable expected results acceptable?	Do input .vs. output test cases and compare the output to reasonable expected results. Count the number of cases encountered by the users with an unacceptable difference from reasonable expected results.	X=A / T A= Number of cases encountered by the users with a difference against to reasonable expected results beyond allowable T= Operation time	0<=X The closer to 0 is the better.	Ratio	A= Count T= Time X= Count/ Time	Req. spec. User operation manual Hearing to users Test report	6.5 Validation 6.3 Quality Assurance	Developer User
FOOTNOTE *Reasonable expected results might be identified in a requirement specification, a user operation manual, or users' expectations.*									
Computational Accuracy	How often do the end users encounter inaccurate results?	Record the number of inaccurate computations based on specifications.	X=A / T A= Number of inaccurate computations encountered by users T= Operation time	0<=X The closer to 0 is the better.	Ratio	A= Count T= Time X= Count/ Time	Req. spec. Test report	6.5 Validation 6.3 Quality Assurance	Developer User
Precision	How often do the end users encounter results with inadequate precision ?	Record the number of results with inadequate precision.	X=A / T A= Number of results encountered by the users with level of precision different from required T= Operation time	0<=X The closer to 0 is the better.	Ratio	A= Count T= Time X= Count/ Time	Req. spec. Test report	6.5 Validation 6.3 Quality Assurance	Developer User

NOTE Data elements for computation of external metrics are designed to use externally accessible information, because it is helpful for end users, operators, maintainers or acquirers to use external metrics. Therefore, the time basis metric often appears in external metrics and is different from internal ones.

Table 8.1.3 Interoperability metrics

External interoperability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Data exchangeability (Data format based)	How correctly have the exchange interface functions for specified data transfer been implemented?	Test each downstream interface function output record format of the system according to the data fields specifications. Count the number of data formats that are approved to be exchanged with other software or system during testing on data exchanges in comparing with the total number.	X= A / B A= Number of data formats which are approved to be exchanged successfully with other software or system during testing on data exchanges B= Total number of data formats to be exchanged	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Req. spec. (User manual) Test report	6.5 Validation	Developer
FOOTNOTE *It is recommended to test specified data transaction.*									
Data exchangeability (User's success attempt based)	How often does the end user fail to exchange data between target software and other software? How often are the data transfers between target software and other software successful? Can user usually succeed in exchanging data?	Count the number of cases that interface functions were used and failed.	a) X= 1 - A / B A= Number of cases in which user failed to exchange data with other software or systems B= Number of cases in which user attempted to exchange data b) Y= A / T T= Period of operation time	0<=X<= 1 The closer to 1.0 is the better. 0<=Y The closer to 0, is the better.	a) Absolute b) Ratio	A= Count B= Count X= Count/ Count Y= Count/ Time T= Time	Req. spec. (User manual) Test report	5.4 Operation	Maintainer

Table 8.1.4 Security metrics

External security metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Access auditability	How complete is the audit trail concerning the user access to the system and data?	Evaluate the amount of accesses that the system recorded in the access history database.	X= A / B A= Number of “user accesses to the system and data” recorded in the access history database B= Number of “ user accesses to the system and data” done during evaluation	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Test spec. Test report	6.5 Validation	Developer

FOOTNOTES

1 *Accesses to data may be measured only with testing activities.*

2 *This metric is suggested as an experimental use.*

3 *It is recommended that penetration tests be performed to simulate attacks, because such security attacks do not normally occur in the usual testing. Real security metrics may only be taken in "real life system environment", that is "quality in use".*

4 *“User access to the system and data” record may include “virus detection record“ for virus protection. The aim of the concept of computer virus protection is to create suitable safeguards with which the occurrence of computer viruses in systems can be prevented or detected as early as possible.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Access controllability	How controllable is access to the system?	Count number of detected illegal operations with comparing to number of illegal operations as in the specification.	X= A / B A= Number of detected different types of illegal operations B= Number of types of illegal operations as in the specification	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Test spec. Test report Operation report	6.5 Validation 6.3 Quality Assurance	Developer

FOOTNOTES

1 *If it is necessary to complement detection of unexpected illegal operations additional intensive abnormal operation testing should be conducted.*

2 *It is recommended that penetration tests be performed to simulate attack, because such security attacks do not normally occur in the usual testing. Real security metrics may only be taken in "real life system environment", that is "quality in use".*

3 *Functions prevent unauthorized persons from creating, deleting or modifying programs or information. Therefore, it is suggested to include such illegal operation types in test cases.*

Table 8.1.4 (*continued*)

External security metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Data corruption prevention	What is the frequency of data corruption events?	Count the occurrences of major and minor data corruption events.	a) X= 1 – A / N A= Number of times that a major data corruption event occurred N= Number of test cases tried to cause data corruption event b) Y= 1- B / N B= Number of times that a minor data corruption event occurred c) Z= A / T or B / T T= period of operation time (during operation testing)	0<=X<= 1 The closer to 1.0 is the better. 0<=Y<= 1 The closer to 1.0 is the better. 0<=Z The closer to 0, is the better.	a) Absolute b) Absolute c) Ratio	A= Count B= Count N= Count X= Count/ Count Y= Count/ Count T= Time Z= Count/ Time	Test spec. Test report Operation report	6.5 Validation 5.3 Qualification testing 5.4 Operation	Maintainer Developer

FOOTNOTES

1 *Intensive abnormal operation testing is needed to obtain minor and major data corruption events.*

2 *It is recommended to grade the impact of data corruption events such as the following examples:*
Major (fatal) data corruption event:
- reproduction and recovery impossible;
- second affection distribution too wide;
- importance of data itself.
Minor data corruption event:
- reproduction or recovery possible and
- no second affection distribution;
- importance of data itself.

3 *Data elements for computation of external metrics are designed to use externally accessible information, because it is helpful for end users, operators, maintainers or acquirers to use external metrics. Therefore, counting events and times used here are different from corresponding internal metric.*

4 *It is recommended that penetration tests be performed to simulate attack, because such security attacks do not normally occur in the usual testing.*
Real security metrics may only be taken in "real life system environment", that is "quality in use"

5 *This metric is suggested as an experimental use.*

6 *Data backup is one of the effective ways to prevent data corruption. The creation of back up ensures that necessary data can be restored quickly in the event that parts of the operative data are lost. However, data back up is regarded as a part of the composition of the reliability metrics in this report.*

7 *It is suggested that this metric be used experimentally.*

Table 8.1.5 Functionality compliance metrics

External functionality compliance metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Functional compliance	How compliant is the functionality of the product to applicable regulations, standards and conventions?	Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance in the specification. Design test cases in accordance with compliance items. Conduct functional testing for these test cases. Count the number of compliance items that have been satisfied.	X = 1 - A / B A= Number of functionality compliance items specified that have not been implemented during testing B= Total number of functionality compliance items specified	0<= X <=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTES

1 *It may be useful to collect several measured values along time, to analyse the trend of increasingly satisfied compliance items and to determine whether they are fully satisfied or not.*

2 *It is suggested to count number of failures, because problem detection is an objective of effective testing and also suitable for counting and recording.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Interface standard compliance	How compliant are the interfaces to applicable regulations, standards and conventions?	Count the number of interfaces that meet required compliance and compare with the number of interfaces requiring compliance as in the specifications.	X= A / B A= Number of correctly implemented interfaces as specified B= Total number of interfaces requiring compliance	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description of compliance and related standards, conventions or regulations Test specification and report	6.5 Validation	Developer

FOOTNOTE

All specified attributes of a standard must be tested.

8.2 Reliability metrics

An external reliability metric should be able to measure attributes related to the behaviours of the system of which the software is a part during execution testing to indicate the extent of reliability of the software in that system during operation. Systems and software are not distinguished from each other in most cases.

8.2.1 Maturity metrics

An external maturity metric should be able to measure such attributes as the software freedom of failures caused by faults existing in the software itself.

8.2.2 Fault tolerance metrics

An external fault tolerance metric should be related to the software capability of maintaining a specified performance level in cases of operation faults or infringement of its specified interface.

8.2.3 Recoverability metrics

An external recoverability metric should be able to measure such attributes as the software with system being able to re-establish its adequate level of performance and recover the data directly affected in the case of a failure.

8.2.4 Reliability compliance metrics

An external reliability compliance metric should be able to measure an attribute such as the number of functions with, or occurrences of compliance problems, in which the software product fails to adhere to standards, conventions or regulations relating to reliability.

Table 8.2.1 Maturity metrics

External maturity metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Estimated latent fault density	How many problems still exist that may emerge as future faults?	Count the number of faults detected during a defined trial period and predict potential number of future faults using a reliability growth estimation model.	X= {ABS(A1 - A2)} / B (X: Estimated residuary latent fault density) ABS()= Absolute Value A1 = total number of predicted latent faults in a software product A2 = total number of actually detected faults B= product size	0<=X It depends on stage of testing. At the later stages, smaller is better.	Absolute	A1= Count A2= Count B= Size X= Count/ Size	Test report Operation report Problem report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation 6.3 Quality Assurance	Developer Tester SQA User

FOOTNOTES

1 *When total number of actually detected faults becomes larger than total number of predicted latent faults, it is recommended to predict again and estimate more larger number. Estimated larger numbers are intended to predict reasonable latent failures, but not to make the product look better.*

2 *It is recommended to use several reliability growth estimation models and choose the most suitable one and repeat prediction with monitoring detected faults.*

3 *It may be helpful to predict upper and lower number of latent faults.*

4 *It is necessary to convert this value (X) to the <0,1> interval if making summarisation of characteristics.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Failure density against test cases	How many failures were detected during defined trial period?	Count the number of detected failures and performed test cases.	X= A1 / A2 A1 = number of detected failures A2 = number of performed test cases	0<=X It depends on stage of testing. At the later stages, smaller is better.	Absolute	A1= Count A2= Count B = Size X= Count/ Count	Test report Operation report Problem report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.3 Quality Assurance	Developer Tester SQA

FOOTNOTES

1 *The larger is the better, in early stage of testing. On the contrary, the smaller is the better, in later stage of testing or operation. It is recommended to monitor the trend of this measure along with the time.*

2 *This metric depends on adequacy of test cases so highly that they should be designed to include appropriate cases: e.g., normal, exceptional and abnormal cases.*

3 *It is necessary to convert this value (X) to the <0,1> interval if making summarisation of characteristics.*

Table 8.2.1 (*continued*)

External maturity metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Failure resolution	How many failure conditions are resolved?	Count the number of failures that did not reoccur during defined trial period under the similar conditions. Maintain a problem resolution report describing status of all the failures.	X= A1 / A2 A1 = number of resolved failures A2 = total number of actually detected failures	0<=X<= 1 The closer to 1.0 is better as more failures are resolved.	a) Absolute	A1= Count A2= Count A3 = Count X= Count/ Count	Test report Operation (test) report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User SQA Maintainer

FOOTNOTES

1 *It is recommended to monitor the trend when using this measure.*

2 *Total number of predicted latent failures might be estimated using reliability growth models adjusted with actual historical data relating to similar software product. In such a case, the number of actual and predicted failures can be comparable and the number of residual unresolved failures can be measurable.*

Fault density	How many faults were detected during defined trial period?	Count the number of detected faults and compute density.	X= A / B A = number of detected faults B = product size	0<=X It depends on stage of testing. At the later stages, smaller is better.	Absolute	A= Count B = Size X= Count/ Size	Test report Operation report Problem report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.3 Quality Assurance	Developer Tester SQA

FOOTNOTES

1 *The larger is the better, in early stage of testing. On the contrary, the smaller is the better, in later stage of testing or operation. It is recommended to monitor the trend of this measure along with the time.*

2 *The number of detected faults divided by the number of test cases indicates effectiveness of test cases.*

3 *It is necessary to convert this value (X) to the <0,1> interval if making summarisation of characteristics.*

4 *When counting faults, pay attention to the followings:*
- Possibility of duplication, because multiple reports may contain the same faults as other report;
- Possibility of others than faults, because users or testers may not figure out whether their problems are operation error, environmental error or software failure.

Table 8.2.1 (*continued*)

External maturity metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Fault removal	How many faults have been corrected?	Count the number of faults removed during testing and compare with the total number of faults detected and total number of faults predicted.	a) X= A1 / A2 A1 = number of corrected faults A2 = total number of actually detected faults b) Y= A1 / A3 A3 = total number of predicted latent faults in the software product	0<=X<= 1 The closer to 1.0 is better as fewer faults remain. 0<=Y The closer to 1.0 is better as fewer faults remain.	a) Absolute b) Absolute	A1= Count A2= Count A3= Count X= Count/ Count Y= Count/ Count	Test report Organization database	5.3 Integration 5.3 Qualification testing 6.5 Validation 6.3 Quality Assurance	Developer SQA Maintainer

FOOTNOTES

1 *It is recommended to monitor the trend during a defined period of time.*

2 *Total number of predicted latent faults may be estimated using reliability growth models adjusted with actual historical data relating to similar software product.*

3 *It is recommended to monitor the estimated faults resolution ratio Y, so that if Y > 1, investigate the reason whether it is because more faults have been detected early or because software product contains an unusual number of faults.*
Otherwise, when Y < 1, investigate whether it is because there are less than the usual number of defects in the software products or because the testing was not adequate to detect all possible faults.

4 *It is necessary to convert this value (Y) to the <0,1> interval if making summarisation of characteristics.*

5 *When counting faults, pay attention to the possibility of duplication, because multiple reports may contain the same faults as other report.*

Mean time between failures (MTBF)	How frequently does the software fail in operation?	Count the number of failures occurred during a defined period of operation and compute the average interval between the failures.	a) X = T1 / A b) Y = T2 / A T1 = operation time T2 = sum of time intervals between consecutive failure occurrences A = total number of actually detected failures (Failures occurred during observed operation time)	0<X,Y The longer is the better as longer time can be expected between failures.	a) Ratio b) Ratio	A = Count T1 = Time T2 = Time X =Time / Count Y =Time/ Count	Test report Operation (test) report	5.3 Integration 5.3 Qualification testing 5.4 Operation testing 5.4 Operation	Maintainer User

FOOTNOTES

1 *The following investigation may be helpful: - distribution of time interval between failure occurrences;*
- changes of mean time along with interval operation time period;
- distribution indicating which function has frequent failure occurrences and operation because of function and use dependency.

2 *Failure rate or hazard rate calculation may be alternatively used.*

3 *It is necessary to convert this value (X,Y) to the <0,1> interval if making summarisation of characteristics.*

Table 8.2.1 (*continued*)

External maturity metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Test coverage (Specified operation scenario testing coverage)	How much of required test cases have been executed during testing?	Count the number of test cases performed during testing and compare the number of test cases required to obtain adequate test coverage.	X= A / B A= Number of actually performed test cases representing operation scenario during testing B= Number of test cases to be performed to cover requirements	0<=X<=1 The closer to 1.0 is the better test coverage.	Absolute	A= Count B= Count X= Count/ Count	Req. spec., Test spec. or User manual Test report Operation report	5.3 Qualification testing 6.5 Validation 6.3 Quality Assurance	Developer Tester SQA
FOOTNOTE *Test cases may be normalised by software size, that is: test density coverage Y= A / C, where C= Size of product to be tested. The larger Y is the better. Size may be functional size that user can measure.*									
Test maturity	Is the product well tested? ***COMMENT(S)*** *This is to predict the success rate the product will achieve in future testing.*	Count the number of passed test cases which have been actually executed and compare it to the total number of test cases to be performed as per requirements.	X= A / B A= Number of passed test cases during testing or operation B= Number of test cases to be performed to cover requirements	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Req. spec., Test spec., or User manual Test report Operation report	5.3 Qualification testing 6.3 Quality Assurance	Developer Tester SQA

FOOTNOTES

1 *It is recommended to perform stress testing using live historical data especially from peak periods.*
It is also recommended to ensure that the following test types are executed and passed successfully:
- User operation scenario;
- Peak stress;
- Overloaded data input.

2 *Passed test cases may be normalised by software size, that is:*
passed test case density Y= A / C, where
C= Size of product to be tested.
The larger Y is better.
Size may be functional size that user can measure.

Table 8.2.2 Fault tolerance metrics

External fault tolerance metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Breakdown avoidance	How often the software product causes the break down of the total production environment?	Count the number of breakdowns occurrence with respect to number of failures. If it is under operation, analyse log of user operation history.	X= 1- A / B A= Number of breakdowns B= Number of failures	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A =Count B =Count X =Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User Maintainer

FOOTNOTES

1 *The breakdown means the execution of any user tasks is suspended until system is restarted, or its control is lost until system is forced to be shut down.*

2 *When none or few failures are observed, time between breakdowns may be more suitable.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Failure avoidance	How many fault patterns were brought under control to avoid critical and serious failures?	Count the number of avoided fault patterns and compare it to the number of fault patterns to be considered	X=A / B A= Number of avoided critical and serious failure occurrences against test cases of fault pattern B= Number of executed test cases of fault pattern (almost causing failure) during testing	0<=X<= 1 The closer to 1.0 is better, as the user can more often avoid critical or serious failure.	Absolute	A= Count B= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTES

1 *It is recommended to categorise failure avoidance levels which is the extent of mitigating impact of faults, for example:*
-Critical: entire system stops / or serious database destruction;
-Serious: important functions become inoperable and no alternative way of operating (workaround);
-Average: most functions are still available, but limited performance occurs with limited or alternate operation (workaround);
-Small: a few functions experience limited performance with limited operation;
-None: impact does not reach end user.

2 *Failure avoidance levels may be based on a risk matrix composed by severity of consequence and frequency of occurrence provided by ISO/IEC 15026 System and software integrity.*

3 *Fault pattern examples*
- out of range data
- deadlock
Fault tree analysis technique may be used to detect fault patterns.

4 *Test cases can include the human incorrect operation.*

Table 8.2.2 (*continued*)

External fault tolerance metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Incorrect operation avoidance	How many functions are implemented with incorrect operations avoidance capability?	Count the number of test cases of incorrect operations which were avoided to cause critical and serious failures and compare it to the number of executed test cases of incorrect operation patterns to be considered.	X=A / B A= Number of avoided critical and serious failures occurrences B= Number of executed test cases of incorrect operation patterns (almost causing failure) during testing	0<=X<= 1 The closer to 1.0 is better, as more incorrect user operation is avoided.	Absolute	A= Count B= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User Maintainer

FOOTNOTES

1 *Also data damage in addition to system failure.*

2 *Incorrect operation patterns*
- Incorrect data types as parameters
- Incorrect sequence of data input
- Incorrect sequence of operation.

3 *Fault tree analysis technique may be used to detect incorrect operation patterns.*

4 *This metric may be used experimentally.*

Table 8.2.3 Recoverability metrics

External recoverability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Availability	How available is the system for use during the specified period of time?	Test system in a production like environment for a specified period of time performing all user operations. Measure the repair time period each time the system was unavailable during the trial. Compute mean time to repair.	a) X= { To / (To + Tr) } b) Y= A1 / A2 To = operation time Tr = time to repair A1= total available cases of user's successful software use when user attempt to use A2= total number of cases of user's attempt to use the software during observation time. This is from the user callable function operation view.	0<=X<=1 The larger and closer to 1.0 is better, as the user can use the software for more time. 0<=Y<=1 The larger and closer to 1.0 is the better.	(a),(b) Absolute	To = Time Tr = Time X= Time/ Time A1= Count A2= Count Y= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User Maintainer

FOOTNOTE
It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Mean down time	What is the average time the system stays unavailable when a failure occurs before gradual start up?	Measure the down time each time the system is unavailable during a specified trial period and compute the mean time.	X= T / N T= Total down time N= Number of observed breakdowns The worst case or distribution of down time should be measured.	0<X The smaller is the better, system will be down for shorter time.	Ratio	T= Time N= Count X= Time/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTES
1 *It is recommended that this recoverability metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.*
2 *It is necessary to convert this value (X) to the <0,1> interval if making summarisation of characteristics.*

Table 8.2.3 (*continued*)

External recoverability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Mean recovery time	What is the average time the system takes to complete recovery from initial partial recovery?	Measure the full recovery times for each of the time the system was brought down during the specified trial period and compute the mean time.	X= Sum(T) / B T= Time to recovery downed software system at each opportunity N= Number of cases which observed software system entered into recovery	0<X The smaller is the better.	Ratio	T= Time N= Count X= Time/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTES

1 *It is recommended to measure the maximum time of the worst case or distribution of recovery time for many cases.*

2 *It is recommended that this recoverability metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.*

3 *It is recommended to distinguish the grades of recovery difficulty, for example, recovery of destroyed database is more difficult than recovery of destroyed transaction.*

4 *It is necessary to convert this value (X) to the <0,1> interval if making summarisation of characteristics.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Restartability	How often the system can restart providing service to users within a required time?	Count the number of times the system restarts and provides service to users within a target required time and compare it to the total number of restarts, when the system was brought down during the specified trial period.	X = A / B A= Number of restarts which met to required time during testing or user operation support B= Total number of restarts during testing or user operation support	0<=X<=1 The larger and closer to 1.0 is better, as the user can restart easily.	Absolute	A =Count B =Count X =Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTES

1 *It is recommended to estimate different time to restart to correspond to the severity level of inoperability, such as data base destruction, lost multi transaction, lost single transaction, or temporary data destruction.*

2 *It is recommended that this recoverability metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Restorability	How capable is the product in restoring itself after abnormal event or at request?	Count the number of successful restorations and compare it to the number of tested restoration required in the specifications. Restoration requirement examples: database checkpoint, transaction checkpoint, redo function, undo function etc.	X= A / B A= Number of restoration cases successfully done B= Number of restoration cases tested as per requirements	0<=X<=1 The larger and closer to 1.0 is better, as he product is more capable to restore in defined cases.	Absolute	A= Count B= Count X= Count/ Count	Req. spec., Test spec. or User manual Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTE

It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.

Table 8.2.3 (*continued*)

External recoverability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Restore effectiveness	How effective is the restoration capability?	Count the number of tested restoration meeting target restoration time and compare it to the number of restorations required with specified target time.	X= A / B A= Number of cases successfully restored meeting the target restore time B= Number of cases performed	0<=X<=1 The larger and closer to 1.0 is the better, as the restoration process in product is more effective.	Absolute	A= Count B= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer

FOOTNOTE

It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.

Table 8.2.4 Reliability compliance metrics

External reliability compliance metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Reliability compliance	How compliant is the reliability of the product to applicable regulations, standards and conventions.	Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance as in the specification.	X = 1 - A / B A= Number of reliability compliance items specified that have not been implemented during testing B= Total number of reliability compliance items specified	0<= X <=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTE
It may be useful to collect several measured values along time, to analyse the trend of increasingly satisfied compliance items and to determine whether they are fully satisfied or not.

8.3 Usability metrics

Usability metrics measure the extent to which the software can be understood, learned, operated, attractive and compliant with usability regulations and guidelines.

Many external usability metrics are tested by users attempting to use a function. The results will be influenced by the capabilities of the users and the host system characteristics. This does not invalidate the measurements, since the evaluated software is run under explicitly specified conditions by a sample of users who are representative of an identified user group. (For general-purpose products, representatives of a range of user groups may be used.) For reliable results a sample of at least eight users is necessary, although useful information can be obtained from smaller groups. Users should carry out the test without any hints or external assistance.

Metrics for understandability, learnability and operability have two types of method of application: user test or test of the product in use.

NOTE 1 User test

Users attempting to use a function test many external metrics. These measures can vary widely among different individuals. A sample of users who are representative of an identified user group should carry out the test without any hints or external assistance. (For general-purpose products, representatives of a range of user groups may be used.) For reliable results a sample of at least eight users is necessary, although useful information can be obtained from smaller groups.

It should be possible for the measures to be used to establish acceptance criteria or to make comparisons between products. This means that the measures should be counting items of known value. Results should report the mean value and the standard error of the mean.

Many of these metrics can be tested with early prototypes of the software. Which metrics are to be applied will depend on the relative importance of different aspects of usability, and the extent of subsequent quality in use testing.

NOTE 2 Test of the product in use

Rather than test specific functions, some external metrics observe the use of a function during more general use of the product to achieve a typical task as part of a test of the quality in use (ISO/IEC 9126-4). This has the advantage that fewer tests are required. The disadvantage is that some functions may only rarely be used during normal use.

It should be possible for the measures to be used to establish acceptance criteria or to make comparisons between products. This means that the measures should be counting items of known value. Results should report the mean value and the standard error of the mean.

8.3.1 Understandability metrics

Users should be able to select a software product, which is suitable for their intended use. An external understandability metric should be able to assess whether new users can understand:

- whether the software is suitable
- how it can be used for particular tasks.

8.3.2 Learnability metrics

An external learnability metric should be able to assess how long users take to learn how to use particular functions, and the effectiveness of help systems and documentation.

Learnability is strongly related to understandability, and understandability measurements can be indicators of the learnability potential of the software.

8.3.3 Operability metrics

An external operability metric should be able to assess whether users can operate and control the software. Operability metrics can be categorized by the dialogue principles in ISO 9241-10:

- suitability of the software for the task
- self-descriptiveness of the software
- controllability of the software
- conformity of the software with user expectations
- error tolerance of the software
- suitability of the software for individualization

The choice of functions to test will be influenced by the expected frequency of use of functions, the criticality of the functions, and any anticipated usability problems.

8.3.4 Attractiveness metrics

An external attractiveness metric should be able to assess the appearance of the software, and will be influenced by factors such as screen design and colour. This is particularly important for consumer products.

8.3.5 Usability compliance metrics

An external usability compliance metric should be able to assess adherence to standards, conventions, style guides or regulations relating to usability.

Table 8.3.1 Understandability metrics

External understandability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Completeness of description	What proportion of functions (or types of functions) is understood after reading the product description?	Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of functions which are adequately understood and compare with the total number of functions in the product.	X = A / B A = Number of functions (or types of functions) understood B = Total number of functions (or types of functions)	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether potential users understand the capability of the product after reading the product description.*									
Demonstration accessibility	What proportion of the demonstrations/ tutorials can the user access?	Conduct user test and observe user behaviour. Count the number of functions that are adequately demonstrable and compare with the total number of functions requiring demonstration capability.	X = A / B A= Number of demonstrations / tutorials that the user successfully accesses B= Number of demonstrations / tutorials available	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether users can find the demonstrations and/or tutorials.*									
Demonstration accessibility in use	What proportion of the demonstrations / tutorials can the user access whenever user actually needs to do during operation?	Observe the behaviour of the user who is trying to see demonstration/tutorial. Observation may employ human cognitive action monitoring approach with video camera.	X = A / B A= Number of cases in which user successfully sees demonstration when user attempts to see demonstration B= Number of cases in which user attempts to see demonstration during observation period	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report User monitoring record (video tape and action record)	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether users can find the demonstrations and/or tutorials while using the product.*									

Table 8.3.1 (*continued*)

External understandability metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Demonstration effectiveness	What proportion of functions can the user operate successfully after a demonstration or tutorial?	Observe the behaviour of the user who is trying to see demonstration/tutorial. Observation may employ human cognitive action monitoring approach with video camera.	X = A / B A= Number of functions operated successfully B= Number of demonstrations/tutorials accessed	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether users can operate functions successfully after an online demonstration or tutorial.*									
Evident functions	What proportion of functions (or types of function) can be identified by the user based upon start up conditions?	Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of functions that are evident to the user and compare with the total number of functions.	X = A / B A = Number of functions (or types of functions) identified by the user B = Total number of actual functions (or types of functions)	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether users are able to locate functions by exploring the interface (e.g. by inspecting the menus).*									
Function understand-ability	What proportion of the product functions will the user be able to understand correctly?	Conduct user test and interview user with questionnaires. Count the number of user interface functions where purposes are easily understood by the user and compare with the number of functions available for user.	X= A / B A= Number of interface functions whose purpose is correctly described by the user B= Number of functions available from the interface	0 <= X <= 1 The closer to 1.0, the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This indicates whether users are able to understand functions by exploring the interface (e.g. by inspecting the menus).*									

Table 8.3.1 (*continued*)

External understandability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Understandable input and output	Can users understand what is required as input data and what is provided as output by software system?	Conduct user test and interview user with questionnaires or observe user behaviour. Count the number of input and output data items understood by the user and compare with the total number of them available for user.	X= A / B A= Number of input and output data items which user successfully understands B= Number of input and output data items available from the interface	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Maintainer

FOOTNOTE

This indicates whether users can understand the format in which data should be input and correctly identify the meaning of output data.

Table 8.3.2 Learnability metrics

External learnability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Ease of function learning	How long does the user take to learn to use a function?	Conduct user test and observe user behaviour.	T= Mean time taken to learn to use a function correctly	0<T The shorter is the better.	Ratio	T= Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTE *This metric is generally used as one of experienced and justified.*									
Ease of learning to perform a task in use	How long does the user take to learn how to perform the specified task efficiently?	Observe user behaviour from when they start to learn until they begin to operate efficiently.	T= Sum of user operation time until user achieved to perform the specified task within a short time	0<T The shorter is the better.	Ratio	T= Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Maintainer
FOOTNOTES 1 *It is recommended to determine an expected user's operating time as a short time. Such user's operating time may be the threshold, for example, which is 70% of time at the first use as the fair proportion.* 2 *Effort may alternatively represent time by person-hour unit.*									
Effectiveness of the user documentation and/or help system	What proportion of tasks can be completed correctly after using the user documentation and/or help system?	Conduct user test and observe user behaviour. Count the number of tasks successfully completed after accessing online help and/or documentation and compare with the total number of tasks tested.	X= A / B A= Number of tasks successfully completed after accessing online help and/or documentation B = Total of number of tasks tested	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTE *Three metrics are possible: completeness of the documentation, completeness of the help facility, or completeness of the help and documentation used in combination.*									

Table 8.3.2 (*continued*)

External learnability metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Effectiveness of user documentation and/or help systems in use	What proportion of functions can be used correctly after reading the documentation or using help systems?	Observe user behaviour. Count the number of functions used correctly after reading the documentation or using help systems and compare with the total number of functions.	X = A / B A = Number of functions that can be used B = Total of number of functions provided	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTE *This metric is generally used as one of experienced and justified metrics rather than the others.*									
Help accessibility	What proportion of the help topics can the user locate?	Conduct user test and observe user behaviour. Count the number of tasks for which correct online help is located and compare with the total number of tasks tested.	X = A / B A = Number of tasks for which correct online help is located B = Total of number of tasks tested	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
Help frequency	How frequently does a user have to access help to learn operation to complete his/her work task?	Conduct user test and observe user behaviour. Count the number of cases that a user accesses help to complete his/her task.	X = A A = Number of accesses to help until a user completes his/her task.	0<= X The closer to 0 is the better.	Absolute	X= Count A =Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

Table 8.3.3 Operability metrics a) Conforms with operational user expectations

External Operability metrics a) Conforms with operational user expectations

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Operational consistency in use	How consistent are the component of the user interface?	Observe the behaviour of the user and ask the opinion.	a) X = 1 - A / B A= Number of messages or functions which user found unacceptably inconsistent with the user's expectation B= Number of messages or functions b) Y = N / UOT N= Number of operations which user found unacceptably inconsistent with the user's expectation UOT= user operating time (during observation period)	0<=X<=1 The closer to 1.0 is the better. 0<=Y The smaller and closer to 0.0 is the better.	a) Absolute b) Ratio	A= Count B= Count X= Count/ Count UOT= Time N= Count Y= Count/ Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTES

1 *User's experience of operation is usually helpful to recognise several operation patterns, which derive user's expectation.*

2 *Both of "input predictability" and "output predictability" are effective for operational consistency.*

3 *This metric may be used to measure "Easy to derive operation" and "Smooth Communication".*

Table 8.3.3 Operability metrics b) Controllable

External Operability metrics b) Controllable

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Error correction	Can user easily correct error on tasks?	Conduct user test and observe user behaviour.	T= Tc - Ts Tc = Time of completing correction of specified type errors of performed task Ts = Time of starting correction of specified type errors of performed task	0<T The shorter is the better.	Ratio	Ts, Tc= Time T= Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTE *User of this metric is suggested to specify types of errors for test cases by considering, for example, severity (displaying error or destroying data), type of input/output error (input text error, output data error to database or graphical error on display) or type of error operational situation (interactive use or emergent operation).*									
Error correction in use	Can user easily recover his/her error or retry tasks?	Observe the behaviour of the user who is operating software.	a) X= A / UOT A= number of times that the user succeeds to cancel their error operation UOT= user operating time during observation period ***COMMENT(S)*** *When function is tested one by one, the ratio can be also calculated, that is the ratio of number of functions which user succeeds to cancel his/her operation to all functions.*	0<=X The higher is the better.	Ratio	A= Count UOT = Time X = Count / Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
	Can user easily recover his/her input?	Observe the behaviour of the user who is operating software.	b) X = A / B A= Number of screens or forms where the input data were successfully modified or changed before being elaborated B = Number of screens or forms where user tried to modify or to change the input data during observed user operating time	0<=X<=1 The closer to 1.0 is the better.	Absolute	A= Count, B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

Table 8.3.3 Operability metrics c) Suitable for the task operation

External Operability metrics c) Suitable for the task operation

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Default value availability in use	Can user easily select parameter values for his/her convenient operation?	Observe the behaviour of the user who is operating software. Count how many times user attempts to establish or to select parameter values and fails, (because user can not use default values provided by the software).	X = 1 - A / B A= The number of times that the user fail to establish or to select parameter values in a short period (because user can not use default values provided by the software) B= Total number of times that the user attempt to establish or to select parameter values	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTES

1 *It is recommended to observe and record operator's behaviour and decide how long period is allowable to select parameter values as "short period".*

2 *When parameter setting function is tested by each function, the ratio of allowable function can be also calculated.*

3 *It is recommended to conduct functional test that covers parameter-setting functions.*

Table 8.3.3 Operability metrics d) Self descriptive (Guiding)

External Operability metrics d) Self descriptive (Guiding)

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Message understand-ability in use	Can user easily understand messages from software system? Is there any message which caused the user a delay in understanding before starting the next action? Can user easily memorise important message?	Observe user behaviour who is operating software.	X = A / UOT A = number of times that the user pauses for a long period or successively and repeatedly fails at the same operation, because of the lack of message comprehension. UOT = user operating time (observation period)	0<=X The smaller and closer to 0.0 is the better.	Ratio	A =Count UOT = Time X = Count / Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTES

1 *The extent of ease of message comprehension is represented by how long that message caused delay in user understanding before starting the next action. Therefore, it is recommended to observe and record operator's behaviour and decide what length of pause is considered a "long period".*

2 *It is recommended to investigate the following as possible causes of the problems of user's message comprehension.*
a)Attentiveness : Attentiveness implies that user successfully recognises important messages presenting information such as guidance on next user action, name of data items to be looked at, and warning of careful operation.
- Does user ever fail to watch when encountering important messages?
- Can user avoid mistakes in operation, because of recognising important messages?
b) Memorability: Memorability implies that user remember important messages presenting information such as guidance on the next user action, name of data items to be looked at, and warning of careful operation.
- Can user easily remember important messages?
- Is remembering important messages helpful to the user?
- Is it required for the user to remember only a few important messages and not so much?

3 *When messages are tested one by one, the ratio of comprehended messages to the total can be also calculated.*

4 *When several users are observed who are participants of operational testing, the ratio of users who comprehended messages to all users can be calculated.*

Self-explanatory error messages	In what proportion of error conditions does the user propose the correct recovery action?	Conduct user test and observe user behaviour.	X= A / B A =Number of error conditions for which the user proposes the correct recovery action B =Number of error conditions tested	0 <= X <= 1 The closer to 1.0 is the better.	Absolute	X =Count/ Count A =Count B =Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTE

This metric is generally used as one of experienced and justified.

Table 8.3.3 Operability metrics e) Operational error tolerant (Human error free)

External operability metrics e) Operational error tolerant (Human error free)

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Operational error recoverability in use	Can user easily recover his/her worse situation?	Observe the behaviour of the user who is operating software.	X = 1 - A / B A= Number of unsuccessfully recovered situation (after a user error or change) in which user was not informed about a risk by the system B= Number of user errors or changes	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count, B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTE *The formula above is representative of the worst case. User of this metric may take account of the combination of 1) the number of errors where the user is / is not warned by the software system and 2) the number of occasions where the user successfully / unsuccessfully recovers the situation.*									
Time between human error operations in use	Can user operate the software long enough without human error?	Observe the behaviour of the user who is operating software.	X = T / N (at time t during [t-T, t]) T = operation time period during observation (or The sum of operating time between user's human error operations) N= number of occurrences of user's human error operation	0<X The higher is the better.	Ratio	T = Time N = Count X = Time / Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTES 1 *Human error operation may be detected by counting below user's behaviour:* *a) Simple human error (Slips): The number of times that the user just simply makes errors to input operation;* *b) Intentional error (Mistakes): The number of times that the user repeats fail an error at the same operation with misunderstanding during observation period;* *c) Operation hesitation pause: The number of times that the user pauses for a long period with hesitation during observation period.* *User of this metric is suggested to measure separately for each type listed above.* 2 *It seems that an operation pause implies a user's hesitation operation.* *It depends on the function, operation procedure, application domain, and user whether it is considered a long period or not for the user to pause the operation. Therefore, the evaluator is requested to take them into account and determine the reasonable threshold time. For an interactive operation, a "long period" threshold range of 1min. to 3 min.*									
Undoability (User error correction)	How frequently does the user successfully correct input errors?	Conduct user test and observe user behaviour.	a) X= A / B A= Number of input errors which the user successfully corrects B= Number of attempts to correct input errors	0<=X<=1 The closer to 1.0 is the better.	a) Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
	How frequently does the user correctly undo errors?	Conduct user test and observe user behaviour.	b) Y= A / B A= Number of error conditions which the user successfully corrects B= Total number of error conditions tested	0 <= Y <= 1 The closer to 1.0 is the better.	b) Absolute	A= Count B= Count Y= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
FOOTNOTE *This metric is generally used as one of experienced and justified.*									

Table 8.3.3 Operability metrics f) Suitable for individualisation

External operability metrics f) Suitable for individualisation

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Customisability	Can user easily customise operation procedures for his/her convenience? Can a user, who instructs end users, easily set customised operation procedure templates for preventing their errors? What proportion of functions can be customised?	Conduct user test and observe user behaviour.	X= A / B A= Number of functions successfully customised B= Number of attempts to customise	0 <= X <= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	User manual Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTES

1 *Ratio of user's failures to customise may be measured.*
Y = 1 - (C / D)
C = Number of cases in which a user fails to customise operation
D = Total number of cases in which a user attempted to customise operation for his/her convenience.
0<=Y<= 1, The closer to 1.0 is the better.

2 *It is recommended to regard the following as variations of customising operations:*
- chose alternative operation, such as using menu selection instead of command input;
- combined user's operation procedure, such as recording and editing operation procedures;
- set constrained template operation, such as programming procedures or making a template for input guidance.

3 *This metric is generally used as one of experienced and justified.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Operation procedure reduction	Can user easily reduce operation procedures for his/her convenience?	Count user's strokes for specified operation and compare them between before and after customising operation.	X = 1 - A / B A = Number of reduced operation procedures after customising operation B = Number of operation procedures before customising operation	0<=X< 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTES

1 *It is recommended to take samples for each different user task and to distinguish between an operator who is a skilled user or a beginner.*

2 *Number of operation procedures may be represented by counting operation strokes such as click, drug, key touch, screen touch, etc.*

3 *This includes keyboard shortcuts.*

Table 8.3.3 f) (*continued*)

External operability metrics f) Suitable for individualisation									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Physical accessibility	What proportion of functions can be accessed by users with physical handicaps?	Conduct user test and observe user behaviour.	X= A / B A= Number of functions successfully accessed B= Number of functions	0 <= X <= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTE
Examples of physical inaccessibility are inability to use a mouse and blindness.

Table 8.3.4 Attractiveness metrics

External attractiveness metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Attractive interaction	How attractive is the interface to the user?	Questionnaire to users.	Questionnaire to assess the attractiveness of the interface to users, after experience of usage	Depend on its questionnaire scoring method.	Absolute	Count	Questionnaire result	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer
Interface appearance customisability	What proportion of interface elements can be customised in appearance to the user's satisfaction?	Conduct user test and observe user behaviour.	X= A / B A= Number of interface elements customised in appearance to user's satisfaction B= Number of interface elements that the user wishes to customise	0 <= X <= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Users' requests Operation (test) report	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

FOOTNOTE

This metric is generally used as one of experienced and justified.

Table 8.3.5 Usability compliance metrics

External usability compliance metrics

Metric name	Purpose	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Usability compliance	How completely does the software adhere to the standards, conventions, style guides or regulations relating to usability?	Specify required compliance items based on standards, conventions, style guides or regulations relating to usability. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	X = 1 - A / B A= Number of usability compliance items specified that have not been implemented during testing B= Total number of usability compliance items specified	0<= X <=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions, style guides or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTE
It may be useful to collect several measured values along time, to analyse the trend of increasingly satisfied compliance items and to determine whether they are fully satisfied or not.

8.4 Efficiency metrics

An external efficiency metric should be able to measure such attributes as the time consumption and resource utilisation behaviour of computer system including software during testing or operations.

It is recommended that the maximal and distribution time are investigated for many cases of testing or operations, because the measure is affected strongly and fluctuates depending on the conditions of use, such as load of processing data, frequency of use, number of connecting sites and so on. Therefore, efficiency metrics may include the ratio of measured actual value with error fluctuation to the designed value with allowed error fluctuation range, required by specification.

It is recommended to list and to investigate the role played by factors such as “CPU” and memory used by other software, network traffic, and scheduled background processes. Possible fluctuations and valid ranges for measured values should be established and compared to requirement specifications.

It is recommended that a task be identified and defined to be suitable for software application: for example, a transaction as a task for business application: a switching or data packet sending as a task for communication application; an event control as a task for control application; and an output of data produced by user callable function for common user application.

NOTE 1 Response time: Time needed to get the result from pressing a transmission key. This means that response time includes processing time and transmission time. Response time is applicable only for an interactive system. There is no significant difference when it is a standalone system. However, in the case of Internet system or other real time system, sometimes transmission time is much longer.

NOTE 2 Processing time: The elapsed time in a computer between receiving a message and sending the result. Sometimes it includes operating overhead time, other times it only means time used for an application program.

NOTE 3 Turn around time: Time needed to get the result from a request. In many cases one turn around time includes many responses. For example, in a case of banking cash dispenser, turn around time is a time from pressing initial key until you get money, meanwhile you must select type of transaction and wait for a message, input password and wait for the next message etc.

8.4.1 Time behaviour metrics

An external time behaviour metric should be able to measure such attributes as the time behaviour of computer system including software during testing or operations.

8.4.2 Resource utilization metrics

An external resource utilization metric should be able to measure such attributes as the utilized resources behaviour of computer system including software during testing or operating.

8.4.3 Efficiency compliance metrics

An external efficiency compliance metric should be able to measure an attribute such as the number of functions with, or occurrences of compliance problems, which is the software product failing to adhere to standards, conventions or regulations relating to efficiency.

Table 8.4.1 Time behaviour metrics a) Response time

External time behaviour metrics a) Response time

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Response time	What is the time taken to complete a specified task? How long does it take before the system response to a specified operation?	Start a specified task. Measure the time it takes for the sample to complete its operation. Keep a record of each attempt.	T = (time of gaining the result) - (time of command entry finished)	0 < T The sooner is the better.	Ratio	T= Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
FOOTNOTE *It is recommended to take account of time bandwidth and to use statistical analysis with measures for a lot of tasks (sample shots) and not for only one task.*									
Response time (Mean time to response)	What is the average wait time the user experiences after issuing a request until the request is completed within a specified system load in terms of concurrent tasks and system utilisation?	Execute a number of scenarios of concurrent tasks. Measure the time it takes to complete the selected operation(s). Keep a record of each attempt and compute the mean time for each scenario.	X = Tmean / TXmean Tmean = ∑(Ti) / N, (for i=1 to N) TXmean = required mean response time Ti= response time for i-th evaluation (shot) N= number of evaluations (sampled shots)	0 <= X The nearer to 1.0 and less than 1.0 is the better.	Absolute	Tmean= Time TXmean= Time Ti= Time N= Count X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
FOOTNOTE *Required mean response time can be derived from specification of required real-time processing, user expectation of business needs or observation of user reaction. A user cognitive of the aspects of human ergonomics might be considered.*									

Table 8.4.1 a) (*continued*)

External time behaviour metrics a) Response time									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Response time (Worst case response time ratio)	What is the absolute limit on time required in fulfilling a function? In the worst case, can user still get response within the specified time limit? In the worst case, can user still get reply from the software within a time short enough to be tolerable for user?	Calibrate the test. Emulate a condition whereby the system reaches a maximum load situation. Run application and monitor result(s).	X= Tmax / Rmax Tmax= MAX(Ti) (for i=1 to N) Rmax = required maximum response time MAX(Ti)= maximum response time among evaluations N= number of evaluations (sampled shots) Ti= response time for i-th evaluation (shot) **COMMENT(S)** *Distribution may be calculated as illustrated below.* *Statistical maximal ratio Y= Tdev / Rmax* *Tdev = Tmean + K (DEV)* *Tdev is time deviated from mean time to the particular time: e.g. 2 or 3 times of standard deviation.* *K: coefficient (2 or 3)* *DEV=SQRT{ ∑((Ti-Tmean) **2) / (N-1)} (for i=1 to N)* *Tmean = ∑(Ti) / N, (for i=1 to N)* *TXmean = required mean response time*	0 < X The nearer to 1 and less than 1 is the better.	Absolute	Tmax= Time Rmax= Time Ti= Time N= Count X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 8.4.1 Time behaviour metrics b) Throughput

External time behaviour metrics b) Throughput

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Target audience
Throughput	How many tasks can be successfully performed over a given period of time?	Calibrate each task according to the intended priority given. Start several job tasks. Measure the time it takes for the measured task to complete its operation. Keep a record of each attempt.	X = A / T A = number of completed tasks T = observation time period	0 < X The larger is the better.	Ratio	A= Count T= Time X= Count/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA
Throughput (Mean amount of throughput)	What is the average number of concurrent tasks the system can handle over a set unit of time?	Calibrate each task according to intended priority. Execute a number of concurrent tasks. Measure the time it takes to complete the selected task in the given traffic. Keep a record of each attempt.	X = Xmean / Rmean Xmean = ∑(Xi)/N Rmean = required mean throughput Xi = Ai / Ti Ai = number of concurrent tasks observed over set period of time for i-th evaluation Ti = set period of time for i-th evaluation N = number of evaluations	0 < X The larger is the better.	Absolute	Xmean= Count Rmean= Count Ai= Count Ti= Time Xi= Count/ Time N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA
Throughput (Worst case throughput ratio)	What is the absolute limit on the system in terms of the number and handling of concurrent tasks as throughput?	Calibrate the test. Emulate the condition whereby the system reaches a situation of maximum load. Run job tasks concurrently and monitor result(s).	X = Xmax / Rmax Xmax = MAX(Xi) (for i = 1 to N) Rmax = required maximum throughput. MAX(Xi) = maximum number of job tasks among evaluations Xi = Ai / Ti Ai = number of concurrent tasks observed over set period of time for i-th evaluation Ti = set period of time for i-th evaluation N= number of evaluations	0 < X The larger is the better.	Absolute	Xmax= Count Rmax= Count Ai= Count Ti= Time Xi= Count/ Time N= Count Xdev= Count X = Count/ Count	Testing report Operation report showing elapse time	5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA

FOOTNOTE

1 *Distribution may be calculated as illustrated below.*
Statistical maximal ratio Y= Xdev / Xmax
Xdev = Xmean + K (DEV)
Xdev is time deviated from mean time to the particular time: e.g. 2 or 3 times of standard deviation.
K: coefficient (2 or 3)
*DEV=SQRT{ ∑((Xi-Xmean) **2) / (N-1)} (for i=1 to N)*
Xmean = ∑(Xi)/N

Table 8.4.1 Time behaviour metrics c) Turnaround time

External time behaviour metrics c) Turnaround time

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Turnaround time	What is the wait time the user experiences after issuing an instruction to start a group of related tasks and their completion?	Calibrate the test accordingly. Start the job task. Measure the time it takes for the job task to complete its operation. Keep a record of each attempt.	T = Time between user's finishing getting output results and user's finishing request	0 < T The shorter the better.	Ratio	T= Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

FOOTNOTE

It is recommended to take account of time bandwidth and to use statistical analysis with measures for many tasks (sample shots), not only one task (shot).

Turnaround time (Mean time for turnaround)	What is the average wait time the user experiences after issuing an instruction to start a group of related tasks and their completion within a specified system load in terms of concurrent tasks and system utilisation?	Calibrate the test. Emulate a condition where a load is placed on the system by executing a number of concurrent tasks (sampled shots). Measure the time it takes to complete the selected job task in the given traffic. Keep a record of each attempt.	X = Tmean/TXmean Tmean = ∑(Ti)/N, (for i=1 to N) TXmean = required mean turnaround time Ti = turnaround time for i-th evaluation (shot) N = number of evaluations (sampled shots)	0 < X The shorter is the better.	Absolute	Tmean= Time TXmean= Time Ti= Time N= Count X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 8.4.1 c) (*continued*)

External time behaviour metrics c) Turnaround time

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Turnaround time (Worst case turnaround time ratio)	What is the absolute limit on time required in fulfilling a job task? In the worst case, how long does it take for software system to perform specified tasks?	Calibrate the test. Emulate a condition where by the system reaches maximum load in terms of tasks performed. Run the selected job task and monitor result(s).	X= Tmax / Rmax Tmax= MAX(Ti) (for i=1 to N) Rmax = required maximum turnaround time MAX(Ti)= maximum turnaround time among evaluations N= number of evaluations (sampled shots) Ti= turnaround time for i-th evaluation (shot)	0 < X The nearer to 1.0 and less than 1.0 is the better.	Absolute	X= Time/ Time Tmax = Time Rmax= Time Ti= Time N= Count Tdev = Time	Testing report Operation report showing elapse time	5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

FOOTNOTE

Distribution may be calculated as illustrated below.

Statistical maximal ratio Y= Tdev / Rmax

Tdev = Tmean + K (DEV)

Tdev is time deviated from mean time to the particular time: e.g. 2 or 3 times of standard deviation.

K: coefficient (2 or 3)

*DEV=SQRT{ ∑((Ti-Tmean) **2) / (N-1)} (for i=1 to N)*

Tmean = ∑(Ti) / N, (for i=1 to N)

TXmean = required mean turnaround time

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Waiting time	What proportion of the time do users spend waiting for the system to respond?	Execute a number of scenarios of concurrent tasks. Measure the time it takes to complete the selected operation(s). Keep a record of each attempt and compute the mean time for each scenario.	X = Ta / Tb Ta = total time spent waiting Tb = task time	0<= X The smaller the better.	Absolute	Ta= Time Tb= Time X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

FOOTNOTE

If the tasks can be partially completed, the Task efficiency metric should be used when making comparisons.

Table 8.4.2 Resource utilization metrics a) I/O devices resource utilization

External resource utilisation metrics	a) I/O devices resource utilisation								
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
I/O devices utilisation	Is the I/O device utilisation too high, causing inefficiencies?	Execute concurrently a large number of tasks, record I/O device utilisation, and compare with the design objectives.	X = A / B A = time of I/O devices occupied B = specified time which is designed to occupy I/O devices	0 <= X <= 1 The less than and nearer to the 1.0 is the better.	Absolute	A= Time B= Time X= Time/ Time	Testing report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer SQA
I/O loading limits	What is the absolute limit on I/O utilisation in fulfilling a function?	Calibrate the test condition. Emulate a condition whereby the system reaches a situation of maximum load. Run application and monitor result(s).	X = Amax / Rmax Amax = MAX(Ai), (for i = 1 to N) Rmax = required maximum I/O messages MAX(Ai) = Maximum number of I/O messages from 1st to i-th evaluation N= number of evaluations.	0<= X The smaller is the better.	Absolute	Amax = Count Rmax = Count Ai = Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
I/O related errors	How often does the user encounter problems in I/O device related operations?	Calibrate the test conditions. Emulate a condition whereby the system reaches a situation of maximum I/O load. Run the application and record number of errors due to I/O failure and warnings.	X = A / T A = number of warning messages or system failures T = User operating time during user observation	0 <= X The smaller is the better.	Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA
Mean I/O fulfilment ratio	What is the average number of I/O related error messages and failures over a specified length of time and specified utilisation?	Calibrate the test condition. Emulate a condition whereby the system reaches a situation of maximum load. Run the application and record number of errors due to I/O failure and warnings.	X = Amean / Rmean Amean = ∑(Ai)/N Rmean = required mean number of I/O messages Ai = number of I/O error messages for i-th evaluation N = number of evaluations	0<= X The smaller is the better.	Absolute	Amean = Count Rmean = Count Ai = Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
User waiting time of I/O devices utilisation	What is the impact of I/O device utilisation on the user wait times?	Execute concurrently a large amount of tasks and measure the user wait times as a result of I/O device operation.	T = Time spent to wait for finish of I/O devices operation	0 < T The shorter is the better.	Ratio	T= Time	Testing report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

FOOTNOTE
It is recommended that the maximal and distributed time are to be investigated for several cases of testing or operating, because the measures are tend to be fluctuated by condition of use.

Table 8.4.2 Resource utilization metrics b) Memory resource utilization

External resource utilisation metrics b) Memory resource utilisation

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Maximum memory utilisation	What is the absolute limit on memory required in fulfilling a function?	Calibrate the test condition. Emulate a condition whereby the system reaches a situation of maximum load. Run application and monitor result(s).	X = Amax / Rmax Amax = MAX(Ai), (for i = 1 to N) Rmax = required maximum memory related error messages MAX(Ai) = Maximum number of memory related error messages from 1st to i-th evaluation N= number of evaluations	0<= X The smaller is the better.	Absolute	Amax= Count Rmax= Count Ai= Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Mean occurrence of memory error	What is the average number of memory related error messages and failures over a specified length of time and a specified load on the system?	Calibrate the test condition. Emulate a condition whereby the system reaches a situation of maximum load. Run the application and record number of errors due to memory failure and warnings.	X = Amean / Rmean Amean = ∑(Ai)/N Rmean = required mean number of memory related error messages Ai = number of memory related error messages for i-th evaluation N = number of evaluations	0<= X The smaller is the better.	Absolute	Amean= Count Rmean= Count Ai= Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Ratio of memory error/time	How many memory errors were experienced over a set period of time and specified resource utilisation?	Calibrate the test conditions. Emulate a condition whereby the system reaches a situation of maximum load. Run the application and record number of errors due to memory failure and warnings.	X = A / T A = number of warning messages or system failures T = User operating time during user observation	0 <= X The smaller is the better.	Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA

Table 8.4.2 Resource utilization metrics c) Transmission resource utilization

External resource utilisation metrics c) Transmission resource utilisation

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Maximum transmission utilisation	What is the absolute limit of transmissions required to fulfil a function?	Evaluate what is required for the system to reach a situation of maximum load. Emulate this condition. Run application and monitor result(s).	X = Amax / Rmax Amax = MAX(Ai), (for i = 1 to N) Rmax = required maximum number of transmission related error messages and failures MAX(Ai) = Maximum number of transmission related error messages and failures from 1st to i-th evaluation N= number of evaluations	0<= X The smaller is the better.	Absolute	Amax = Count Rmax = Count Ai = Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Media device utilisation balancing	What is the degree of synchronisation between different media over a set period of time?	Calibrate the test conditions. Emulate a condition whereby the system reaches a situation of maximum transmission load. Run the application and record the delay in the processing of different media types.	X = SyncTime/T SyncTime = Time devoted to a continuous resource T = required time period during which dissimilar media are expected to finish their tasks with synchronisation	The smaller is the better.	Ratio	SyncTime = Time T = Time X = Time/Time	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA
Mean occurrence of transmission error	What is the average number of transmission-related error messages and failures over a specified length of time and specified utilisation?	Calibrate the test condition. Emulate a condition whereby the system reaches a situation of maximum load. Run the application and record number of errors due to transmission failure and warnings.	X = Amean / Rmean Amean = ∑(Ai)/N Rmean = required mean number of transmission related error messages and failures Ai = Number of transmission related error messages and failures for i-th evaluation N = number of evaluations	0<= X The smaller is the better.	Absolute	Amean= Count Rmean= Count Ai= Count N= Count X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Mean of transmission error per time	How many transmission-related error messages were experienced over a set period of time and specified resource utilisation?	Calibrate the test conditions. Emulate a condition whereby the system reaches a situation of maximum transmission load. Run the application and record number of errors due to transmission failure and warnings.	X = A / T A = number of warning messages or system failures T = User operating time during user observation	0 <= X The smaller is the better.	Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA

Table 8.4.2 c) (*continued*)

External resource utilisation metrics	**c) Transmission resource utilisation**								
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Transmission capacity utilisation	Is software system capable of performing tasks within expected transmission capacity?	Execute concurrently specified tasks with multiple users, observe transmission capacity and compare specified one.	X = A / B A = transmission capacity B = specified transmission capacity which is designed to be used by the software during execution	0 <= X <= 1 The less than and nearer to the 1.0 is the better.	Absolute	A= Size B= Size X= Size / Size	Testing report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer SQA

FOOTNOTE

It is recommended to measure dynamically peaked value with multiple users.

Table 8.4.3 Efficiency compliance metrics

Efficiency compliance metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Efficiency Compliance	How compliant is the efficiency of the product to applicable regulations, standards and conventions?	Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance in the specification.	X = 1 - A / B (X: Ratio of satisfied compliance items relating to efficiency) A= Number of efficiency compliance items specified that have not been implemented during testing B= Total number of efficiency compliance items specified	0<= X <=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTE

It may be useful to collect several measured values along time, to analyse the trend of increasing satisfied compliance items and to determine whether they are fully satisfied or not.

8.5 Maintainability metrics

An external maintainability metric should be able to measure such attributes as the behaviour of the maintainer, user, or system including the software, when the software is maintained or modified during testing or maintenance.

8.5.1 Analysability metrics

An external analysability metric should be able to measure such attributes as the maintainer's or user's effort or spent of resources when trying to diagnose deficiencies or causes of failures, or for identifying parts to be modified.

8.5.2 Changeability metrics

An external changeability metric should be able to measure such attributes as the maintainer's or user's effort by measuring the behaviour of the maintainer, user or system including the software when trying to implement a specified modification.

8.5.3 Stability metrics

An external stability metric should be able to measure attributes related to unexpected behaviour of the system including the software when the software is tested or operated after modification.

8.5.4 Testability metrics

An external testability metric should be able to measure such attributes as the maintainer's or user's effort by measuring the behaviour of the maintainer, user or system including software when trying to test the modified or non-modified software.

8.5.5 Maintainability compliance metrics

An external maintainability compliance metric should be able to measure an attribute such as the number of functions or occurrences of compliance problems, where the software product fails to adhere to required standards, conventions or regulations relating to maintainability.

Table 8.5.1 Analysability metrics

External analysability metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Audit trail capability	Can user identify specific operation which caused failure? Can maintainer easily find specific operation which caused failure?	Observe behaviour of user or maintainer who is trying to resolve failures.	X= A / B A= Number of data actually recorded during operation B= Number of data planned to be recorded enough to monitor status of software during operation	0<=X The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Diagnostic function support	How capable are the diagnostic functions in supporting causal analysis? Can user identify the specific operation which caused failure? (User may be able to avoid falling into the same failure occurrence again with alternative operation.) Can maintainer easily find cause of failure?	Observe behaviour of user or maintainer who is trying to resolve failures using diagnostics functions.	X= A / B A= Number of failures which maintainer can diagnose (using the diagnostics function) to understand the cause-effect relationship B= Total number of registered failures	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Failure analysis capability	Can user identify specific operation which caused failure? Can maintainer easily find cause of failure?	Observe behaviour of user or maintainer who is trying to resolve failures.	X=1- A / B A= Number of failures of which causes are still not found B= Total number of registered failures	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer Operator

Table 8.5.1 (*continued*)

External analysability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Failure analysis efficiency	Can user efficiently analyse cause of failure? (User sometimes performs maintenance by setting parameters.) Can maintainer easily find cause of failure?	Observe behaviour of user or maintainer who is trying to resolve failures.	X= Sum(T) / N	0<=X The shorter is the better.	Ratio	T= Time	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTES

1 *It is recommended to measure maximal time of the worst case and time duration (bandwidth) to represent deviation.*

2 *It is recommended to exclude number of failures of which causes are not yet found when measurement is done. However, the ratio of such obscure failures should be also measured and presented together.*

3 *From the individual user's point of view, time is of concern, while effort may also be of concern from the maintainer's point of view. Therefore, person-hours may be used instead of time.*

Status monitoring capability	Can user identify specific operation which caused failure by getting monitored data during operation? Can maintainer easily find cause of failure by getting monitored data during operation?	Observe behaviour of user or maintainer who is trying to get monitored data recording status of software during operation.	X= 1- A / B A= Number of cases which maintainer (or user) failed to get monitor data B= Number of cases which maintainer (or user) attempted to get monitor data recording status of software during operation	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer Operator

Table 8.5.2 Changeability metrics

External changeability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Target audience
Change cycle efficiency	Can the user's problem be solved to his satisfaction within an acceptable time scale?	Monitor interaction between user and supplier. Record the time taken from the initial user's request to the resolution of problem.	Average Time : Tav = Sum(Tu) / N Tu= Trc - Tsn Tsn= Time at which user finished to send request for maintenance to supplier with problem report Trc= Time at which user received the revised version release (or status report) N= Number of revised versions	0<Tav The shorter is the better., except of the number of revised versions was large.	Ratio	Tu= Time Trc, Tsn = Time N= Count Tav= Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Maintainer Operator
Change implementation elapse time	Can the maintainer easily change the software to resolve the failure problem?	Observe the behaviour of the user and maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	Average Time : Tav = Sum(Tm) / N Tm=Tout - Tin Tout= Time at which the causes of failure are removed with changing the software (or status is reported back to user) Tin= Time at which the causes of failures are found out N= Number of registered and removed failures	0<Tav The shorter is the better, except of the number of failures was large.	Ratio	Tm= Time Tin, Tout = Time Tav= Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator

FOOTNOTES

1 *It is recommended to measure maximal time of the worst case and time bandwidth to represent deviation.*

2 *It is recommended to exclude the failures for which causes have not yet been found when the measurement is done. However, the ratio of such obscure failures should be also measured and presented together.*

3 *From the individual user's point of view, time is of concern, while effort may also be of concern from the maintainer's point of view. Therefore, person-hours may be used instead of time.*

Table 8.5.2 (*continued*)

External changeability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Modification complexity	Can the maintainer easily change the software to resolve problem?	Observe behaviour of maintainer who is trying to change the software. Otherwise, investigate problem resolution report or maintenance report and product description.	T = Sum (A / B) / N A= Work time spent to change B= Size of software change N= Number of changes	$0<T$ The shorter is the better or the required number of changes were excessive.	Ratio	A= Time B= Size N= Count T= Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTE
A size of software change may be changed executable statements of program code, number of changed items of requirements specification, or changed pages of document etc.

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Parameterised modifiability	Can the user or the maintainer easily change parameter to change software and resolve problems?	Observe behaviour of the user or the maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	X=1- A / B A= Number of cases which maintainer fails to change software by using parameter B= Number of cases which maintainer attempts to change software by using parameter	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator User
Software change control capability	Can the user easily identify revised versions? Can the maintainer easily change the software to resolve problems?	Observe the behaviour of user or maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	X= A / B A= Number of change log data actually recorded B= Number of change log data planned to be recorded enough to trace software changes	0<=X<= 1 The closer to 1.0 is the better or the closer to 0 the fewer changes have taken place.	Absolute	A= Count B= Count X= Count/ Count	User manual or specification Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 8.5.3 Stability metrics

External stability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Change success ratio	Can user operate software system without failures after maintenance? Can maintainer easily mitigate failures caused by maintenance side effects?	Observe behaviour of user or maintainer who is operating software system after maintenance. Count failures which user or maintainer encountered during operating software before and after maintenance. Otherwise, investigate problem resolution report, operation report or maintenance report.	X= Na / Ta Y = { (Na / Ta) / (Nb / Tb) } Na = Number of cases which user encounters failures during operation after software was changed Nb = Number of cases which user encounters failures during operation before software is changed Ta = Operation time during specified observation period after software is changed Tb = Operation time during specified observation period before software is changed	0<=X,Y The smaller and closer to 0 is the better.	Ratio	Na, Nb= Count Ta,Tb= Time X= Count/ Time Y=[(Count/ Time) / (Count/ Time)]	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTES

1 *X and Y imply " frequency of encountering failures after change" and "fluctuated frequency of encountering failures before/after change".*

2 *User may need specified period to determine side effects of software changes, when the revision-up of software is introduced for resolving problems.*

3 *It is recommend to compare this frequency before and after change.*

4 *If changed function is identified, it is recommended to determine whether encountered failures are detected in the changed function itself or in the other ones. The extent of impacts may be rated for each failure.*

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Modification impact localisation (Emerging failure after change)	Can user operate software system without failures after maintenance? Can maintainer easily mitigate failures caused by maintenance side effects?	Count failures occurrences after change, which are mutually chaining and affected by change.	X= A / N A= Number of failures emerged after failure is resolved by change during specified period N= Number of resolved failures	0<=X The smaller and closer to 0 is the better.	Absolute	A= Count N= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTE

X implies "chaining failure emerging per resolved failure". It is recommend to give precise measure by checking whether cause of current failure is attributed to change for previous failure resolution, as possible.

Table 8.5.4 Testability metrics

External testability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Availability of built-in test function	Can user and maintainer easily perform operational testing without additional test facility preparation?	Observe behaviour of user or maintainer who is testing software system after maintenance.	X= A / B A= Number of cases in which maintainer can use suitably built-in test function B= Number of cases of test opportunities	0 <= X <=1 The larger and the closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
FOOTNOTE *Examples of built-in test functions include simulation function, pre-check function for ready to use, etc.*									
Re-test efficiency	Can user and maintainer easily perform operational testing and determine whether the software is ready for operation or not?	Observe behaviour of user or maintainer who is testing software system after maintenance.	X= Sum(T) / N T= Time spent to test to make sure whether reported failure was resolved or not N= Number of resolved failures	0<X The smaller is the better.	Ratio	T= Time N= Count X= Time /Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
FOOTNOTE *X implies "average time (effort) to test after failure resolution". If failures are not resolved or fixed, exclude them and separately measure ratio of such failures.*									
Test restartability	Can user and maintainer easily perform operational testing with check points after maintenance?	Observe behaviour of user or maintainer who is testing software system after maintenance.	X = A / B A = Number of cases in which maintainer can pause and restart executing test run at desired points to check step by step B= Number of cases of pause of executing test run	0 <= X <=1 The larger and the closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 8.5.5 Maintainability compliance metrics

External maintainability compliance metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Maintainability compliance	How compliant is the maintainability of the product to applicable regulations, standards and conventions?	Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance in the specification.	$X = 1 - A / B$ A= Number of maintainability compliance items specified that have not been implemented during testing B= Total number of maintainability compliance items specified	$0 <= X <= 1$ The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTE
It may be useful to collect several measured values along time, to analyse the trend of increasing satisfied compliance items and to determine whether they are fully satisfied.

8.6 Portability metrics

An external portability metric should be able to measure such attributes as the behaviour of the operator or system during the porting activity.

8.6.1 Adaptability metrics

An external adaptability metric should be able to measure such attributes as the behaviour of the system or the user who is trying to adapt software to different specified environments. When a user has to apply an adaptation procedure other than previously provided by software for a specific adaptation need, user's effort required for adapting should be measured.

8.6.2 Installability metrics

An external installability metric should be able to measure such attributes as the behaviour of the system or the user who is trying to install the software in a user specific environment.

8.6.3 Co-existence metrics

An external co-existence metric should be able to measure such attributes as the behaviour of the system or the user who is trying to use the software with other independent software in a common environment sharing common resources.

8.6.4 Replaceability metrics

An external replaceability metric should be able to measure such attributes as the behaviour of the system or the user who is trying to use the software in place of other specified software in the environment of that software.

8.6.5 Portability compliance metrics

An external portability compliance metric should be able to measure such attributes as the number of functions with, or occurrences of compliance problems, where the software product fails to adhere to required standards, conventions or regulations relating to portability.

Table 8.6.1 Adaptability metrics

External adaptability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Adaptability of data structures	Can user or maintainer easily adapt software to data sets in new environment?	Observe user's or maintainer's behaviour when user is trying to adapt software to operation environment.	X = A / B A = The number of data which are operable and but are not observed due to incomplete operations caused by adaptation limitations B= The number of data which are expected to be operable in the environment to which the software is adapted	0<=X<=1 The larger and closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
FOOTNOTE *These data mainly include types of data such as data files, data tuples or databases to be adapted to different data volumes, data items or data structures. A and B of the formula are necessary to count the same types of data. Such an adaptation may be required when, for example, the business scope is extended.*									
Hardware environmental adaptability (adaptability to hardware devices and network facilities)	Can user or maintainer easily adapt software to environment? Is software system capable enough to adapt itself to operation environment?	Observe user's or maintainer's behaviour when user is trying to adapt software to operation environment.	X= 1 - A / B A= Number of operational functions of which tasks were not completed or not enough resulted to meet adequate levels during combined operating testing with environmental hardware B= Total number of functions which were tested	0<=X<=1 The larger is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
FOOTNOTE *It is recommended to conduct overload combination testing with hardware environmental configurations which may possibly be operationally combined in a variety of user operational environments.*									
Organisational environment adaptability **(Organisation adaptability to infrastructure of organisation)**	Can user or maintainer easily adapt software to environment? Is software system capable enough to adapt itself to the operational environment?	Observe user's or maintainer's behaviour when user is trying to adapt software to operation environment.	X= 1 - A / B A= Number of operated functions in which the tasks were not completed or not enough resulted to meet adequate levels during operational testing with user's business environment B= Total number of functions which were tested	0<=X<=1 The larger is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTES

1 *It is recommended to conduct testing which takes account of the varieties of combinations of infrastructure components of possible user's business environments.*

2 *"Organisational environment adaptability" is concerned with the environment of the business operation of the user's organisation. "System software environmental adaptability" is concerned with the environment of the technical operation of systems. Therefore, there is a clear distinction.*

Table 8.6.1 (*continued*)

External adaptability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Porting user friendliness	Can user or maintainer easily adapt software to environment?	Observe user's or maintainer's behaviour when user is trying to adapt software to operational environment.	T= Sum of user operating time spent to complete adaptation of the software to user's environment, when user attempt to install or change setup	0<T The shorter is the better.	Ratio	T=Time	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
FOOTNOTE *T implies "user effort required to adapt to user's environment". Person-hour may be used instead of time.*									
System software environmental adaptability (adaptability to OS, network software and co-operated application software)	Can user or maintainer easily adapt software to environment? Is software system capable enough to adapt itself to operation environment?	Observe user's or maintainer's behaviour when user is trying to adapt software to operation environment.	X= 1 - A / B A= Number of operational functions of which tasks were not completed or were not enough resulted to meet adequate level during combined operating testing with operating system software or concurrent application software B= Total number of functions which were tested	0<=X<=1 The larger is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTES

1 *It is recommended to conduct overload combination testing with operating system softwares or concurrent application softwares which are possibly combined operated in a variety of user operational environments.*

2 *"Organisational environment adaptability" is concerned with the environment for business operation of user's organisation. "System software environmental adaptability" is concerned with the environment for technical operations on systems. Therefore, there is a clear distinction.*

Table 8.6.2 Installability metrics

External installability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Ease of installation	Can user or maintainer easily install software to operation environment?	Observe user's or maintainer's behaviour when user is trying to install software to operation environment.	X = A / B A = Number of cases which a user succeeded to in changing the install operation for his/her convenience B = Total number of cases which a user attempted to change the install operation for his/her convenience	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTES

1 *This metric is suggested as experimental use.*

2 *When time basis metric is required, spent time for installation may be measurable.*

Table 8.6.2 (*continued*)

External installability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Ease of Setup Retry	Can user or maintainer easily re-try set-up installation of software?	Observe user's or maintainer's behaviour when user is trying to re-try set-up installation of software.	X = 1 - A / B A = Number of cases in which user fails in re-trying set-up during set-up operation B = Total number of cases in which user attempt to re-try setup during set-up operation	0<=X<= 1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTE

This metric is suggested as experimental use.

FOOTNOTES

The following complementary metrics may be used.

1 *Effortless installation User's manual actions for installation X = A A= The number of user's manual actions needed for installation 0<X he smaller is the better.*

2 *Installation ease*
Installation supporting level X = A A is rated with, for example:
- Only executing installation program where nothing more is needed (excellent);
- Instructional guide for installation (good);
- Source code of program needs modification for installation (poor).
X= Direct Interpretation of measured value

3 *Operational installation effort reduction*
User Install Operation Procedure Reduction Ratio X = 1- A / B
A = Number of install operation procedures which a user had to do after procedure reduction
B = Number of install operation procedures normally
0<= X <=1 The closer to 1.0 is the better.

4 *Ease of user's manual install operation*
Easiness level of user's manual install operation X = Score of easiness level of user's manual operation Examples of easiness level are following:
[very easy] requiring only user's starting of install or set-up functions and then observing installation;
[easy] requiring only user's answering of question from install or set-up functions;
[not easy] requiring user's looking up parameters from tables or filling-in boxes;
[complicated] requiring user's searching parameter files, looking up parameters from files to be changed and writing them.
X= Direct Interpretation of measured value

Table 8.6.3 Co-existence metrics

External co-existence metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Available co-existence	How often user encounters any constraints or unexpected failures when operating concurrently with other software?	Use evaluated software concurrently with other software which user often uses.	X = A / T A = Number of any constraints or unexpected failures which user encounter during operating concurrently with other software T = Time duration of concurrently operating other software	0<=X The closer to 0 is the better.	Ratio	A= Count T= Time X= Count/ Time	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer SQA Operator

Table 8.6.4 Replaceability metrics

External replaceability metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Continued use of data	Can user or maintainer easily continue to use the same data after replacing this software to previous one? Is software system migration going on successfully?	Observe user's or maintainer's behaviour when user is replacing software to previous one.	X = A / B A = number of data which are used in other software to be replaced and are confirmed that they are able to be continuously used B = number of data which are used in other software to be replaced and planned to be continuously reusable	0<= X <=1 The larger is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTE
This metric can be applied to both cases of replacing an entirely different software and a different version of the same software series to previous one.

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Function inclusiveness	Can user or maintainer easily continue to use similar functions after replacing this software to previous one? Is software system migration going on successfully?	Observe user's or maintainer's behaviour when user is replacing software to previous one.	X = A / B A = number of functions which produce similar results as previously produced and where changes have not been required B = number of tested functions which are similar to functions provided by another software to be replaced	0<= X <=1 The larger is the better.	Absolute	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

FOOTNOTE
This metric can be applied to both cases of replacing an entirely different software and a different version of the same software series to previous one.

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
User support functional consistency	How consistent are the new components with existing user interface?	Observe the behaviour of the user and ask the opinion.	X= 1 - A1 / A2 A= Number of new functions which user found unacceptably inconsistent with the user's expectation B= Number of new functions	0<=X Larger is better.	Absolute	A1= Count A2= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.3 Quality Assurance	User User interface designer Maintainer Developer Tester SQA

FOOTNOTES

1 *The case that a different software is introduced to replace for a previous software, a new different software can be identified as a current version.*

2 *In case that the pattern of interaction is changed to improve user interface in a new version,, it is suggested to observe user's behaviour and to count the number of cases which the user fails to access functions caused by unacceptable conformity against user's expectation derived from previous version.*

Table 8.6.5 Portability compliance metrics

External portability compliance metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measure-ment	ISO/IEC 12207 SLCP Reference	Target audience
Portability compliance	How compliant is the portability ofthe product to applicable regulations, standards and conventions ?	Count the number of items requiring compliance that have been met and compare with the number of items requiring compliance in the specification.	X = 1- A / B A= Number of portability compliance items specified that have not been implemented during testing B= Total number of portability compliance items specified	0<= X <=1 The closer to 1.0 is the better.	Absolute	A= Count B= Count X= Count/ Count	Product description (User manual or Specification) of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

FOOTNOTE

It may be useful to collect several measured values along time, analyse the trend of increasing satisfied compliance items, and determine whether they are fully satisfied.

Annex A
(informative)

Considerations When Using Metrics

A.1 Interpretation of measures

A.1.1 Potential differences between test and operational contexts of use

When planning the use of metrics or interpreting measures it is important to have a clear understanding of the intended context of use of the software, and any potential differences between the test and operational contexts of use. For example, the “time required to learn operation” measure is often different between skilled operators and unskilled operators in similar software systems. Examples of potential differences are given below.

a) Differences between testing environment and the operational environment

Are there any significant differences between the testing environment and the operational execution in user environment?

The following are examples:

- testing with higher / comparable / lower performance of CPU of operational computer;
- testing with higher / comparable / lower performance of operational network and communication;
- testing with higher / comparable / lower performance of operational operating system;
- testing with higher / comparable / lower performance of operational user interface.

b) Differences between testing execution and actual operational execution

Are there any significant differences between the testing execution and operational execution in user environment?

The following are examples:

- coverage of functionality in test environment;
- test case sampling ratio;
- automated testing of real time transactions;
- stress loads;
- 24 hour 7 days a week (non stop) operation;
- appropriateness of data for testing of exceptions and errors;
- periodical processing;
- resource utilisation;
- levels of interruption;
- production pressures;
- distractions.

c) User profile under observation

Are there any significant differences between test user profiles and operational user profiles?

The following are examples:

- mix of type of users;
- user skill levels;
- specialist users or average users;
- limited user group or public users.

A.1.2 Issues affecting validity of results

The following issues may affect the validity of the data that is collected.

(a) procedures for collecting evaluation results:

- automatically with tools or facilities / manually collected / questionnaires or interviews;

(b) source of evaluation results

- developers' self reports / reviewers' report / evaluator's report;

(c) results data validation

- developers' self check / inspection by independent evaluators.

A.1.3 Balance of measurement resources

Is the balance of measures used at each stage appropriate for the evaluation purpose?

It is important to balance the effort used to apply an appropriate range of metrics for internal, external and quality in use measures.

A.1.4 Correctness of specification

Are there significant differences between the software specification and the real operational needs?

Measurements taken during software product evaluation at different stages are compared against product specifications. Therefore, it is very important to ensure by verification and validation that the product specifications used for evaluation reflect the actual and real needs in operation.

A.2 Validation of Metrics

A.2.1 Desirable Properties for Metrics

To obtain valid results from a quality evaluation, the metrics should have the properties stated below. If a metric does not have these properties, the metric description should explain the associated constraint on its validity and, as far as possible, how that situation can be handled.

a) **Reliability (of metric):** Reliability is associated with random error. A metric is free of random error if random variations do not affect the results of the metric.

b) **Repeatability (of metric):** repeated use of the metric for the same product using the same evaluation specification (including the same environment), type of users, and environment by the same evaluators, should produce the same results within appropriate tolerances. The appropriate tolerances should include such things as fatigue, and learning effect.

c) **Reproducibility (of metric):** use of the metric for the same product using the same evaluation specification (including the same environment), type of users, and environment by different evaluators, should produce the same results within appropriate tolerances.

NOTE 1 It is recommended to use statistical analysis to measure the variability of the results.

d) **Availability (of metric):** The metric should clearly indicate the conditions (e.g. presence of specific attributes) which constrain its usage.

e) **Indicativeness (of metric):** Capability of the metric to identify parts or items of the software which should be improved, given the measured results compared to the expected ones.

NOTE 2 The selected or proposed metric should provide documented evidence of the availability of the metric for use, unlike those requiring project inspection only.

f) **Correctness (of measure):** The metric should have the following properties:

1) Objectivity (of measure): the metric results and its data input should be factual: i.e., not influenced by the feelings or the opinions of the evaluator, test users, etc. (except for satisfaction or attractiveness metrics where user feelings and opinions are being measured).
2) Impartiality (of measure): the measurement should not be biased towards any particular result.
3) Sufficient precision (of measure): Precision is determined by the design of the metric, and particularly by the choice of the material definition used as the basis for the metric. The metric user will describe the precision and the sensitivity of the metric.

g) **Meaningfulness (of measure):** the measurement should produce meaningful results about the software behaviour or quality characteristics.

The metric should also be cost effective: that is, more costly metrics should provide higher value results.

A.2.2 Demonstrating the Validity of Metrics

The users of metrics should identify the methods for demonstrating the validity of metrics, as shown below:

(a) Correlation

The variation in the quality characteristics values (the measures of principal metrics in operational use) explained by the variation in the metric values, is given by the square of the linear coefficient.

An evaluator can predict quality characteristics without measuring them directly by using correlated metrics.

(b) Tracking

If a metric M is directly related to a quality characteristics value Q (the measures of principal metrics in operational use), for a given product or process, then a change value Q(T1) to Q(T2), would be accompanied by a change metric value from M(T1) to M(T2), in the same direction (for example, if Q increases, M increases).

An evaluator can detect movement of quality characteristics along a time period without measuring directly by using those metrics which have tracking ability.

(c) Consistency

If quality characteristics values (the measures of principal metrics in operational use) Q1, Q2,..., Qn, corresponding to products or processes 1, 2,..., n, have the relationship Q1 > Q2 > ...> Qn, then the corresponding metric values would have the relationship M1 > M2 > ...> Mn.

An evaluator can notice exceptional and error prone components of software by using those metrics which have consistency ability.

(d) Predictability

If a metric is used at time T1 to predict a quality characteristic value Q (the measures of principal metrics in operational use) at T2, prediction error, which is {(predicted Q(T2) - actual Q(T2)) / actual Q(T2)}, would be within allowed prediction error range.

An evaluator can predict the movement of quality characteristics in the future by using these metrics, which measure predictability.

(e) Discriminative

A metric would be able to discriminate between high and low quality software.

An evaluator can categorize software components and rate quality characteristics values by using those metrics which have discriminative ability.

A.3 Use of metrics for estimation (judgement) and prediction (forecast)

Estimation and prediction of the quality characteristics of the software product at the earlier stages are two of the most rewarding uses of metrics.

A.3.1 Quality characteristics prediction by current data

(a) Prediction by regression analysis

When predicting the future value (measure) of the same characteristic (attribute) by using the current value (data) of it (the attribute), a regression analysis is useful based on a set of data that is observed in a sufficient period of time.

For example, the value of MTBF (Mean Time Between Failures) that is obtained during the testing stage (activities) can be used to estimate the MTBF in operation stage.

(b) Prediction by correlation analysis

When predicting the future value (measure) of a characteristic (attribute) by using the current measured values of a different attribute, a correlation analysis is useful using a validated function which shows the correlation.

For example, the complexity of modules during coding stage may be used to predict time or effort required for program modification and test during maintenance process.

A.3.2 Current quality characteristics estimation on current facts

(a) Estimation by correlation analysis

When estimating the current values of an attribute which are directly unmeasurable, or if there is any other measure that has strong correlation with the target measure, a correlation analysis is useful.

For example, because the number of remaining faults in a software product is not measurable, it may be estimated by using the number and trend of detected faults.

Those metrics which are used for predicting the attributes that are not directly measurable should be estimated as explained below:

- Using models for predicting the attribute;
- Using formula for predicting the attribute;
- Using basis of experience for predicting the attribute;
- Using justification for predicting the attribute.

Those metrics which are used for predicting the attributes that are not directly measurable may be validated as explained below:

- Identify measures of attributes which are to be predicted;
- Identify the metrics which will be used for prediction;
- Perform a statistical analysis based validation;
- Document the results;
- Repeat the above periodically.

A.4 Detecting deviations and anomalies in quality problem prone components

The following quality control tools may be used to analyse deviations and anomalies in software product components:

(a) process charts (functional modules of software)

(b) Pareto analysis and diagrams

(c) histograms and scatter diagrams

(d) run diagrams, correlation diagrams and stratification

(e) Ishikawa (Fishbone) diagrams

(f) statistical process control (functional modules of software)

(g) check sheets

The above tools can be used to identify quality issues from data obtained by applying the metrics.

A.5 Displaying measurement results

(a) Displaying quality characteristics evaluation results

The following graphical presentations are useful to display quality evaluation results for each of the quality characteristic and subcharacteristic.

Radar chart; Bar chart numbered histogram, multi-variates chart, Importance Performance Matrix, etc.

(b) Displaying measures

There are useful graphical presentations such as Pareto chart, trend charts, histograms, correlation charts, etc.

Annex B
(informative)

Use of Quality in Use, External & Internal Metrics (Framework Example)

B.1 Introduction

This framework example is a high level description of how the ISO/IEC 9126 Quality model and related metrics may be used during the software development and implementation to achieve a quality product that meets user's specified requirements. The concepts shown in this example may be implemented in various forms of customization to suit the individual, organization or project. The example uses the key life cycle processes from ISO/IEC 12207 as a reference to the traditional software development life cycle and quality evaluation process steps from ISO/IEC 14598-3 as a reference to the traditional Software Product Quality evaluation process. The concepts can be mapped onto other models of software life cycles if the user so wishes as long as the underlying concepts are understood.

B.2 Overview of Development and Quality Process

Table B.1 depicts an example model that links the Software Development life cycle process activities (activity 1 to activity 8) to their key deliverables and the relevant reference models for measuring quality of the deliverables (i.e., Quality in Use, External Quality, or Internal Quality).

Row 1 describes the software development life cycle process activities. (This may be customized to suit individual needs). Row 2 describes whether an actual measure or a prediction is possible for the category of measures (i.e., Quality in Use, External Quality, or Internal Quality). Row 3 describes the key deliverable that may be measured for Quality and Row 4 describes the metrics that may be applied on each deliverable at each process activity.

Table B.1 Quality Measurement Model

	Activity 1	Activity 2	Activity 3	Activity 4	Activity 5	Activity 6	Activity 7	Activity 8
Phase	Requirement analysis (Software and systems)	Architectural design (Software and systems)	Software detailed design	Software coding and testing	Software integration and software qualification testing	System integration and system qualification testing	Software installation	Software acceptance support
9126 series model reference	Required user quality, Required internal quality, Required external quality	Predicted quality in use, Predicted external quality, Measured internal quality	Predicted quality in use, Predicted external quality, Measured internal quality	Predicted quality in use, Measured external quality, Predicted external quality, Measured internal quality	Predicted quality in use, Measured external quality, Predicted external quality, Measured internal quality	Predicted quality in use, Measured external quality, Measured internal quality	Predicted quality in use, Measured external quality, Measured internal quality	Measured quality in use, Measured external quality, Measured internal quality
Key deliverables of activity	User quality requirements (specified), External quality requirements (specified), Internal quality requirements (specified)	Architecture design of Software / system	Software detailed design	Software code, Test results	Software product, Test results	Integrated system, Test results	Installed system	Delivered software product
Metrics used to measure	Internal metrics (External metrics may be applied to validate specifications)	Internal metrics	Internal metrics	Internal metrics External metrics	Internal metrics External metrics	Internal metrics External metrics	Internal metrics External metrics	Quality in use metrics Internal metrics External metrics

B.3 Quality Approach Steps

B.3.1 General

Evaluation of the Quality during the development cycle is divided into the following steps. Step 1 has to be completed during the Requirement Analysis activity. Steps 2 to 5 have to be repeated during each process activity defined above.

B.3.2 Step #1 Quality requirements identification

For each of the Quality characteristics and subcharacteristics defined in the Quality model determine the User Needs weights using the two examples in Table B.2 for each category of the measurement. (Quality in Use, External and Internal Quality). Assigning relative weights will allow the evaluators to focus their efforts on the most important subcharacteristics.

Table B.2 User Needs Characteristics & Weights

(a)

Quality in Use		
	CHARACTERISTIC	WEIGHT (High/Medium/Low)
	Effectiveness	H
	Productivity	H
	Safety	L
	Satisfaction	M

(b)

External & Internal Quality		
CHARACTERISTIC	SUBCHARACTERISTIC	WEIGHT (High/Medium/Low)
Functionality	Suitability	H
	Accuracy	H
	Interoperability	L
	Security	L
	Compliance	M
Reliability	Maturity (hardware/software/data)	L
	Fault tolerance	L
	Recoverability (data, process, technology)	H
	Compliance	H
Usability	Understandability	M
	Learnability	L
	Operability	H
	Attractiveness	M
	Compliance	H
Efficiency	Time behaviour	H
	Resource utilization	H
	Compliance	H
Maintainability	Analyzability	H
	Changeability	M
	Stability	L
	Testability	M
	Compliance	H

Table B.2 b) (*continued*)

Portability	Adaptability	H
	Installability	L
	Co-existence	H
	Replaceability	M
	Compliance	H

NOTE Weights can be expressed in the High/Medium/Low manner or using the ordinal type scale in the range 1-9 (e.g.: 1-3 = low, 4-6 = medium, 7-9 = high).

B.3.3 Step #2 Specification of the evaluation

This step is applied during every development process activity.

For each of the Quality subcharacteristics defined in the Quality model identify the metrics to be applied and the required levels to achieve the User Needs set in Step 1 and record as shown in the example in Table B.3.

Basic input and directions for the content formulation can be obtained from the example in Table B.1 that explains what can be measured at this stage of the development cycle.

NOTE It is possible, that some of the rows of the tables would be empty during the specific activities of the development cycle, because it would not be possible to measure all of the subcharacteristics early in the development process.

Table B.3 Quality measurement tables

(a)

Quality in Use Measurement Category				
	CHARACTERISTIC	METRICS	REQUIRED LEVEL	ASSESSMENT ACTUAL RESULT
	Effectiveness			
	Productivity			
	Safety			
	Satisfaction			

(b)

External Quality Measurement Category				
CHARACTERISTIC	SUBCHARACTERISTIC	METRICS	REQUIRED LEVEL	ASSESSMENT ACTUAL RESULT
Functionality	Suitability			
	Accuracy			
	Interoperability			
	Security			
	Compliance			
Reliability	Maturity (hardware/software/data)			
	Fault tolerance			
	Recoverability (data, process, technology)			
	Compliance			

Usability	Understandability			
	Learnability			
	Operability			
	Attractiveness			
	Compliance			
Efficiency	Time behaviour			
	Resource utilisation			
	Compliance			
Maintainability	Analyzability			
	Changeability			
	Stability			
	Testability			
	Compliance			
Portability	Adaptability			
	Instability			
	Co-existence			
	Replaceability			
	Compliance			

(c)

Internal Quality Measurement Category				
CHARACTERISTIC	SUBCHARACTERISTIC	METRICS	REQUIRED LEVEL	ASSESSMENT ACTUAL RESULT
Functionality	Suitability			
	Accuracy			
	Interoperability			
	Security			
	Compliance			
Reliability	Maturity (hardware/software/data)			
	Fault tolerance			
	Recoverability (data, process, technology)			
	Compliance			
Usability	Understandability			
	Learnability			
	Operability			
	Attractiveness			
	Compliance			
Efficiency	Time behaviour			
	Resource utilisation			
	Compliance			
Maintainability	Analyzability			
	Changeability			
	Stability			
	Testability			
	Compliance			
Portability	Adaptability			
	Instability			
	Co-existence			
	Replaceability			
	Compliance			

B.3.4 Step #3 Design of the evaluation

This step is applied during every development process activity.

Develop a measurement plan (similar to example in Table B.4) containing the deliverables that are used as input to the measurement process and the metrics to be applied.

Table B.4 Measurement plan

SUBCHARACTERISTIC	DELIVERABLES TO BE EVALUATED	INTERNAL METRICS TO BE APPLIED	EXTERNAL METRICS TO BE APPLIED	QUALITY IN USE METRICS TO BE APPLIED
1. Suitability	1. 2. 3.	1. 2. 3.	1. 2. 3.	(Not Applicable)
2. Satisfaction	1. 2. 3.	(Not Applicable)	(Not Applicable)	1. 2. 3.
3.				
4.				
5.				
6.				

B.3.5 Step #4 Execution of the evaluation

This step is applied during every development process activity.

Execute the evaluation plan and complete the column as shown in the examples in Table B.3. ISO/IEC 14598 series of standards should be used as a guidance for planning and executing the measurement process.

B.3.6 Step #5 Feedback to the organization

This step is applied during every development process activity.

Once all measurements have been completed map the results into Table B.1 and document conclusions in the form of a report. Also identify specific areas where quality improvements are required for the product to meet the user needs.

Annex C
(informative)

Detailed explanation of metric scale types and measurement types

C.1 Metric Scale Types

One of the following measurement metric scale types should be identified for each measure, when a user of metrics has the result of a measurement and uses the measure for calculation or comparison. The average, ratio or difference values may have no meaning for some measures. Metric scale types are: Nominal scale, Ordinal scale, Intervals scale, Ratio scale, and Absolute scale. A scale should always be defined as M'=F(M), where F is the admissible function. Also the description of each measurement scale type contains a description of the admissible function (if M is a metric then M'=F(M) is also a metric).

(a) Nominal Scale

M'=F(M) where F is any one-to-one mapping.

This includes classification, for example, software fault types (data, control, other). An average has a meaning only if it is calculated with frequency of the same type. A ratio has a meaning only when it is calculated with frequency of each mapped type. Therefore, the ratio and average may be used to represent a difference in frequency of only the same type between early and later cases or two similar cases. Otherwise, they may be used to mutually compare the frequency of each other type respectively.

Examples: Town transport line identification number, Compiler error message identification number

Meaningful statements are Numbers of different categories only.

(b) Ordinal Scale

M'=F(M) where F is any monotonic increasing mapping that is, M(x)>=M(y) implies M'(x)>=M'(y).

This includes ordering, for example, software failure by severity (negligible, marginal, critical, catastrophic). An average has a meaning only if it is calculated with frequency of the same mapped order. A ratio has a meaning only when it is calculated with the frequency of each mapped order. Therefore, the ratio and the average may be used to represent a difference in frequency of only the same order between early and later cases or two similar cases. Otherwise, they may be used to compare mutually the frequency of each order.

Examples: School exam.result (excellent, good, acceptable, not acceptable),

Meaningful statements: Each will depend on its position in the order, for example the median.

(c) Interval Scale

M'=aM+b (a>0)

This includes ordered rating scales where the difference between two measures has an empirical meaning. However the ratio of two measures in an interval scale may not have the same empirical meaning.

Examples: Temperature (Celsius, Fahrenheit, Kelvin), difference between the actual computation time and the time predicted

Meaningful statements: An arithmetic average and anything that depends on an order

(d) Ratio Scale

M'=aM (a>0)

This includes ordered rating scales, where the difference between two measures and also the proportion of two measures have the same empirical meaning. An average and a ratio have meaning respectively and they give actual meaning to the values.

Examples: Length, Weight, Time, Size, Count

Meaningful statements: Geometrical mean, Percentage

(e) Absolute Scale

M'=M they can be measured only in one way.

Any statement relating to measures is meaningful. For example the result of dividing one ratio scale type measure by another ratio scale type measure where the unit of measurement is the same is absolute. An absolute scale type measurement is in fact one without any unit.

Example: Number of lines of code with comments divided by the total lines of code

Meaningful statements: Everything

C.2 Measurement Types

C.2.0 General

In order to design a procedure for collecting data, interpreting fair meanings, and normalizing measures for comparison, a user of metrics should identify and take account of the measure type of measurement employed by a metric.

C.2.1 Size Measure Type

C.2.1.0 General

A measure of this type represents a particular size of software according to what it claims to measure within its definition.

NOTE Software may have many representations of size (like any entity can be measured in more than one dimension - mass, volume, surface area etc.).

Normalizing other measures with a size measure can give comparable values in terms of units of size. The size measures described below can be used for software quality measurement.

C.2.1.1 Functional Size Type

Functional size is an example of one type of size (one dimension) that software may have. Any one instance of software may have more than one functional size depending on, for example:

(a) the purpose for measuring the software size (It influences the scope of the software included in the measurement);

(b) the particular functional sizing method used (It will change the units and scale).

The definition of the concepts and process for applying a functional size measurement method (FSM Method) is provided by the standard ISO/IEC 14143-1.

In order to use functional size for normalization it is necessary to ensure that the same functional sizing method is used and that the different software being compared have been measured for the same purpose and consequently have a comparable scope.

Although the following often claim that they represent functional sizes, it is not guaranteed they are equivalent to the functional size obtained from applying a FSM Method compliant with ISO/IEC 14143-1. However, they are widely used in software development:

1. **number of spread sheets;**

2. **number of screens;**

3. **number of files or data sets which are processed;**

4. **number of itemized functional requirements described in user requirements specifications.**

C.2.1.2 Program size type

In this clause, the term 'programming' represents the expressions that when executed result in actions, and the term 'language' represents the type of expression used.

1. Source program size

The programming language should be explained and it should be provided how the non executable statements, such as comment lines, are treated. The following measures are commonly used.

Non-comment source statements (NCSS) include executable statements and data declaration statements with logical source statements.

NOTE 1 New program size

A developer may use newly developed program size to represent development and maintenance work product size.

NOTE 2 Changed program size

A developer may use changed program size to represent size of software containing modified components.

NOTE 3 Computed program size

Example of computed program size formula is new lines of code + 0.2 x lines of code in modified components (NASA Goddard).

It may be necessary to distinguish a type of statements of source code into more detail as follows:

i. Statement Type

Logical Source Statement (LSS). The LSS measures the number of software instructions. The statements are irrespective of their relationship to lines and independent of the physical format in which they appear.

Physical Source Statement (PSS). The PSS measures the number of software source lines of code.

ii. Statement attribute

Executable statements;

Data declaration statements;

Compiler directive statements;

Comment source statements.

iii. Origin

Modified source statements;

Added source statements;

Removed source statements;

- Newly Developed source statements: (= added source statements + modified source statements);
- Reused source statements: (= original - modified - removed source statements);

2. Program word count size

The measurement may be computed in the following manner using the Halstead's measure:

Program vocabulary = n1+n2; Observed program length = N1+N2, where:

- n1: Is the number of distinct operator words which are prepared and reserved by the program language in a program source code;
- n2: Is the number of distinct operand words which are defined by the programmer in a program source code;
- N1: Is the number of occurrences of distinct operators in a program source code;
- N2: Is the number of occurrences of distinct operands in a program source code.

3. Number of modules

The measurement is counting the number of independently executable objects such as modules of a program.

C.2.1.3 Utilized resource measure type

This type identifies resources utilized by the operation of the software being evaluated. Examples are:

(a) Amount of memory, for example, amount of disk or memory occupied temporally or permanently during the software execution;

(b) I/O load, for example, amount of traffic of communication data (meaningful for backup tools on a network);

(c) CPU load, for example, percentage of occupied CPU instruction sets per second (This measure type is meaningful for measuring CPU utilization and efficiency of process distribution in multi-thread software running on concurrent/parallel systems);

(d) Files and data records, for example, length in bytes of files or records;

(e) Documents, for example, number of document pages.

It may be important to take note of peak (maximal), minimum and average values, as well as periods of time and number of observations done.

C.2.1.4 Specified operating procedure step type

This type identifies static steps of procedures which are specified in a human-interface design specification or a user manual.

The measured value may differ depending on what kinds of description are used for measurement, such as a diagram or a text representing user operating procedures.

C.2.2 Time measure type

C.2.2.0 General

The user of metrics of time measure type should record time periods, how many sites were examined and how many users took part in the measurements.

There are many ways in which time can be measured as a unit, as the following examples show.

(a) Real time unit

This is a physical time: i.e. second, minute, or hour. This unit is usually used for describing task processing time of real time software.

(b) Computer machinery time unit

This is computer processor's clock time: i.e. second, minute, or hour of CPU time.

(c) Official scheduled time unit

This includes working hours, calendar days, months or years.

(d) Component time unit

When there are multiple sites, component time identifies individual site and it is an accumulation of individual time of each site. This unit is usually used for describing component reliability, for example, component failure rate.

(e) System time unit

When there are multiple sites, system time does not identify individual sites but identifies all the sites running, as a whole in one system. This unit is usually used for describing system reliability, for example, system failure rate.

C.2.2.1 System operation time type

System operation time type provides a basis for measuring software availability. This is mainly used for reliability evaluation. It should be identified whether the software is under discontinuous operation or continuous operation. If the software operates discontinuously, it should be assured that the time measurement is done on the periods the software is active (this is obviously extended to continuous operation).

(a) Elapsed time

When the use of software is constant, for example in systems operating for the same length of time each week.

(b) Machine powered-on time

For real time, embedded or operating system software that is in full use the whole time the system is operational.

(c) Normalized machine time

As in “machine powered-on time”, but pooling data from several machines of different “powered-on-time” and applying a correction factor.

C.2.2.2 Execution time type

Execution time type is the time which is needed to execute software to complete a specified task. The distribution of several attempts should be analysed and mean, deviation or maximal values should be computed. The execution under the specific conditions, particularly overloaded condition, should be examined. Execution time type is mainly used for efficiency evaluation.

C.2.2.3 User time type

User time type is measured upon time periods spent by individual users on completing tasks by using operations of the software. Some examples are:

(a) Session time

Measured between start and end of a session. Useful, as example, for drawing behaviour of users of a home banking system. For an interactive program where idling time is of no interest or where interactive usability problems only are to be studied.

(b) Task time

Time spent by an individual user to accomplish a task by using operations of the software on each attempt. The start and end points of the measurement should be well defined.

(c) User time

Time spent by an individual user using the software from time started at a point in time. (Approximately, it is how many hours or days user uses the software from beginning).

C.2.2.4 Effort type

Effort type is the productive time associated with a specific project task.

(a) Individual effort

This is the productive time which is needed for the individual person who is a developer, maintainer, or operator to work to complete a specified task. Individual effort assumes only a certain number of productive hours per day.

(b) Task effort

Task effort is an accumulated value of all the individual project personnel: developer, maintainer, operator, user or others who worked to complete a specified task.

C.2.2.5 Time interval of events type

This measure type is the time interval between one event and the next one during an observation period. The frequency of an observation time period may be used in place of this measure. This is typically used for describing the time between failures occurring successively.

C.2.3 Count measure type

If attributes of documents of the software product are counted, they are static count types. If events or human actions are counted, they are kinetic count types.

C.2.3.1 Number of detected fault type

The measurement counts the detected faults during reviewing, testing, correcting, operating or maintaining. Severity levels may be used to categorize them to take into account the impact of the fault.

C.2.3.2 Program structural complexity number type

The measurement counts the program structural complexity. Examples are the number of distinct paths or the McCabe's cyclomatic number.

C.2.3.3 Number of detected inconsistency type

This measure counts the detected inconsistent items which are prepared for the investigation.

(a) Number of failed conforming items

Examples:

- Conformance to specified items of requirements specifications;
- Conformance to rule, regulation, or standard;
- Conformance to protocols, data formats, media formats, character codes.

(b) Number of failed instances of user expectation

The measurement is to count satisfied/unsatisfied list items, which describe gaps between user's reasonable expectation and software product performance.

The measurement uses questionnaires to be answered by testers, customers, operators, or end users on what deficiencies were discovered.

The following are examples:

- Function available or not;
- Function effectively operable or not;
- Function operable to user's specific intended use or not;
- Function is expected, needed or not needed.

C.2.3.4 Number of changes type

This type identifies software configuration items which are detected to have been changed. An example is the number of changed lines of source code.

C.2.3.5 Number of detected failures type

The measurement counts the detected number of failures during product development, testing, operating or maintenance. Severity levels may be used to categorize them to take into account the impact of the failure.

C.2.3.6 Number of attempts (trial) type

This measure counts the number of attempts at correcting the defect or fault. For example, during reviews, testing, and maintenance.

C.2.3.7 Stroke of human operating procedure type

This measure counts the number of strokes of user human action as kinetic steps of a procedure when a user is interactively operating the software. This measure quantifies the ergonomic usability as well as the effort to use. Therefore, this is used in usability measurement. Examples are number of strokes to perform a task, number of eye movements, etc.

C.2.3.8 Score type

This type identifies the score or the result of an arithmetic calculation. Score may include counting or calculation of weights checked on/off on checklists. Examples: Score of checklist; score of questionnaire; Delphi method; etc.

Annex D
(informative)

Term(s)

D.1 Definitions

Definitions are from ISO/IEC 14598-1 and ISO/IEC 9126-1 unless otherwise indicated.

D.1.1 Quality

External quality: The extent to which a product satisfies stated and implied needs when used under specified conditions.

Internal quality: 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 “attribute” is used (rather than the term “characteristic” used in 3.1.3) as the term “characteristic” is used in a more specific sense in ISO/IEC 9126 series.

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

NOTE 2 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.

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 3 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 4 The definition of quality in use in ISO/IEC 14598-1 does not currently include the new characteristic of “safety”.

Quality model: The set of characteristics and the relationships between them, which provide the basis for specifying quality requirements and evaluating quality.

D.1.2 Software and user

Software: All or part of the programs, procedures, rules, and associated documentation of an information processing system. (ISO/IEC 2382-1:1993)

NOTE 1 Software is an intellectual creation that is independent of the medium on which it is recorded.

Software product: The set of computer programs, procedures, and possibly associated documentation and data designated for delivery to a user. [ISO/IEC 12207]

NOTE 2 Products include intermediate products, and products intended for users such as developers and maintainers.

User: An individual that uses the software product to perform a specific function.

NOTE 3 Users may include operators, recipients of the results of the software, or developers or maintainers of software.

D.1.3 Measurement

Attribute: A measurable physical or abstract property of an entity.

Direct measure: A measure of an attribute that does not depend upon a measure of any other attribute.

External measure: 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.

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

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

Indicator: A measure that can be used to estimate or predict another measure.

NOTE 4 The measure may be of the same or a different characteristic.

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

Indirect measure: A measure of an attribute that is derived from measures of one or more other attributes.

NOTE 6 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.

Internal measure: A measure derived from the product itself, either direct or indirect; it is not derived from measures of the behaviour of the system of which it is a part.

NOTE 7 Lines of code, complexity, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself.

Measure (noun): The number or category assigned to an attribute of an entity by making a measurement.

Measure (verb): Make a measurement.

Measurement: The process of assigning a number or category to an entity to describe an attribute of that entity.

NOTE 8 “Category” is used to denote qualitative measures of attributes. For example, some important attributes of software products, e.g. the language of a source program (ADA, C, COBOL, etc.) are qualitative.

Metric: A measurement scale and the method used for measurement.

NOTE 9 Metrics can be internal or external.

Metrics includes methods for categorizing qualitative data.

ICS 35.080
Price based on 86 pages