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

ISO/IEC JTC1/SC7 N2073

1999/03/10

Document Type	2nd PDTR Ballot
Title	PDTR Ballot of PDTR 9126-2: Software Engineering - Product Quality Part 2 - External Metrics.
Source	JTC1/SC7 Secretariat
Project	07.13.01.02
Status	2nd PDTR Ballot
References	N2071
Action ID	ACT
Due Date	1999/06/10
Mailing Date	1999/03/10
Distribution	SC7_AG
Medium	Encoded Acrobat
No. of Pages	114
Note	Please send early comments directly to Prof. Azuma so they can be discussed at the upcoming Plenary meeting.

Address reply to: ISO/IEC JTC1/SC7 Secretariat
Bell Canada - Quality Engineering & Research
1050 Beaver Hall Hill, 2nd Floor, Montréal (Québec) Canada H2Z 1S4
Tel.: +1 (514) 391-8286 Fax: +1 (514) 870-2246
sc7@qc.bell.ca

ISO/IEC JTC1/SC7 PDTR 9126-2.2	
Date 1999/03/09	Reference number ISO/JTC 1/SC 7 N2073
Supersedes document	

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

ISO/JTC 1/SC 7 Committee Title **Software Engineering** Secretariat: Standards Council of Canada (SCC)	Circulated to P- and O-members, and to technical committees and organizations in liaison for: X voting by (P-members only) **1999/06/09** Please return all votes and comments in electronic form directly to the SC 7 Secretariat by the due date indicated.

ISO/IEC JTC1/SC7

Title: PDTR Ballot of PDTR 9126-2: Software Engineering - Product Quality Part 2 - External Metrics.

Project: 07.13.01.02

Introductory note: See page ii of the document

Medium: Encoded Acrobat

No. of pages: 114

Vote on PDTR 9126-2.2	
Date of circulation **1999/03/09** Closing date 1999/06/09	Reference number ISO/JTC 1/SC 7 N2073

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

ISO/IEC JTC1/SC7

Title: PDTR Ballot of PDTR 9126-2: Software Engineering - Product Quality Part 2 - External Metrics.

Project: 07.13.01.02

Vote:

__ APPROVAL OF THE DRAFT AS PRESENTED

__ APPROVAL OF THE DRAFT WITH COMMENTS AS GIVEN ON THE ATTACHED

__ general:

__ technical:

__ editorial:

__ DISAPPROVAL OF THE DRAFT FOR REASONS ON THE ATTACHED

__ Acceptance of these reasons and appropriate changes in the text will change our vote to approval

__ ABSTENTION (FOR REASONS BELOW):

P-member voting:
National Body (Acronym)

Date:
YYYY-MM-DD

Submitted by:
Your Name

ISO/IEC JTC1/SC7/WG6

EVALUATION AND METRICS

TITLE	**ISO/IEC 9126-2 : Software ENgineering – Product quality Part 2 : External Metrics**
DATE:	**Jan. 18, 1999**
SOURCE:	**ISO/IEC JTC1/SC7/WG6**
WORK ITEM:	**Project 7-13-01-02**
STATUS:	**Version 5.7 – reflects ISO/IEC JTC1/SC7/WG6 Sydney, Australia meeting in Nov. , 1998) and revised for 2nd PDTR Ballot**
REFERENCE:	**6/N 363 (7N1563), 7N1669, 6/N 403R(7N 1767) ,7N1809,WG6-ROMA12R, 6/N 425**
DOCUMENT TYPE:	**Proposed Draft Technical Report**
ACTION:	**For 2nd PDTR Ballot.**
EDITOR:	**Atsushi Yamada** **S&S Laboratory, Research & Development Center, Toshiba Corporation** **70 Yanagi-cho, Saiwai-ku, Kawasaki-city** **Kanagawa 210-8501, Japan** **TEL: +81-44-548-5480 , FAX: +81-44-520-5855** **E-mail: yamada@ssel.toshiba.co.jp**
CO-EDITOR:	**Dr. Chris Vermaak** **South Africa** **TEL: - , FAX: -** **E-mail: absa.abcv001@memo.absa.co.za**
REVIEWERS	**Ms. Antonia Jeanrenaud (Italy), Mr. Dave Hufton (Singapore),** **Dr. Jili Vanicek (Czech Republic),** **Mr. Vipula Godamunne (Australia), Dr. Alain Abran (Canada),** **Ms. Carole Nadeau (Canada: Previous Co-Editor),** **Mr. Alain April (Canada: Previous Co-Editor)**

REVISION HISTORY

Document #	Version	Date	Description
6/N 443	5.7	Jan. 99	Proposed draft techinical report for 2nd PDTR ballot reflecting Sydney meeting discussion
6/N	5.6	Nov.98	Version 5.6 – revised working draft after PDTR registration (for Sydney, Australia meeting Nov. 23, 1998)
6/N 425	5.5	May. 98	Working version for South Africa meeting discussion, which is under revising to resolve both of ballot comments before Roma meeting and additional comments after Roma.
Roma 12R	5.4	Nov.97	Working version reflecting discussion and work to coordinate 9126 part2 and 3 as a meeting work product.
Roma 12	-	Nov. 97	Working version reflecting discussion and work to coordinate 9126 part2 and 3 as a intermediate meeting work product.
7N 1809	-	Nov. 97	PDTR letter ballot summary - approved with comments
7N 1767 6/N 403R	5.3.2	Jun. 97	Working version after the review in California June 1997 and circulated in SC7 for PDTR Ballot
403R	5.3.2, 5.3.1 5.3	Jun. 97	Working version after the review in California June 1997 (3rd Walnut Creek meeting version proposing more widely resolutions to the ballot comments)
Walnut Creek 14R	5.2	Jun. 97	Working version for review in California June 1997 (2nd Walnut Creek meeting version proposing more widely resolutions to the ballot comments)
Walnut Creek 14	5.1	Jun. 97	Working version for review in California June 1997 (1st Walnut Creek meeting version proposing more widely resolutions to the ballot comments)
7N 1669	-	Feb. 97	WD & PDTR Registration letter ballot summary - approved with comments
	5.0	Jan. 97	Working version for review in California June 1997
Curitiba	4.9	Nov. 96	Working document for review at Curitiba meeting
6/N 363	4.8	July 96	Working document for PDTR after Prague meeting May 1996 and circulated in SC7 for WD & PDTR Registration Ballot
6/N 320		May. 96	Working document for review at Prague meeting May 1996
		May. 95	Working document for review at Brisbane meeting May 1995
		May. 94	Working document for review at Ottawa meeting May 1994

ISO/IEC JTC 1/SC 7

Date: Jan. 18 1999

PDTR

ISO/IEC JTC 1/SC 7/WG 6

Secretariat: Canada and Japan

Software Product Quality – part2 : External Metrics

Document type: Technical Report type 2
Document subtype: Not applicable
Document stage: (30) Committee
Document language: E

Contents

Foreword

Boilerplate text for ISO standards:

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Boilerplate text for JTC 1 standards:

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.

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Introduction

This international technical report (ITR) provides external metrics and quality in use for measuring attributes of six quality characteristics defined in ISO/IEC 9126-1. This report also provides data element, which are commonly used for composing metrics. The metrics and data elements listed in this ITR are considered to be a basic set. Developers, evaluators, quality manager and acquirers may select some metrics and data elements from this technical report for defining requirement, evaluating software products, measuring quality aspect and other purposes. But they may also use metrics and data element, which are not included here. This report is applicable to any kind of software product, although each of metrics is not always applicable to every kind of software.

ISO/IEC 9126-1 defines terms for global characteristics and how these characteristics are decomposed in sub-characteristics. ISO/IEC 9126-1 however does not describe how to state requirements with respect to sub-characteristics, or how, for a given product, any of these sub-characteristics could be measured. These two objectives cannot be achieved by further decomposing the sub-characteristics recursively in the same way. Something new is required. To attempt to partially fill this gap between the sub-characteristic concept (ISO/IEC 9126-1) and a measurable characteristic, technical reports are presented: 9126-2 on external metrics, 9126-3 on internal metrics and 9126-4 quality in use metrics.

Software Product Quality – part2 : External Metrics

1 Scope

This international technical report (ITR) present external and quality in use metrics for quantitatively measuring software quality in terms of characteristics and sub-characteristics defined in ISO/IEC 9126-1 and explains methods for determining characteristics values from attributes values.

This ITR 9126-2 External Metrics present:

- a general intent of quality metrics (Clause 4)
- which measurable characteristics contribute to which software quality characteristics (including sub-characteristics) so that they can serve as metrics for these characteristics
- identify the metrics desirable properties
- an quality approach experimentation example

This report contains the terminology related to the metrics measurements, the usage of metrics in the life cycle process, and the introductory basic sets of external and quality in use metrics for each software quality characteristic and sub-characteristic. This report provides a guide to the user of metrics for planning evaluation, selecting metrics, designing metrics, applying metrics, and interpreting measurement data.

NOTE : Out of scope

The ITR 9126-2 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 value, which does not depend on specific user needs but depends on generic factors, for example, human cognitive factors.

Users of this ITR include, as an example:

1. acquirer (an organization that acquires or procures a system, software product or software service from a supplier)
2. evaluator (an 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)
3. developer (an organization that performs development activities, including requirements analysis, design, testing through acceptance during the software life cycle process)
4. maintainer (an organization that performs maintenance activities)
5. supplier (an organization that enter 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;
6. user (an individual that uses the software product to perform a specific function) when evaluating quality of software product at acceptance test;
7. quality managers (an organization that perform an systematic examination of the software product or software services) when evaluating software quality at qualification test

2 Conformance

3 Reference(s)

1. ISO 8402: 1994, Quality management and quality assurance - Vocabulary
2. ISO/IEC 9126-1: (new), Information technology - Software product quality – Part 1 : Quality model.
3. ISO/IEC 9126-3: (new), Information technology - Software product quality – Part 3 : Internal metrics.
4. ISO/IEC 9126-4: (new), Information technology - Software product quality – Part 4 : Quality in use metrics.
5. ISO/IEC 14598-1: (new), Information technology - Software product evaluation – Part 1 : General Overview
6. ISO/IEC 14598-2: (new), Information technology - Software product evaluation – Part 2 : Planning and management
7. ISO/IEC 14598-3: (new), Information technology - Software product evaluation – Part 3 : Process for developers
8. ISO/IEC 14598-4: (new), Information technology - Software product evaluation – Part 4 : Process for acquirers
9. ISO/IEC 14598-5: (new), Information technology - Software product evaluation – Part 5 : Process for evaluators
10. ISO/IEC 14598-6: (new), Information technology - Software product evaluation – Part 6 : Documentation of evaluation modules
11. ISO/IEC 12207: 1995, Information technology - Software life-cycle processes.
12. ISO 2382-20: 1990, Information technology - Vocabulary.

4 Term(s) and definition(s)

For the purposes of this ISO/IEC 9126-2, the definitions contained in ISO/IEC 14598-1 and ISO/IEC 9126-1apply.

5 Symbols (and abbreviated terms)

6 General intent of the quality metrics

These reports (9126-2 and 9126-3) provides a suggested set of quality metrics (external, quality in use and internal metrics) to be used with the 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 ITRs, the user should specify how the metrics relate to the 9126-1 quality model or any others substitute quality model that is being used.

The user of these ITRs may select the quality characteristics and sub characteristics to be evaluated, from ISO/IEC 9126-1; identify the appropriate direct and indirect measures to be applied, identify the relevant metrics and then interpret the measurement result in a objective manner.

The metrics desirable properties (clause 6.3) should be applied to the new or modified metric

6.1 Concept descriptions

This sub clause will reinforce the ISO/IEC 9126-1 concepts of internal, external and quality in use metrics before explaining how to use these quality metrics.

The internal metrics may be applied to a non-executable software product during it s 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 detect quality issues and take corrective actions during the early stages of the development life cycle process.

The external metrics may be used to measure the quality of the software product by measuring the behavior of the system of which it is a part. The external metrics should only be used during the testing stages of the life cycle process and during any operational stages. This is achieved by executing the software product in the system environment that it is intended for. (It can be performed in in-house testing site or independent testing laboratory over the system environment site that it is intended for.)

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

What bind these three types of metrics (internal, external and quality in use) in an objective manner during an evaluation process are the quality criteria to be used for determining the quality of the software product. The metrics should be applied in a manner that supplements each other during the evaluation.

When the software quality requirements are defined, the software quality characteristics or sub-characteristics, which represent the quality requirements, are listed. Then, the appropriate external metrics and acceptable ranges are specified to quantify the quality criteria, which validate that the software meets the user needs. The internal quality attributes of the software are then defined and specified to plan to achieve the required external quality characteristics finally and to build them into the intermediate product during development. Appropriate internal metrics and acceptable range are specified to quantify the internal quality characteristics so that they can be used for verifying that the intermediate software meets the internal quality specifications during the development. Quality in use metrics measure the extent to which a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction. The relationship with others characteristics depends on the type of the user.

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

In relation to these definitions the next sub clause will present the interpretation of these new concepts.

6.1.1 External metrics

External metrics should be designed to:

1. represent software product quality which includes software quality characteristics and sub-characteristics defined in ISO/IEC 9126-1, during testing or operation;

2. validate that the software satisfies external quality requirements;

3. predict the actual quality in use;

4. describe the extent to which the software product keeps on satisfying user's stated or implied needs during the actual operation by user.

External metrics should be designed to employ the following measurements:

1. Measurements of software behavior in testing and operating, and in cooperation with other software, hardware, or system;

2. Measurements of user behavior.

NOTES:

1. Measurements of software behavior include measurements of incidents which are threats to human life and health, environmental natural resources, data destruction, inconsistency or misleading of information, broken security, degrade of service, advantages or profits in a market, gain or loss of economy etc.

2. The external metrics may be used to predict and to indicate the density of potential faults remaining in the software.

3. Internal measurements may be used to calculate external measures. For example, the number of program steps, which is an internal measure, may be used to normalize the external measure.

6.1.2 Internal metrics

Internal metrics provide users, evaluators, testers, developers, quality managers, or managers with benefits whereby they are able to evaluate software quality of intermediate and final products during the time period before the software product is executable.

Internal metrics should be designed to:

1. represent software quality of intermediate and final products for characteristics which includes software quality characteristics and sub-characteristics defined in ISO/IEC 9126-1, during software product is not executable;

2. guide to planning and implementation of designs, programs or processes improvement, which affect intermediate and final software products;

3. verify that the intermediate and final software products satisfy internal quality requirements which are quality improvement plans for designs, programs or processes;

4. predict the external metrics or quality.

Internal metrics should be designed to employ following measurement:

Measurements of static software attributes which have appeared in the text of source code, in a graph or table representation of control, data flow, or state transition structure, or in documentation of the product itself. Examples of metrics for quality in use are given in ISO/IEC 9126-4.

6.1.3 Quality in use metrics

The goal is to achieve the necessary and sufficient quality to meet the real needs of users. For systems with end users, this means that specified types of users should be able to carry out specific types of tasks to a required level of productivity and user satisfaction in specified environments. Evaluating quality in use validates software quality in specific user-task scenarios.

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

Quality in use may be influenced by any of the quality characteristics, and is thus broader than usability, which is only concerned with the ease of use and attractiveness. It can be measured by the extent to which the specified users can achieve their goals with effectiveness, productivity (task efficiency), safety and satisfaction. Effectiveness can be measured by the accuracy and completeness with which users achieve specified goals, productivity (task efficiency) by the resources expended in relation to task effectiveness, safety by the level of impact with possibility againt to specific risk arising scenarios, and satisfaction by attributes to the use of the product. Examples of metrics for quality in use are given in ISO/IEC 9126-4.

6.2 Interpretation of measurement

The purpose of this sub clause is to procure to the reader the interpretation of the measurement concept according to the ISO/IEC 9126-1 and ISO/IEC 14598-1.

6.2.1 Direct measure for software

A direct measure is a measure of an attribute that does not depend upon a measure of any other attribute.

Some measurements of software attributes can be done without being influenced by factors such as the choice of the server where the software is executed, behavior of a user, or other external factor. Those measurements are referenced here as direct measures. Some examples are size of source and executable code, total number of menu options of an application or number of configuration items.

6.2.2 Indirect measure for software

An indirect measure is derived from (direct or indirect) measures of one or more attribute. For example, measure of response time is not only affected by the evaluated software itself, but also by the operating environment including but not restricted to computer hardware and operating system software. So, the response time of software is derived from the response time of the computing environment as well.

6.2.3 Indicators

Some measures can be estimated or predicted from other measures. Those measures are referenced here as indicators, and may be useful to estimate or predict attributes that cannot be measured directly, or can not be measured at all without a model. For example, the response time is not measurable while the software is still a non-executable intermediate product. Therefore, program path length may be measured and used as an indicator to predict the future response time before the software becomes executable.

As another example, in the case that the software is executed in in-house testing, the response time under a testing environment is measurable but may vary from the response time experienced by an actual user in the final operating environment. Therefore, the response time of the software in the final user environment can be predicted from the response time of the testing environment.

Finally, in the case that measuring efficiency of the software is dominantly dependent on time measurement, the response time may be used as an indicator to represent efficiency quantitatively in the software quality evaluation.

6.3 Metrics desirable properties

The accuracy and correctness of a quality evaluation relies strongly on the metrics used on that process. In order to improve metrics confidence, a list of validation requirements, which should be present on every metric applied in an evaluation, is given below.

Whenever a metric does not satisfy these validation requirements, the metric description should explain the associated constraint and, as far as possible, how that situation can be handled.

NOTE: Those properties are suited to the requirements on measurements of ISO/IEC 14598-1, and to the requirements on the evaluation of ISO/IEC 14598-5.

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

a) repeatability: repeated use of the metric in the same product to the same evaluation specification (including the same environment) by the same evaluators, test users and environment should produce results that can be accepted as being identical,

b) reproducibility: use of the metric in the same product to the same evaluation specification (including the same environment) by different evaluators, test users and environment should produce results that can be accepted as being identical.

Indicativeness: Capability of a metrics, which shows the part to be improved, when measure (value) is not sufficient to clear the criteria for the evaluation.

Availability: the selected or proposed metric should provide documented evidence of the availability of the metric for use.

Cost Effectiveness: user of the metric should provide documented evidence of the cost of applying the metric.

Correctness: This requirement regroups the properties associated with metrics objectivity, impartiality and precision

a) objectivity: the metric results and its data input should be factual, i.e. not colored by the feelings or the opinions of the evaluator, test users, etc.

NOTE: Since it is impossible to avoid subjective factors, especially on usability evaluation, it is recommended that procedure for assigning the number or category is enough well reviewed to be agreeable. This does not mean that it is not applicable such a metric with user sensitive testing based on questionnaire or interview.

Impartiality: the measurement should not be biased towards any particular result.

a) Precision: 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.

Meaningfulness: the measurement should produce meaningful results about the software behavior or quality characteristics. User of metrics will describe the goal or question the metric result aims at fulfilling.

7 Basic use of external metrics for quality characteristics (Metrics tables)

The following set of metrics are recommended to be used as basic metrics which give measures or may be applied as checklists to represent software quality characteristics. Although, there are some practical experiences in progress, these metrics are draft and need more validation and feedback. Therefore they are listed in order of software quality characteristics and sub-characteristics.

Additional specific metrics for particular purposes are not included, but are provided in other related documents, such as functional size measurement or precise time efficiency measurement.

Applicable metrics are not limited to these listed here. If there is a trial to apply a new metric, the appropriate model and the practical experiences should be evaluated and specified.

The title terms used in these metrics tables are followings.

a) Metrics name : Metrics name characterizes measureable attribute of software and represents a unique or group of measurements. Metrics name has the same or similar name in internal, external and quality in use metrics, when they are intended to be mutually corresponded respectively.

b) Purpose of the metrics : This helps to identify what user of metric can know by using the metric. This is described as a questionary style to look up easily in accordance with Goal / Question / Metric flamework.

c) Method of application : This helps to understand what way are useful and recommended to apply metrics.

d) Measurement, formula and data element computations : This helps to understand what kind of measurement, formula and data element are used to compute measure.

e) Interpretation of measured value : This helps to understand the range of measured value and the interpreted better range.

f) Scale types : The measurement scale types should be identified for each measure, when a user of metrics has the result of a measurement and use measure to calculate together or compared with other. The average, ratio or difference values may have no meaning for some measures. Such scale types are: Nominal scale, Ordinal scale, Intervals scale, Ratio scale, Absolute scale. M'=F(M), where F is the admissible function, explain what the admissible function is (if M is a metric then M'=F(M) is also a metric).

g) Measurements types : For designing procedure for collecting data, interpreting fair meanings, and normalizing measures for comparison, a user of metrics should identify and take account of measure type of measurement employed by a metric.

Size measure type : A measure of this type represents a particular size of software according what it claims to measure within its definition. Normalizing other measures with a size measure can give comparable values in terms of units of size. The size measures in this report can be used for software quality metrics. Functional size, as well as source lines of code, is an example of one type of size that software may have.

Time measure type : The user of metrics of time measure type should record time periods, how many examined sites and how many users took part of measurements. The user of metrics should be aware that there are many ways in which time can be measured as a unit.

Count measure type : This measure type identifies number of counted number, turn, event or incident with investigation activity. Investigation activity includes reviewing, testing and operating, and using from view of human-engineering and ergonomics.

h) Input to measurement : This helps to understand what kinds of information or documents are generally required to do measurement.

i) ISO/IEC 12207 SLCP Reference : This suggest processes of software life-cycle processes (SLCP) defined in ISO/IEC 12207, in which the metric is beneficially applicable.

j) Perspective/Beneficer : This helps to understand whose view and benefit are strogly related with the metric, though any other personel and party related to the software are also receive benefit.

NOTE:

1. All the title terms used in these tables are described farther more in annex A.

2. All the definitions related to this clause are presented in ISO/IEC 9126-1.

3. All metrics described in this clause are related to the following:

The behavior of the system may be observed with following aspects:

a) differences between actual executed results and quality requirements specifications (a view of quality validation testing);

b) occurrences of unexpected time behavior or resource utilization during operation, which may be due to not stated implied needs.

7.1 Functionality metrics

An external functionality metric should be able to measure an attribute such as behavior of system containing the software.

The behavior of the system may be observed with following aspects:

a) differences between actual executed results and quality requirements specification (a view of quality in use validation testing);
b) insufficient performed function during actual user operation that may be implied not stated (a view of quality in use).

7.1.1 Suitability metrics

An external suitability metric should be able to measure an attribute such as the occurrence of the unsatisfied functions or occurrence of unsatisfied operations from behavior of system during testing and user operation.
Unsatisfied function implies:

a) a function performing task which does not conform to specified ones in requirements specifications or user manuals, or functions specified in the user manual that are simply not performed;

b) a function, which does not meet appropriately the tasks for user's reasonable, intended specific use.

7.1.2 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 calculation results or less precision results beyond the range of erroneous of calculation, for example, it is caused why it is too shortage of significance to calculate;
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.

7.1.3 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 is transferred easily between the software product and other systems, other software products, or equipment which are connected.

7.1.4 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:

- failing to prevent leak of secure output information or data;
- failing to prevent lost of important data;
- failing to defend against illegal access or illegal operation.

7.1.5 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 is of the software product to failing to adhere to standards, conventions or regulations in laws and similar prescriptions which are required to be adhered.

Table 7.1.1 Suitability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measure ment	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Functional implementation completeness	How much 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.	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.	X=Abs.	A=Count B= Count X=Count/ Count	Require-ment specifica-tion Evaluation report	6.5 Validation, 6.3 Quality Assurance, 5.3 Qualificati on testing	Developer, SQA
	NOTE: 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.			NOTE : 2. This metric is suggested as experimental use.					
Functional implementation coverage	What is the amount of functions that are done according to requirement specifications ?	Do functional tests (black box test) of the system according to the requirement specifications. Count the number of functions that are complete versus the ones that are not.	X = A / B A = Number of correctly implemented functions confirmed in evaluation B = Number of functions described in requirement specifications	0<=X<= 1 The closer to 1.0 is the better.	X=Abs.	A=Count B= Count X=Count/ Count	Require-ment specifica-tion Evaluation report	6.5 Validation, 6.3 Quality Assurance, 5.3 Qualificati on testing	Developer, SQA
	NOTE: 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.			NOTE : 2. This measure represents a binary gate checking of determining whether presence or absence of a feature.					
Functional specification stability (volatility)	How much stable are the functional specifications of the system right after entering operation ?	Count the number of existing functional specifications that had to be changed after the system is put into operation	X = 1- A / B A= Number of functions changed after entering operation starting from entering operation B= Number of functions described in requirement specification (or any function size measure) NOTE : This metric is suggested as experimental use.	0<=X<= 1 The closer to 1.0 is the better.	X=Abs.	A= Count B= Size X= Count/ Size	Require-ment specifica-tion Evaluation report	6.8 Problem Resolution 5.4 Operation	Maintainer

Table 7.1.2 Accuracy metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Accuracy to expectation	Are differences allowable between the actual and reasonable expected results ?	Do input .vs. output test cases and compare the output to reasonable expected results. Count the number of test cases beyond allowable difference against reasonable expected results.	X=A / T A= Number of users encountered cases beyond allowable difference against reasonable expected results T= Operation time NOTE : Reasonable expected results may be identified in a requirement specification, a user operation manual, or hearing to users	0<=X The closer to 0 is the better.	X= Abs.	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
Computational Accuracy	How often does 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.	X= Abs.	A= Count T= Time X= Count/ Time	Req. spec. Test report	6.5 Validation 6.3 Quality Assurance	Developer User
Precision	How often does 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.	X= Abs.	A= Count T= Time X= Count/ Time	Req. spec. Test report	6.5 Validation 6.3 Quality Assurance	Developer User

Table 7.1.3 Interoperability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Data exchangeability (Data format based)	How complete are the downstream interface functions for specified data transfer ?	Test each downstream interface function output record format of the system according to the data fields specifications.	Data exchangeable format X= A / B A= Number of data formats which are approved to be exchanged with other software or system during testing on data exchanges, B= Total number of data formats to be exchanged NOTE : It is recommended to test specified data transaction.	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count		6.5 Validation	Developer
Data exchangeability (User's success attempt based)	How often successful are the data transfers between target software and the other software ? Can user usually succeed to exchange data ?	Count the number of times that an interface functions was used and failed.	User successful data exchange ratio Y= 1 – (A / B) A= Number of cases which user fail to exchange data with other software or systems B= Number of cases which user attempt to exchange data	0<=Y<= 1 The closer to 1.0 is the better.	Y= Abs.	A=Count B= Count Y=Count/ Count		5.4 Operation	Maintainer
Intersystem interface standard consistency	Is the standard for interface design identified in the specifications followed consistently ?	Test each interface of the system for consistency with the standard identified in the specifications.	Interface standard consistent ratio X= A / B A= Number of check items of intersystem interface which are approved at testing that they are consistent with standard/rule of intersystem, B= Total number of check items of intersystem interface	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count		6.5 Validation	Developer

Table 7.1.4 Security metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Access auditability	How complete is the audit trail concerning the user access to the system and data ?	Evaluate the amount of access record that the system records in the access history database.	Access history record holding ratio 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 NOTE :1. Accesses to data may be measured only with testing activities. 2. This metric is suggested as experimental use.	0<=X<=1 The closer to 1 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Test spec. Test report	6.5 Validation	Developer
Access controllability	How complete is the detection of user access to the system ?	Try to get access to the system by unauthorized ways.	Illegal operation access detectable ratio X= A / B A= Number of detected illegal operations B= Number of illegal operations anticipated in specification NOTE : If it is necessary to complement unexpected illegal operations to be detected, additional intensive abnormal operation testing should be conducted.	0<=X<=1 The closer to 1 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Test spec. Test report 5.4 Operation report	6.5 Validation 6.3 Quality Assurance	Developer

Table 7.1.4 Security metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Data corruption Prevention	How often fatal data corruption occur ?	Count the occurrences of data corruption, and of fatal data coruptions events.	Frequency of fatal data corruption events X= 1 – A / N A= Number of times that a major data corruption event occurred N= Number of test cases tried to occur data corruption event Frequency of data corruption events Y= 1- B / N B= Number of times that a minor data corruption event occurred NOTE : 1. Intensive abnormal operation testing is needed to obtain minor and major data corruption events. 2. It is recommended to grade impact of data corruption event such are following examples. 3. Major data corruption event : - reproduce and recover impossible ; - second affection distribution to wide ; - importance of data itself. B) Minor data corruption event : - reproduce or recover possible and – no second affection distribution ; - importance of data itself. 3. This metric is suggested as experimental use.	0<=X<= 1 The closer to 1.0 is the better.	X,Y= Abs. A,B= Ord.	A=Count B=Count X=Count/ Count Y= Count/ Count	Test spec. Test report Operation report	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	Maintainer

Table 7.1.5 Compliance metrics (compliance for functionality)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfied compliance items coverage	How completely does the software adhere the standards, conventions or regulations in laws ?	Previously specify required compliance items based on standards, conventions or regulations in laws which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	Ratio of satisfied compliance items X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE : 1. 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. 2. It is suggested to count number of fail, because problem detection is an objective of effective testing and also suitable for counting and recording.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specification of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

7.2 Reliability metrics

An external reliability metric should be able to measure attributes related to the behaviors 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 case.

7.2.1 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.

7.2.2 Fault tolerance metrics

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

7.2.3 Recoverability metrics

An external recoverability metric should be able to measure such attribute as the software with system being able to re-establish its adequate level of performance with minimal lost of data.

7.2.4 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, which is of the software product to failing to adhere to standards, conventions or regulations relating to reliability which are required to be adhered.

Table 7.2.1 Maturity metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Estimated latent failure density	How many problems still potentially exist and may emerge ? How often may users come across failures or faults in the future ?	Count the number of detected failures and predict potential number by using a reliability growth estimation model.	Estimated residuary latent failure density X= {ABS(NPFI - NAFI)} / SIZE ABS()= Absolute Value NPFI= total number of predicted latent failures NAFI= total number of actually detected failures SIZE= product size	0<=X It depends on stage of testing. Finally, the smaller is the better.	X= Abs.	NPFI= Count NAFI= Count SIZE= Size X= Count/ Size	Test report Operation report Problem report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation 6.5 Validation 6.3 Quality Assurance	Developer Tester
	NOTE : 1.When total number of actually detected failures becomes larger than total number of predicted latent failures, it is recommended to predict again and estimate more larger number. 2. It is recommended to use several reliability growth estimation models and chose the most likelihood fit one and repeat prediction with monitoring detected failures.			NOTE : 3. It may be helpful to predict upper and lower number of latent failures. 4. It is necessary to convert this value (X) to the <0,1> interval if making summarization of characteristics					
Estimated latent fault density	How many problems still potentially exist and may arise failures ?	Count the number of detected faults and predict potential number by using several reliability growth estimation models.	Estimated residuary latent fault density X= {ABS(NPFU - NAFU)} / SIZE ABS()= Absolute Value NPFU= total number of predicted latent faults in a software product NAFU= total number of actually detected faults SIZE= product size	0<=X It depends on stage of testing. Finally, the smaller is the better.	X= Abs.	NPFT= Count NAFT= Count SIZE= Size X= Count/ Count	Test report Operation report Problem report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation 6.5 Validation 6.3 Quality Assurance	Developer Tester SQA
	NOTE : 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. 2. It is recommended to use several reliability growth estimation models and chose the most likelihood fit one and repeat prediction with monitoring detected faults.			NOTE : 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 summarization of characteristics					

Table 7.2.1 Maturity metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Failure density **(Fault density)**	How many problems were detected ?	Count the number of detected failures (or faults) and calculate density.	3. Failure density X= NFAI / SIZE b) Fault density Y= NFAU / SIZE NFAI= number of detected failures NFAU= number of detected faults SIZE= product size	0<=X,Y It depends on stage of testing. Finally, the smaller is the better.	X,Y= Abs.	NFAI= Count NFAU= Count SIZE= Size X,Y= Count/ Size	Test report Operation report Problem report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation 6.3 Quality Assurance	Developer Tester SQA
	NOTE : 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. 2. The number of detected failures (or faults) divided by the number of test cases indicates effectiveness of test cases			NOTE : 3. It is necessary to convert this value (X,Y) to the <0,1> interval if making summarization of characteristics					

Table 7.2.1 Maturity metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Failure Resolution	Evaluate the amount of failure conditions that are resolved.	Failure resolution is ensured by observation that the same type of failure never occurs again with the execution condition being acceptably identical to the condition with which failure once occurred. Maintain a problem resolution report describing status of all the failures.	a) Ratio to actually detected number X= NRFI / NAFI NRFI= number of resolved failures NAFI= total number of actually detected failures b) Ratio to estimated number Y= NRFI / NPFI NPFI= total number of predicted latent failures	0<=X<= 1 The closer to 1.0 is the better. More failures resolved 0<=Y The closer to 1.0 is the better.	X= Abs. Y= Abs.	NRFI= Count NAFI= Count NPFI = Count X= Count/ Count Y= Count/ Count	Test report Operation (test) report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation	Maintainer
	NOTE: 1. It is recommended to monitor trend of this measure along with time. 2. Total number of predicted latent failures may be estimated with one of reliability growth models adjusted by parameter data based on the past history. 3. It is recommended to monitor this estimated failure resolution ratio Y and when Y > 1, to investigate whether it is going well to detect and to resolve early more enough defects before delivery than estimation, or it is going worse because of more defects contained in the software than usual. Otherwise, when Y < 1, then it is recommended to investigate whether it is going well because of more defects contained in the software than usual, or it is going worse because of defects detected less than usual. 4. Multiple problem reports may be received for the same failure, when the software is under user beta testing or multiple sites testing.			NOTE: 5. It is necessary to convert this value (Y) to the <0,1> interval if making summarization of characteristics					

Table 7.2.1 Maturity metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Fault Removed	How many faults have been corrected ?	Count the number of detected faults and permanently corrected faults. Fault correction is ensured by observation of resolved failure or by testing or reviewing modification. Maintain a problem resolution report describing status of all the failures.	a) Ratio to actually detected number X= NCFU / NAFU NCFU= number of corrected faults NAFU= total number of actually detected faults b) Ratio to estimated number Y= NCFU / NPFU NCFU= number of corrected faults NPFU= total number of predicted latent faults in a software product	0<=X<= 1 The closer to 1.0 is the better, less faults remain. 0<=Y The closer to 1.0 is the better, less faults shall remain	X= Abs. Y= Abs.	NCFU= Count NAFU= Count NPFU= Count X= Count/ Count Y= Count/ Count	Test report Organization database	5.3 Integration 5.3 Qualifica-tion testing 6.5 Validation 6.3 Quality Assurance	Developer

NOTE:

1. It is recommended to monitor trend of this measure along with time.

2. Total number of predicted latent faults may be estimated with one of reliability growth models adjusted by parameter data based on the past history.

3. It is recommended to monitor this estimated faults resolution ratio Y and when Y > 1,
to investigate whether it is going well to detect and to resolve early more enough defects before delivery than estimation, or it is going worse because of more defects contained in the software than usual.

Otherwise, when Y < 1,
then it is recommended to investigate whether it is going well because of more defects contained in the software than usual, or it is going worse because of defects detected less than usual.

NOTE:

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

Table 7.2.1 Maturity metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Mean time between failures (MTBF)	How frequent are the failures of the system in operation?	Trace failures occurred during observed testing or operation time.	Mean time between failures a) X = TOP / NAFI b) Y = TSI / NAFI TOPT = operation time TSIB = sum of time interval between failure occurrence and the next one NFAI = total number of actually detected failure (Failures occurred during observed operation time.)	0<X,Y The longer is the better. The longer time can be expected between failures.	X,Y= Ratio.	NAFI= Count TOPT= Time TSIB= Time X=Time / Count Y=Time/ Count	Test report Operation (test) report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation testing 5.4 Operation	Maintainer User
	NOTE: 1. The following investigation may be helpful: - distribution of time interval between failure occurrence to the next; - 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.			NOTE: 2. Failure rate or hazard rate calculation may be alternatively used. 3. It is necessary to convert this value to the <0,1> interval if making summarization of characteristics					

Table 7.2.1 Maturity metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Test coverage	Have enough user operation scenario test cases been executed?	Count the number of test cases from user operation view which are actually tested.	Specified operation scenario testing coverage X= A / B A= Number of actually performed test cases representing operation scenario during testing B= Number of test cases to be performed NOTE: 1. Test cases may be normalized 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.	0<=X<=1 The closer is to 1, the better test coverage.	X= Abs.	A= Count B= Count X= Count/ Count	Test spec. Test report Operation report	5.3 Qualifica-tion testing 6.5 Validation 6.3 Quality Assurance	Developer Tester SQA
Test overcome	Is product going to pass successfully a lot of test cases?	Count the number of passed test cases which have been actually executed.	Passed test case ratio X= A / B A= Number of passed test cases during testing or operation B= Number of test cases to be performed. NOTE: 1. It is recommended to perform heavily stress testing, for example, to use actual log data at highly peak period of operation for testing. 2. Passed test cases may be normalized by software size, that is: passed test case density 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.	0<=X<=1 The closer is to 1 the better overcome of tests.	X= Abs.	A= Count B= Size X= Count/ Size	Test spec. Test report Operation report	5.3 Qualifica-tion testing 6.3 Quality Assurance	Developer Tester SQA

Table 7.2.2 Fault tolerance metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Breakdown avoidance	How often can user avoid breakdown of system, even if critical failures occurred?	Count the number of breakdowns occurrence with respect to number of failures. If it is under operation, analyse log of user operation history.	Breakdown avoidance ratio X= 1-(A / B) A= Number of breakdowns B= Number of failures NOTE: The breakdown means executing of any user task is suspended until system is restarted up, or its control is lost until system is enforced to be shut down.	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation	User Maintainer

Fault tolerance metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Failure avoidance	How often can user avoid critical or serious failure?	Count the number of critical and serious failure occurrence with respect to number of causes.	X=A / B A= Number of avoided critical and serious failure emergence 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 is the better, the user can more often avoid critical or serious failure.	X= Abs.	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
	NOTE: 1. It is recommended to categorize 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 out of service and no alternative way of operating (workaround); -Average: most functions are still available, but limited performance occur with limited or alternate operation (workaround); -Small: a few functions experience limited performance with limited operation; -None: impact does not reach end user			NOTE: 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. test case can include the human incorrect operation					
Incorrect operation avoidance	How often can user avoid failure, even if user operates incorrectly?	Do operational test or analyse log of user operation history. Count the number of critical and serious failure occurrence with respect to number of user's incorrect operation.	X=A / B A= number of avoided user's erroneous operations or inputs during operation B= number of user's erroneous operation or input during operation NOTE: This metric may be used experimentally.	0<=X<= 1 The closer to 1.0 is better, more incorrect user operation avoided	X= Abs.	A=Count B=Count X=Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User Maintainer

Table 7.2.3 Recoverability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Availability	Can user use software system enough every time?	Investigate recovery testing.	Availability X= { To / (To + Tr) } User operation available ratio Y= C / D (This is from software system operation view.) To = operation time Tr = time to repair C= total available turns of user's successful software use when user attempt to use D= total number of turns of user's attempt to use the software during observation time. This is from user callable function operation view. NOTE: It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.	0<=X<=1 The larger is the better, the user can use the software for more time. 0<=Y<=1 The larger is the better.	X,Y= Abs.	To = Time Tr = Time X= Time/ Time C= Count D= Count Y= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation	User Maintainer
Mean down time	How long is usually system down?	Investigate recovery testing.	Mean time: X= T / B T= Total down time B= 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.	X= Ratio	T=Time B=Count X=Time/Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer
	NOTE: 1. It is recommended that this recovery metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.			NOTE: 2. It is necessary to convert this value (X) to the <0,1> interval if making summarization of characteristics					

Table 7.2.3 Recoverability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Recovery	How long it will take to recover if system went down?	Let system to be down and compare estimated time with actual time to restart system.	Recovery time Mean time =X= Sum(T) / B T= Time to recovery downed software system at each opportunity B= Number of turns which observed software system downs entered into recovery	0<X The smaller is the better.	X=Ratio	T=Time B=Count X=Time/Count	Test report Operation report	5.3 Integration 5.3 Qualification testing 5.4 Operation 6.5 Validation	User Maintainer
	NOTE: 1. It is recommended to measure maximal time of the worst case or distribution of recovery time for many cases. 2. It is recommended that this recovery metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.			NOTE: 3. It is recommended to distinguish the grades of recovery difficulty, for example, recovery of destroyed data base is more difficult than recovery of destroyed transaction. 4. It is necessary to convert this value (X) to the <0,1> interval if making summarization of characteristics					
Restartability	Can user restart easily?	Let system to be down, and compare estimated time with actual time to restart system.	X = A / B A= Number of restarts which met to estimated 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, the user can restart easily.	X=Ratio	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
	NOTE: 1. It is recommended to estimate different time to restart to correspond to the severity level of down, such as data base destruction, lost multi transaction, lost single transaction, or temporal data destruction. 2. It is recommended that this recovery metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.								

Table 7.2.3 Recoverability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Restorability	How is the product capable to restore in defined cases ?	Let the system to perform defined cases	X= A / B A= Number of cases successfully restored B= Number of cases performed NOTE: It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.	0<=X<=1 The larger is the better, the product is more capable to restore in defined cases	X= Ratio	A= Count B= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation 6.5 Validation	User Maintainer
Restore effectiveness	How effective is the restoration process in the product ?	Let the system to perform defined cases	X=A/B A= Number of cases successfully restored meeting the target restore time B= Number of cases performed NOTE: It is recommended that this metric includes only the automatic recovery provided by the software and excludes the maintenance work of human.	0<=X<=1 The larger is the better, the restoration process in product is more effective.	X= Ratio	A= Count B= Count X= Count/ Count	Test report Operation report	5.3 Integration 5.3 Qualifica-tion testing 5.4 Operation 6.5 Validation	User Maintainer

Table 7.2.4 Compliance metrics (compliance for reliability)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measure ment	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfaction coverage of compliance items relating to reliability	How completely does the software adhere the standards, conventions or regulations relating to reliability?	Previously specify required compliance items based on standards, conventions or regulations relating to reliability which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	Ratio of satisfied compliance items relating to reliability X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE: 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.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specifica-tion of complian-ce and related standards, conven-tions or regulation s Test specifica-tion and report	5.3 Qualifica-tion testing 6.5 Validation	Supplier User

7.3 Usability Metrics

External usability metrics should be able to measure software attributes related to its operation, with regard to easiness of use and adaptation of new operators. Although the evaluation focus is the software, it's expected to be difficult to distinguish between actual software attributes, and the host system influence over usability characteristics. This does not invalidate the measurements, since the evaluated software is ran under explicitly specified conditions, which encompass the required system."

An external usability metric should be able to measure such attributes as user behavior, operator or system including software, to represent the extent of ease of use.

NOTE: User tests

Most external usability metrics are based on testing users. A sample of users who are representative of an identified user group should carry out the test without any hints or external assistance. 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 taken 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.

7.3.1 Understandability metrics

Understandability metrics should be capable of evaluating the behavior of users without previous knowledge on software operation and measure their difficulty on understanding software functions, operation and concepts. On doing this, it may be considered entities such as documentation (in all available formats, as on-line or printed), software interface and vocabulary. The condition of observing users without any background on the evaluated software is not always neccesarry, but is expected to be helpful on measuring the capability of the software to communicate with new users; at that point, usability problems are more related to software problems then to user psychological aspects, as stress on trying to operate the system.

An external understandability metric should be able to measure such attribute as users' effort by measuring the behavior of user who, before regularly use of software, try to understand:
a) what kind of tasks are performed or outputs produced by the software product;
b) what kind of users' manual tasks, operations, or inputs are needed to use the software.
These assist users to select the software product which is suitable to their intended use.

7.3.2 Learnability metrics

Learnability metrics should be capable of evaluating or drawing the user curve of performance on software operation, from a start point of no knowledge about the evaluated software.

The main measurable factors are time and success ratio on completing tasks by using software operations. When using prepared or chosen test tasks, the evaluator must be aware of the effects of inducing user behavior, like ordering those tasks in an increasing difficulty list.

Learnability measuring is strongly related to understandability; in the evaluation results analysis, and during the evaluation process, understandability measurements probably can be indicators of the learnability potential of the evaluated software."

An external learnability metric should be able to measure such attribute as the behavior of user who is learning how to use the software. Measurement may include measures derived from interviewing to user.

7.3.3 Operability metrics

An external operability metric should be able to measure such attribute as user's human behavior during operational testing, usability testing or user operation.

The followings should be remarked:

a) Operation testing should be performed and operation of users should be observed and analyzed;
b) The software or functions which are specified to be used infrequently or intermittently could be excluded (depending of the nature of the needs). Examples are emergency call function or restart function;
c) Frequency distribution for each of functions may be helpful to understand user's operational profile.

NOTE: When user's operation profile is comprehended, it is useful to apply usability testing to frequently used function with priority derived from profile, and to get close operability measurement results to user's actual evaluation.

NOTE: Users may be observed who are participants of operation testing.

7.3.4 Attractiveness metrics

An external attractiveness metric should be able to measure such attribute as user's human behavior expressing the extent of which user likes the software during operational testing, usability testing or user operation.

7.3.5 Compliance metrics

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

Table 7.3.1 Understandability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Completeness of description	What proportion of functions (or types of function) are understood after reading the product description?	User test	X = A/B A = Number of functions (or types of functions) understood B = Total of number of functions (or types of functions)	0<=X<= 1 The closer to 1.0 is the better.	X =Abs.	A=Count B=Count X=Count/ Count	Operation (test) report	5.3 Qualifica-tion testing 5.4 Operation	User Maintainer
Demonstration Availability	Whenever can user see demonstration?	Observe user behavior who is trying to see demonstration. Observation may employ human cognitive action monitoring approach with video camera.	User's demonstration success ratio X = (A / B) A= Number of turns which user successfully see demonstration when user attempts to see demonstration. B= Number of turns which user attempts to see demonstration during observation period.	0<=X<= 1 The closer to 1.0 is the better.	X =Abs.	A=Count B=Count X=Count/ Count	Operation (test) report User monitoring record (video tape and action record)	5.3 Qualifica-tion testing 5.4 Operation	User Maintainer
Evident functions	What proportion of functions (or types of function) can be identified by the user based upon start up conditions?	User test	X = A/B A = Number of functions (or types of functions) identified by the user B = Total of number of actual functions (or types of functions)	0<=X<= 1 The closer to 1.0 is the better.	X =Abs.	A=Count B=Count X=Count/ Count	Operation (test) report	5.3 Qualifica-tion testing 5.4 Operation	User Maintainer
Function understand-ability	What proportion of interface functions are understandable?	User test	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.	X=absol ute X=count /count	A=count B=count	Operation (test) report	5.3 Qualifica-tion testing 5.4 Operation	User Maintainer
Understandable Input and Output	Is easy for user to understand what is required as input data and what is provided as output by software system?	Observe user behavior who is trying to understand input and output of the software.	Understandable input and output data items ratio X=(A / B) A= Number of input and output data items which user success to understand during enough short time within a few trial use B= Number of input and output data items with which user attempts to understand during observation period	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Operation (test) report User monitorin g record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Maintainer

Table 7.3.2 Learnability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Ease of function learning	How long does the user take to learn to use a function?	User test	Time required to learn function T= Mean time taken to learn to use a function correctly NOTE: This metric is generally used as one of experienced and justified.	0<T The shorter is the better.	T=Ratio	T=Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Maintainer
Ease of performing task learning	How long does the user take to learn how to perform the specified task smoothly ?	Observe user behavior from they started to learn till became to operate smoothly.	Time required to learn operation to perform task 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.	T=Ratio	T=Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Maintainer
	NOTE: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.			NOTE: 2. Effort may alternatively represent time by person-hour unit.					
Ease of use of help system	Can the user find online help?	User test	X = A/B A = Number of tasks for which correct online help is located B = Total of number of tasks tested	0<=X The closer to 1.0 is the better.	X= Abs., Ratio	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 7.3.2 Learnability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Effectiveness of help system	What proportion of tasks can be completed correctly after using the help systems?	User test	X= A/B A= Number of tasks successfully completed after accessing online help B = Total of number of tasks tested	0<=X The closer to 1.0 is the better.	X= Abs., Ratio	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Human interface designer
Effectiveness of user documentation and help systems	What proportion of functions can be used correctly after reading the documentation or using help systems?	User test	X = A/B A = Number of functions that can be used B = Total of number of functions provided NOTE: This metric is generally used as one of experienced and justified metrics rather than the others.	0<=X,Y<=1 The closer to 1.0 is the better.	X= Abs., Ratio.	A= Count B= Count X= Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Human interface designer
Tutorial Readiness	Can user easily use tutorial?	Observe user behavior when they try to learn by themselves.	User's tutorial using success ratio X=(A / B) A= Number of turns which user successfully use tutorial functions when user attempts to use demonstration. B= Number of turns which user attempts to use tutorial functions during observation period.	0<=X<= 1 The closer to 1.0 is the better.	X =Abs.	A=Count B=Count X=Count/ Count Y=Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Maintainer
	NOTE: 1 (Complementary measurement formula) Function covered by tutorial ratio Y=(C / D) C= Number of functions of which tutorial functions are used well when user attempts to use. D= Number of functions of which user attempts to use tutorial functions during observation period.			NOTE: 2 It is recommended to determine previously functions which need tutorials for beginners and to examine later the extent of implementation.					

Table 7.3.3 Operability metrics

Conforms with operational user expectations									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Operational Consistency	How easily user send his/her intention to or receive his/her expecting results from software through what user see?	Observe user behavior who is operating software	1. Smooth Communicativeness Conformity to WYSIWYG (What You See Is What You Get) X = 1 - (A / B) A= Number of reports or functions which resulted within unacceptable conformity against user's expectation derived from what are visible at the screen B= Number of reports or functions which are visible at the screen	0<=X<=1 The closer to 1.0 is the better.	X= Abs.	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
			2. Easy to derive operation Frequency of expected results Y = N / UOT N= Number of operations which resulted within unacceptable conformity against user's expectation derived from experience of operation UOT= user operating time (observation period) NOTE: User's experience of operation is usually helpful to recognize several operation pattern which derives user's expectation.	0<=Y<= 1 The smaller and closer to 0.0 is the better.	Y= Abs.	UOT= time Y= Count/ Time			

Operability metrics (continued)

Controllable

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Error correction	Can user easily correct error on tasks?	User test	Time for Error Correction on Task T=TCC - TSC TCC = Time completing correct specified type errors of performing task TSC = Time starting correct specified type errors of performing task	0<T The shorter is the better.	T=Ratio	T=Time			
User operation cancelability	Can user easily recover his/her error or retry tasks?	Observe user behavior who is operating software	1. Frequency of cancel success X= A / UOT A= number of turns which user success to cancel their error operation UOT= user operating time during observation period. NOTE: 1. When function is tested each by each, the ratio can be also calculated, that is the ratio of number of functions which user success to cancel their operation to all functions.	0<=X<= 1 The higher and closer to 1.0 is the better.	X=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 user behavior who is operating software	2. User's successful input change ratio 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,Y<=1 The closer to 1.0 is the better.	X= Abs.	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 7.3.3 Operability metrics (continued)

Suitable for the task operation

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Default value availability	Can user easily select parameter values for his/her convenient operation?	Observe user behavior who is operating software. Count how many times user attempts to establish or to select parameter values and fails to do them, because user can not use default values provided by the software.	Frequency of default value available X = 1 - (A / B) A= The number of turns which 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 turns which user attempt to establish or to select parameter values NOTE: 1) It is recommended to take accounts of operator's behavior 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 which covers parameter setting functions	0<=X<= 1 The closer to 1.0 is the better.	X=Ratio	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
User operating time adequacy	Is user become to deal task with very short operating time after sufficient training?	Observe user behavior who is operating software before/after sufficiently trained.	1. Ultimate Operating Time Tul = Operating time needed to perform a specified task for ultimately short time skillful or trained user) 2. Beginner's Operating Time Tbe = Operating time needed for beginner to perform a specified task 3. Reminder's Operating Time Tre = Operating time needed for reminder to perform a specified task	0<Tul 0<Tbe 0<Tre 0<T The shorter is the better.	Tul, Tbe, Tre and T= Ratio	Tul, Tbe, Tre, Tf,Ts and T = Time	Operation (test) report User monitor record	6.5 Validation 5.3 Qualification testing 5.4 Operation	User Human interface designer

4. Operating time needed to complete task
Time to complete a task of user's attempt
T= Tf – Ts
Tf = Time completing a specified task
Ts= Time starting operation for task

NOTE:
1. Beginner means that a specified task has been never done by that user or he/she is beginner of the software.

NOTE:
2. Reminder means that task is used only few times with very long intervals and user may forget details of operation.

Operability metrics (continued)

Self descriptive (Guidable)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Guidability	Is everyone adequately guided to success his/her intended task?	Observe user behavior who is operating software Count how many tasks does user attempt and fail/success.	1. Interactive guidability User's attempt task success ratio X= (A / B) A = Number of tasks of which objectives are completed successfully by user with on-line interactive guide B = Number of tasks attempted by user NOTE: It is recommended to investigate the followings with this metric: - Can user complete his/her intended task, even if it is the first attempt to do that? - Can beginner user complete his/her intended task?	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	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
	Is everyone adequately guided to success his/her intended task?	Count how many users fail/success, who are participants of operation testing.	2. Common user guidability Ratio of task success users Y = (C / N) C = Number of users successfully completing task N = Number of users who attempted to do the task NOTE: Common user may be ordinary people or trained operator. Users may be observed who are participants of operation testing.	0<=Y<= 1 The closer to 1.0 is the better.	Y= Abs.	C=Count N=Count Y=Count/ Count			

Observe user's fail or hesitation during operation.	3. User understandable status or progress Frequency of user's fail to understand status or progress Z = D / UOT D= number of turns which user pause for a long period or repeat fail successively at the same operation, because of user not comprehending status or progress of executing task UOT= user operating time (observation period) NOTE: 1. It is recommended to take accounts of operator's behavior and decide how long period is not allowable to pause as "long period". 2. When function is tested each by each, the ratio of allowable function can be also calculated.	0<=Z<= 1 The smaller and closer to 0.0 is the better.	X= Abs.	D = Count UOT= Time Z = Count / Time

Operability metrics (continued)

Self descriptive (Guidable)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Message readiness	Can user easily understand messages from software system? Is there any message which brought delay for user to understand and to start next action? Can user easily memorize important message?	Observe user behavior who is operating software	Comprehensive messages Frequency of user's comprehending message X = A / UOT A = number of turns which user pause for a long period or repeat fail successively at the same operation, because of user not comprehending messages UOT = user operating time (observation period)	0<=X<= 1 The smaller and closer to 0.0 is the better.	X=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

NOTE:

1. The extent of easiness of message comprehending is represented by how long that message brought delay for user to understand and to start next action. Therefore, it is recommended to take accounts of operator's behavior and decide how long period is not allowable to pause as "long period".

2. It is recommended to investigate the followings as one of the problems of user's comprehending messages.

1) Attentiveness:

Attentiveness implies that user successfully recognize attentive important messages presenting such the next user action guidance, name of data items to be looked, and warning for careful operation.

- Can user never fail to watch when user is encountering attentive important messages?
- Can user avoid to mistake operation, because of user's recognizing attentive important messages?

2) Memorability:

Memorability implies that user memorize important messages presenting such the next user action guidance, name of data items to be looked, and warning for careful operation.

- Can user easily memorize important message?
- Is it helpful for user to keep user's remembrance?
- Is it required for user to remember only few and not so much?

3. When message is tested each by each, the ratio of comprehensive messages to the total can be also calculated.

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

Operability metrics (continued)

Self descriptive (Guidable)									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Self-explanatory error messages	In what proportion of error conditions does the user propose the correct recovery action?	User test	X=A/B A=Number of error conditions for which the user proposes the correct recovery action B=Number of error conditions tested NOTE: This metric is generally used as one of experienced and justified.	0 <= X <= 1 The closer to 1, the better.	X=absolute	X=count/count A=count B=count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Human interface designer

Operability metrics (continued)

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	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Operational error recoverability	Can user easily recover his/her worse situation?	Observe user behavior who is operating software	recoverable situation frequency X = 1 – (A / B) A= Number of unsuccessfully recovered situation (after a user error or change) and user was not informed about this risk by the system B= Number of user errors or changes	0<X The less is the better.	X=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
Time Between Human Error Operations	Can user operate the software long enough without human error?	Observe user behavior who is operating software	Time Between Human Error Operations X = Mean time = 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.	X=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

NOTE:
1. Human error operation may be detected by counting below user's behavior :
a) Simple human error: The number of turns which user just simply mistakes to input operation;
b) Intentional error: The number of turns which user repeats fail several times at the same operation with miss-understanding during observation period;
c) Operation hesitation pause: The number of turns which user pause for a long period with hesitation during observation period.
NOTE:
2. It is seemed that operation pause implies user is puzzled and hesitate operation.
It depends on function, operation procedure, application domain and user, whether it is long period or not for user to pause operation. Therefore, evaluator is requested to take account into them and determine threshold time. For an interactive operation, it is generally supposed that "long period" threshold range is from 1min. to 3 min.

Operability metrics (continued)

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	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Undoability	How frequently does the user successfully correct input errors?	User test	Input undoability X=A/B A= Number of input errors which the user successfully corrects B= Number of attempts to correct input errors NOTE: This metric is generally used as one of experienced and justified.	0<=X,Y<=1 The closer to 1.0 is the better.	X= Abs.	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?	User test	Error undoability X=A/B A= Number of error conditions which the user successfully corrects B= Total number of error conditions tested	0 <= X <= 1 The closer to 1, the better.	X=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

Operability metrics (continued)

Suitable for individualisation

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Customisability	Can user easily customize operation procedures for his/her convenience? Can user, who instructs end users, easily set customized operation procedure templates for preventing his/her error? What proportion of functions can be customised?	User test	X=A/B A= Number of functions successfully customised B= Number of attempts to customise	0 <= X <= 1 The closer to 1, the better.	X=absolute	X=count/count A=count B=count			

NOTE:
1. Ratio of user's fail to customize may be measured.
Y = 1 - (C / D)
C = Number of turns which a user fail to customize operation
D = Total number of turns which a user attempted to customize operation for his/her convenience.
0<=Y<= 1, The closer to 1.0 is the better.

2. It is recommended to regard the followings as variations of customise operation:
- chose alternative operation, such as to use menu selection or command input;
- combine user's operation procedure, such as to record and edit operation procedure;
- set constrained template operation, such as to program procedures or to make template for input guidance.

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

Operability metrics (continued)

Suitable for individualisation									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
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 customizing operation.	Operation Procedure Reduction Ratio X = 1 - (A / B) A = Number of reduced operation procedures after customizing operation B = Number of operation procedures before customizing operation. NOTE: 1. It is recommended to take samples for each different user task and to distinguish operator who is skillful user or beginner. 2. Number of operation procedures may be represented by counting operation strokes such click, drug, key touch, screen touch etc.	0<=X< 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Human interface designer

Table 7.3.4 Attractiveness metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Attractive interface	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						
Interface appearance customisability	What proportion of interface elements can be customised in appearance to the user's satisfaction?	User test	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 NOTE: This metric is generally used as one of experienced and justified.	0 <= X <= 1 The closer to 1, the better.	X= Abs.	X= Count/ Count A= count B= count			
User operational frequency	Does user like to use software frequently?	Observation of usage	User's operational frequency X = N / OT N= number of turns which user use the specific software functions OT= operation time NOTE 1. User may be enforced to use the software frequently whether user likes it or not 2. It is recommended to ensure whether user actually feels good and is in favor of using the software by interviewing end user.	0<X The more is the better.	X=Ratio	N= Count, OT= Time X= Count/ Time	Operation (test) report User monitoring record	6.5 Validation 5.3 Qualifica-tion testing 5.4 Operation	User Human interface designer

NOTE:
The following complementary metrics may be used.
1) Attractive interaction
Frequency of user's feeling attractive
X = A / T
Ratio of attractive presentation
Y = (A / B)

A= Number of turns which user feel attractive during operation
B= Number of turns which user is encountering presentations
T= User operating time (observation period)
0<=X The larger is the better.
0<=Y<= 1 The closer to 1.0 is the better.

2) Favorable input/output expression selection availability
Favorable input/output expression selection availability X= 1 - (A / N)

A= Number of turns which user failed to select input/output expression
N= Number of turns which user tried to select input/output expression

Input/output expression may include multiple modal expressions such are fill-in-the-box, button, list-box, text, graph, map, pictogram, color, visual picture, voice recognition/reading and so on.
0 <= X <= 1 The closer to 1.0 is the better.

3) Favorable user popularity
Favorable user popularity
X= A / B * 100(%)
A= Number of people who replied favorable answer
B= Number of people who answered questionnaires

0 <= X <=100(%) The closer to 100% is the better.

Make questionnaires to user who once operate the software and analyse the answers statistically.

Table 7.3.5 Compliance metrics (compliance for usability)

Metric Name	Purpose	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfaction coverage of compliance items relating to usability	How completely does the software adhere the standards, conventions, style guides or regulations relating to usability?	Previously specify required compliance items based on standards, conventions, style guides or regulations relating to usability which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	Ratio of satisfied compliance items relating to usability X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE: 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.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specification of compliance and related standards, conventions, style guides or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

7.4 Efficiency metrics

An external efficiency metric should be able to measure such attribute as the behavior of computer system including software during testing or operating.

It is recommended that the maximal and distribution time are to be investigated for many cases of testing or operating, because the measure is affected strongly and fluctuated by condition of use, such are 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 which are required in specification.

It is recommended to list and to investigate the role played by factors such as "CPU" and memory load by other software's, network traffic, 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 is identified and defined to be suitable for software application, for examples: 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; an output of data produced by user callable function for common user application.

NOTE:
Response time: Time needed to get the result from pressing a transmissionkey. This means that response time includes processing time andtransmission time. Response time is applicable only for a interactive system. There isno 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.

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

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

7.4.1 Time behavior metrics

An external time behavior metric should be able to measure such attribute as the time behavior of computer system including software during testing or operating.

7.4.2 Resource utilization metrics

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

7.4.3 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 of the software product to failing to adhere to standards, conventions or regulations relating to efficiency which are required to be adhered.

Table 7.4.1 Time behavior metrics

Response time									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Response time	What is the wait time the user experiences when issuing a request and for the application to finish the process? How long does it take for users to wait for next response from the software?	Start a specified task. Measure the time it takes for the sample to complete its operation. Keep a record of each attempt.	Response time T = (time of gaining the result) - (time of command entry finished) NOTE: It is recommended to take account of time bandwidth and to use statistical analysis with measures for a lot of tasks (sample shots) not for only one task.	0 < T The sooner is the better.	T= Ratio	T= Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Mean response fulfillment ratio	What is the average wait time the user experiences once issuing a request and its completion within the specified load upon the system in terms of concurrent tasks and system utilisation?	Execute a number of concurrent tasks (or sample shots of these tasks). Measure the time it takes to complete the selected operation(s) in the given traffic. Keep a record of each attempt.	Mean response time satisfaction 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) NOTE: 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 aspect of human ergonomics may be considerable.	0 <= X The nearer to 1 and less than 1 is the better.	X= Ratio	X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.1 Time behavior metrics (Continued)

Response time									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
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 enough short time 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) NOTE: 1. Distribution may be measured like 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,Y The nearer to 1 and less than 1 is the better.	X=Ratio	X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.1 Time behavior metrics (Continued)

Throughput									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Throughput time	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 / B A = number of completed tasks B = unit of time	0 < X The larger is the better.	X= Ratio	X= Count/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Mean throughput fulfillment ratio	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.	X= Ratio	X = Time / Time	Testing report Operation report showing elapse time	5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.1 Time behavior metrics (Continued)

Throughput									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
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 Tmax = MAX(Xi) (for i = 1 to N) Rmax = required maximum throughput. MAX(Xi) = number of job tasks for i-th evaluation. 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. NOTE: 1. Distribution may be measured like 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	0 < X,Y The larger is the better.	X= Ratio	X = Time / Time	Testing report Operation report showing elapse time	5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA

Table 7.4.1 Time behavior metrics (Continued)

Turnaround time									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Turnaround time	What is the wait time the user experiences when issuing an instruction to start a group of related tasks and the completion of these related tasks?	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 of user's finishing getting output result) - (Time of user's finishing request) NOTE: It is recommended to take account of time bandwidth and to use statistical analysis with measures for a lot of tasks (sample shots) not for only one task (shot).	0 < T The shorter the better.	T= Ratio	T= Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Mean turnaround fulfillment ratio	What is the average wait time the user experiences when issuing an instruction to start a group of related tasks and the completion within a specified load the system has 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.	X= Ratio	X= Time/ Time	Testing report Operation report showing elapse time	5.3 Sys./Sw. Integration 5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.1 Time behavior metrics (Continued)

Turnaround time									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
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) NOTE: 1. Distribution may be measured like 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	0 < X,Y The nearer to 1 and less than 1 is the better.	X= Ratio	X= Time/ Time Tmax = Time Tdev = Time	Testing report Operation report showing elapse time	5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.2 Resource utilisation metrics

I/O devices resource utilization

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
I/O devices utilisation satisfaction	Is software system capable to perform tasks without using I/O devices long time?	Execute concurrently a lot of tasks and observe time of I/O devices occupied.	I/O devices utilisation 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 is the better.	X= Abs.	A= Time B= Time X= Time/ Time	Testing report Operation report	5.3 Qualification testing 5.4 Operation Maintenance	Developer Maintainer SQA
Mean I/O fulfillment 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 messages for ith evaluation N = number of evaluations	0<= X The smaller is the better	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA
User waiting time of I/O devices utilisation	Can user perform tasks without waiting for finish of I/O devices operation?	Execute concurrently a lot of tasks and observe time of I/O devices occupied with user environment.	User waiting time of I/O devices utilisation T T = Time spent to wait for finish of I/O devices operation NOTE: 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.	0 < T The shorter is the better.	X= Ratio.	T= Time	Testing report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA

Table 7.4.2 Resource utilisation metrics (continued)

I/O devices resource utilization									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Visible I/O utilisation	How many I/O 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 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	X= Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA
Worst case I/O utilisation	What is the absolute limit on I/O 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 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	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA

Table 7.4.2 Resource utilisation metrics (continued)

Memory resource utilization

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Mean memory fulfillment ratio	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 messages. Ai = number of memory messages for i-th evaluation N = number of evaluations	0<= X The smaller is the better	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA
Visible memory utilisation	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	X= Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Maintainer SQA
Worst case 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 messages MAX(Ai) = Maximum number of memory messages from 1st to i-th evaluation. N= number of evaluations.	0<= X The smaller is the better	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Developer Maintainer SQA

Table 7.4.2 Resource utilisation metrics (continued)

Transmission resource utilization									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Mean transmission fulfillment ratio	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	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Transmission capacity utilisation satisfaction	Is software system capable to perform tasks within expected transmission capacity?	Execute concurrently specified tasks with multiple users, observe transmission capacity and compare specified one.	Transmission capacity utilisation satisfaction X = A / B A = transmission capacity B = specified transmission capacity which is designed to be used by the software during execution. NOTE: It is recommended to measure dynamically peaked value with multiple users.	0 <= X <= 1 The less than and nearer to the 1 is the better.	X= Abs.	A= Size B= Size X= Size / Size	Testing report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer SQA
Visible transmission utilisation	How many transmission error messages were experienced over a set period of time and specified resource utilisation?	Calibrate the test conditions. Emulate a condition whereby the system to 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	X= Ratio	A = Count T = Time X = Count/ Time	Testing report Operation report showing elapse time	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Maintainer SQA

Table 7.4.2 Resource utilisation metrics (continued)

Transmission resource utilization									
Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Worst case transmission utilisation	What is the absolute limit on transmission required in 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	X = Ratio	X = Count/ Count	Testing report Operation report showing elapse time	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	User Developer Maintainer SQA
Visible synchronisation	What is the degree of synchronisation between dissimilar media over a set period of time?	Calibrate the test conditions. Emulate a condition whereby the system to 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 = unit of time	The smaller the better	X = ratio	A = size T = Time X = ratio	Testing report Operation report showing elapse time		User Maintainer SQA

Table 7.4.3 Compliance metrics (compliance for efficiency)

Metric Name	Purpose	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfaction coverage of compliance items relating to efficiency	How completely does the software adhere the standards, conventions or regulations relating to efficiency?	Previously specify required compliance items based on standards, conventions or regulations relating to efficiency which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	Ratio of satisfied compliance items relating to efficiency X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE: 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.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specification of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

7.5 Maintainability metrics

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

7.5.1 Analyzability metrics

An external analyzability metric should be able to measure such attributes as maintainer's or user's effort or spent resources when they try to diagnose for deficiencies or causes of failures, or for identification of parts to be modified.

7.5.2 Changeability metrics

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

7.5.3 Stability metrics

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

7.5.4 Testability metrics

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

7.5.5 Compliance metrics

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

Table 7.5.1 Analysability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Diagnostic function support	How many causes of failures are identified by the diagnostic function ? Can user identify 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 behavior of user or maintainer who is trying to resolve failures using diagnostics function	X= A / B A= Number of failures of which maintainer can diagnostic (using the diagnostics function) to understand cause effect relation B= Total number of registered failures	0<=X<= 1 The closer to 1.0 is the better.	X=Abs	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Data recording during operation	Can user identify specific operation which caused failure? Can maintainer easily find specific operation which caused failure?	Observe behavior of user or maintainer who is trying to resolve failures.	Activity recording during operation ratio 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.	X=Abs.	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.5.1 Analysability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Failure analysis time	Can user easily analyse cause of failure? (User sometimes performs maintenance by setting parameter.) Can maintainer easily find cause of failure? How easy to analyse the cause of failure?	Observe behavior of user or maintainer who is trying to resolve failures.	X= Sum(T) / N T= Tout - Tin Tout = Time at which the causes of failure are found out (or reported back to user) Tin = Time at which the failure report is received N= Number of registered failures	0<=X The shorter is the better.	X=Abs and Ratio	X= Time/ Count T= Time	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
	NOTE: 1. It is recommended to measure maximal time of the worst case and time bandwidth to represent deviation. 2. It is recommended to exclude number of failures of which causes are not yet found when measurement is done. Though, the ratio of such obscure failures should be also measured and presented together.			NOTE: 3. From individual user's view, time is concerned, while effort may also concerned from maintainer's view. Therefore, person-hours may be used instead of time.					
Finding results of failure case	Can user identify specific operation which caused failure? Can maintainer easily find cause of failure?	Observe behavior of user or maintainer who is trying to resolve failures.	Failure cause finding success ratio 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.	X=Abs.	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Status monitoring during operation	Can user identify specific operation which caused failure? Can maintainer easily find cause of failure?	Observe behavior of user or maintainer who is trying to resolve failures.	Successful status monitoring ratio X= 1- (A / B) A= Number of turns which maintainer (or user) failed to get monitor data B= Number of turns 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.	X= Abs.	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.5.2 Changeability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Change recordability	Can the user easily identify revised versions? Can the maintainer easily change the software to resolve problems?	Observe the behavior of user or maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	Change recording ratio 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 is 0 the less changes have taken place	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator
Ease of parameterisation	Can the user or the maintainer easily change parameter to change software and resolve problems?	Observe behavior of the user or the maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	Change by using parameter success ratio X=1- (A / B) A= Number of turns which maintainer fails to change software by using parameter B= Number of turns which maintainer attempts to change software by using parameter	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator User
Readiness for change	Can the maintainer easily change the software to resolve problem?	Observe behavior of maintainer who is trying to change the software. Otherwise, investigate problem resolution report or maintenance report and product description.	Required effort (in person hours) to change software T = Sum (A / B) / N A= Work time spent to change B= Changed software size N= Number of changes NOTE: 1. A changed software size may be changed executable statements of program code, number of changed items of requirements specification, or changed pages of document etc.	0<T The shorter is the better or the required number of changes were excessive	T= Ratio.	T= Time A= Time B= Size N= Count	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator

Table 7.5.2 Changeability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Time spent to implement the change for user's satisfaction	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.	Time spent to update user's version Average Time = 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<Tu The shorter is the better., except of the number of revised versions was large.	Tu= Ratio	Tu= Time Trc, Tsn = Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	User Maintainer Operator
Time spent to implement change by the maintainer	Can the maintainer easily change the software to resolve the failure problem?	Observe the behavior of the user and maintainer while trying to change the software. Otherwise, investigate problem resolution report or maintenance report.	Time to change for maintainer Average Time = 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<Tm The shorter is the better, except of the number of failures was large.	Tm= Ratio	Tm= Time Tin, Tout = Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator

NOTE: 1. It is recommended to measure maximal time of the worst case and time bandwidth to represent deviation.
2. It is recommended to exclude number of failures of which causes are not yet found when measurement is done. Though, the ratio of such obscure failures should be also measured and presented together.
3. From individual user's view, time is concerned, while effort may also concerned from maintainer's view. Therefore, person-hours may be used instead of time.

Table 7.5.3 Stability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Less encountering failures after change	Can user operate software system without failures after maintenance? Can maintainer easily mitigate failures caused by maintenance side effects?	Observe behavior 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.	Frequency of encountering failures after change X= Na / Ta Fluctuated frequency of encountering failures before/after change Y = {(Na / Ta) / (Nb / Tb) } Na = Number of turns which user encounters failures during operation after software was changed Nb = Number of turns 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 is the better. The closer to 0.0 is the better.	X,Y= Ratio.	A= Count T= Time X= Count/ Time	Problem resolution report Maintenance report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator
	NOTE: 1. User may need specified period to determine side effects of software changes, when revision-up of software is introduced for resolving problems. 2. It is recommend to compare this frequency before and after change. 3. If changed function is identified, it is recommended to sort out whether encountered failures are detected in the changed function itself or the any other ones. The extent of impacts may be rated for each failure.								

Table 7.5.3 Stability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Localisation of modification (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.	Chaining failure emerging per resolved failure X= A / N A= Number of failures emerged after failure is resolved by change during specified period N= Number of resolved failures NOTE: It is recommend to give precise measure by checking whether cause of current failure is attributed to change for previous failure resolution, as possible.	0<X The smaller and the closer to 0.0 is the better.	X= Abs.	A= Count N= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.5.4 Testability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Effortless testing	Can user and maintainer easily perform operational testing and determine whether the software is ready to operation or not?	Observe behavior of user or maintainer who is testing software system after maintenance.	Time (effort) to test after failure resolution Average X= Sum(T) / N T = Time spent to test to make sure whether reported failure which was resolved or not. N= Number of resolved failures NOTE: If failure which is not resolved or degraded, exclude them and separately measure ratio of such failures.	0<X The smaller is the better.	X= Ratio	T= Time N= Count X= Time /Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator
Readiness of built-in test function	Can user and maintainer easily perform operational testing after maintenance and get ready for operation?	Observe behavior of user or maintainer who is testing software system after maintenance.	Built-in test function use success ratio X= A / B A= Number of turns which maintainer can use suitably built-in test function B= Number of turns of test opportunities NOTE: Examples of built-in test functions include simulation function, pre-check function for ready to use and so on.	0 < X <1 The larger and the closer to 1.0 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator
Test restartability	Can user and maintainer easily perform operational testing with checking step by step after maintenance?	Observe behavior of user or maintainer who is testing software system after maintenance.	Pause and restart of executing test run X = A / B A = Number of turns which maintainer can pause and restart executing test run at desired points to check step by step B= Number of turns of pause of executing test run	0 < X <1 The larger and the closer to 1.0 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualification testing 5.4 Operation 5.5 Maintenance	Developer Maintainer Operator

Table 7.5.5 Compliance metrics (compliance for maintainability)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measure ment	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfaction coverage of compliance items relating to maintainability	How completely does the software adhere the standards, conventions or regulations relating to maintainability?	Previously specify required compliance items based on standards, conventions or regulations relating to maintainability which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	Ratio of satisfied compliance items relating to maintainability X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE: 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.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specifica-tion of complian-ce and related standards, conven-tions or regulation s Test specifica-tion and report	5.3 Qualifica-tion testing 6.5 Validation	Supplier User

7.6 Portability metrics

External portability metrics should be able to measure such attribute as the behavior of the operator or system during the porting activity.

7.6.1 Adaptability metrics

External adaptability metric should be able to measure such attribute as the behavior of user who is trying to adapt software to different specified environments. When 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.

7.6.2 Installability metrics

External installability metrics should be able to measure such attribute as the behavior of user who is trying to install the software in a user specific environment.

7.6.3 Replaceability metrics

External replaceability metrics should be able to measure such attribute as the behavior of user who is trying to use the software in place of other specified software in the environment of that software.

7.6.4 Co-existence metrics

External co-existence ability metrics should be able to measure such attribute as the behavior of user who is trying to the software with other independent software in a common environment sharing common resources.

7.6.5 Compliance metrics

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

Table 7.6.1 Adaptability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Adaptable data	Can user or maintainer easily adapt software to data sets in new environment?	Observe user's or maintainer's behavior when user is trying to adapt software to operation environment?	Limitation free operation after data adaptation X = (A / B) A = The number of data which are operable and not observed being incomplete operations caused by adaptation limitations; B= The number of data which is expected to be operable in the environment to which the software is adapted. NOTE: These data mainly include data files, data tupples or databases to be adapted to different data volumes, data items or data structures when for example, business scope is extended.	0<X<1 The larger and close to 1.0 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Environmental adaptability (Organization adaptability to infrastructure of organization)	Can user or maintainer easily adapt software to environment? Is software system enough capable to adapt itself to operation environment?	Observe user's or maintainer's behavior when user is trying to adapt software to operation environment?.	Adaptability to environmental business environments X= 1 - (A / B) A= Number of operated functions of which tasks were not completed or not enough resulted to meet adequate level during operation testing with user's business environment B= Total number of functions which were tested. NOTE: It is recommended to conduct testing which takes account of varieties of combination of infrastructure components of possible user's business environments.	0<X<1 The larger is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.6.1 Adaptability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Environmental hardware adaptability (adaptability to hardware devices and network facilities)	Can user or maintainer easily adapt software to environment? Is software system enough capable to adapt itself to operation environment?	Observe user's or maintainer's behavior when user is trying to adapt software to operation environment?.	Adaptability to environmental hardwares X= 1 - (A / B) A= Number of operated functions of which tasks were not completed or not enough resulted to meet adequate level during co-operating testing with environmental hardwares B= Total number of functions which were tested. NOTE: It is recommended to conduct overloaded combination testing with environmental hardwares which are possibly co-operated in variety user operation environments.	0<X<1 The larger is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Environmental software adaptability (adaptability to OS, network software and co-operated application software)	Can user or maintainer easily adapt software to environment? Is software system enough capable to adapt itself to operation environment?	Observe user's or maintainer's behavior when user is trying to adapt software to operation environment?.	Adaptability to environmental softwares X= 1 - (A / B) A= Number of operated functions of which tasks were not completed or were not enough resulted to meet adequate level during co-operating testing with operating system softwares or concurrent application softwares B= Total number of functions which were tested. NOTE: It is recommended to conduct overloaded combination testing with operating system softwares or concurrent application softwares which are possibly co-operated in variety user operation environments.	0<X<1 The larger is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.6.1 Adaptability metrics (continued)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
User effortless adaptation	Can user or maintainer easily adapt software to environment?	Observe user's or maintainer's behavior when user is trying to adapt software to operation environment?.	User effort required to adapt to user's environment T=Sum of user operating time to be spared to complete adaptation of the software to user's environment, when user attempt to install or change setup. (Person-hour may be used instead of time.)	0<T. The shorter is the better.	T=Ratio	T=Time	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.6.2 Installability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Easiness of Setup Re-try	Can user or maintainer easily re-try setup installation of software?	Observe user's or maintainer's behavior when user is trying to re-try setup installation of software?	X = 1 - (A / B) A = Number of turns which user fail to re-try setup during setup operation B = Total number of turns which user attempt to re-try setup during setup operation NOTE: 1. This metric is suggested as experimental use.	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
Operational installation flexibility	Can user or maintainer easily install software to operation environment?	Observe user's or maintainer's behavior when user is trying to install software to operation environment	X = A / B A = Number of turns which a user succeeded to change install operation for his/her convenience B = Total number of turns which a user attempted to change install operation for his/her convenience NOTE: 1. This metric is suggested as experimental use.	0<=X<= 1 The closer to 1.0 is the better.	X= Abs.	A=Count B=Count X=Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.6.2 Installability metrics (continued)

NOTE: The following complementary metrics may be used. 1) Effortless installation User manual actions for installation X = A A= The number of user manual actions for installation 0<X The smaller is the better. 2) Installation easiness Installation supporting level X = A A is rated with, for example: - Just only executing installation program and nothing more is needed (excellent); - Instucting function guide for installation (good); - Source code of program is needed to be modified 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 reduced B = Number of install operation procedures normally 0< X <1 The closer to 1.0 is the better. 4) Easiness of user's manual install operation Easiness level of user's manual install operation S = Score of easiness level of user's manual operation Examples of easiness level are following: [very easy] only user's watching except just start install or setup functions; [easy] only user's answering to question from install or setup functions; [not so easy] user's looking up parameters from tables or fill-in-boxes to be changed and setting them; [complicated] user's seeking parameter files, looking up parameters from files to be changed and writing them. X= Direct Interpretation of measured value

Table 7.6.3 Replaceability metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Data continuation	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 behavior when user is replacing software to previous one?	Data continuously use ratio 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.	X=Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator
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 behavior when user is replacing software to previous one?	Function inclusion ratio X = A / B A = number of functions which produce as enough similar results as used to be produced and of which changes have not to be required; B = number of tested functions which are similar to functions provided by other software to be replaced.	0<= X <=1 The larger is the better.	X=Abs.	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer Operator

Table 7.6.4 Co-existence metrics

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Concurrent multiple software use with less constraints	How often user encounter any constraints or unexpected failures when user operate concurrently other software?	Use concurrently the software to be evaluated and several other software which user often use.	Concurrent multiple software usable ratio X = A / T A = Number of any constraints or unexpected failures which user encounter during operating concurrently other software T = Time to operate concurrently other software	0<=X<= 1 The closer to 0 is the better.	X= Ratio	A= Count B= Count X= Count/ Count	Problem resolution report Operation report	5.3 Qualifica-tion testing 5.4 Operation 5.5 Mainte-nance	Developer Maintainer SQA Operator

Table 7.6.5 Compliance metrics (compliance for portability)

Metric Name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Perspective Beneficer
Satisfaction coverage of compliance items relating to portability	How completely does the software adhere the standards, conventions or regulations relating to portability?	Previously specify required compliance items based on standards, conventions or regulations relating to portability which to be adhered by software. Design test cases in accordance with compliance items. Conduct functional testing for these test cases.	X = 1 – (A / B) A= Number of failed compliance items during testing B= Number of total compliance items NOTE: 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.	0<= X <=1 The closer to 1 is the better.	X= Abs.	A= Count B= Count X= Count/ Count	Specification of compliance and related standards, conventions or regulations Test specification and report	5.3 Qualification testing 6.5 Validation	Supplier User

Annex A (Informative)
Descriptions of the metrics tables

The purpose of these descriptions is to clarify the reader comprehension for the quality metrics formulas proposal in the clause 7 in the technical reports 9126-2, 3and 4.

A.1 Metrics name

Metrics name characterizes measureable attribute of software and represents a unique or group of measurements.

Metrics name has the same or similar name in internal, external and quality in use metrics, when they are intended to be mutually corresponded respectively.

A.2 Purpose of the metrics

This helps to identify what user of metric can know by using the metric.

NOTE: This is described as a questionary style to look up easily in accordance with Goal / Question / Metric flamework.

A.3 Method of application

This helps to understand what way are useful and recommended to apply metrics.

A.4 Measurement, formula and data element computations

This helps to understand what kind of measurement, formula and data element are used to compute measure.

A.5 Interpretation of measured value

This helps to understand the range of measured value and the interpreted better range.

A.6 Scale types

The following measurement 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 comparision. The average, ratio or difference values may have no meaning for some measures. Such scale types are: Nominal scale, Ordinal scale, Intervals scale, Ratio scale, Absolute scale. M'=F(M), where F is the admissible function, explain what the admissible function is (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 compare mutually frequency of each other type respectively.

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 frequency of each mapped order. Therefore, the ratio and 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 mutual frequency of each order.

c) **Intervals scale**

M'=aM+b (a>0)

This includes artificial rating scales, for example, rating scales of sensitive questionnaire for asking about usability. These rating scales are comparable and an average has meaning only if it is calculated with the same rated scales for the same asking, but a ratio of such rating scales has no meaning. Even when double scores, it is pointed out only that a score is twice as much as the another, but not that anything is double.

d) **Ratio scale**

M'=aM (a>0)

This includes time interval, for example, time between software failures occurred. An average and a ratio has meaning respectively and they give actual meaning to the values. When the value is double, it is pointed out it takes twice as long as the another.

e) **Absolute scale**

M'=M

This includes density and frequency, for example, detected software fault density. An average has no meaning, while a ratio has meaning. In a 2 Kstep program, 20 faults was detected and in another 8Kstep, 160 faults was detected, resulting 10 faults / Kstep and 20 faults / Kstep respectively. It may seem that the average measure is 15 faults / Kstep, but actual average is 180 faults for 10 kilo step code, that is, 18 faults / Kstep.

A.7 Measurements types

For designing procedure for collecting data, interpreting fair meanings, and normalizing measures for comparison, a user of metrics should identify and take account of measure type of measurement employed by a metric.

A.7.1.Size measure type

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

NOTE: that 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 in this report can be used for software quality metrics.

A.7.1.1 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:

- the purpose for measuring the software size (It influences the scope of the software included in the measurement);
- 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-Part1.

In order to use Functional Size for normalization a quality assessor needs 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 an FSM Method and compliant with ISO/IEC 14143-Part1. 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.

A.7.1.2 Program size type

In this clause, the term ‘programming’ represents the number of executions resulting in an action, and the term ‘language’ represents the type of expression used.

1) Program source size

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

a) Non-comment source statements (NCSS)

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

b) New program size

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

c) Language use

For a same executable statement, the program size depend on the language use.

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

NOTE: 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 needed to distinguish a type of statements of source code into more detail like followings.

i) Statement type

- **Logical Source Statement(LSS).** The LSS measures the number of software instructions. The statements are irrespective of its 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 by the following so-called Halstead's measure:

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

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

3) Number of modules

The measurement is counting the number of modules of a program.

A.7.1.3 Utilized resource size measure type

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

a) **Amount of memory** ex.) amount of disk memory occupied temporally or stable during the software execution;

b) **I/O load** ex.) bit size of communication data (meaningful for backup tools on a network);

c) **CPU load** ex.) percentage of occupied CPU instruction sets per second (meaningful for CPU utilization and efficiency of process distribution in multi-thread software's running on concurrent/parallel systems);

d) **Files and data records** ex.) bit size of file or record;

e) **Documents** ex.) number of document pages.

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

A.7.1.4 Specified operating procedure step type

This type identifies static steps of procedure 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 procedure.

A.7.2 Time measure type

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

The user of metrics should be aware that there are many ways in which time can be measured as a unit, including following:

a) Real time unit

This is 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 site but all the sites running, because whole sites are involved into one system. This unit is usually used for describing system reliability, for example, system failure rate.

A.7.2.1 System operation time type

System operation time type provides a basis for measuring time of 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 time measurement is done just on the periods the software is active (this is obviously extended to continuous operation).

a) Elapsed time

When use 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 power and applying a correction factor.

A.7.2.2 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 analyzed and mean, deviation or maximal should be computed. The execution under the specific conditions, particularly overloaded condition, should be examined. Execution time measure type is mainly used for efficiency evaluation.

A.7.2.3 User time type

User time type is measured upon time periods spent by individual user 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 behavior 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) **User operating time (task time)**

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

c) **User time**

Time spent by an individual user to use the software from the getting started to now. (Approximately, it is how many hours or days user uses the software from beginning.)

A.7.2.4 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 developer, maintainer, or operator to work to complete a specified task. Individual effort assumes productive hours only according to a certain number of productive hour per day.

b) Task effort

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

A.7.2.5 Time interval of events types

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

A.7.3 Count measure type

This measure type identifies number of counted number, turn, event or incident with investigation activity.

Investigation activity includes reviewing, testing and operating, and using from view of human-engineering and ergonomics. This measure type should be attached to informative description on the context including periods, and number and profile of experimented site and users during counting are performed, because the measure strongly depends on those.

If attributes of documents or software product are counted, they are static count type. If events or human actions are counted, they are kinetic count type. The followings are belong to static count type.

A.7.3.1 Number of detected fault type

The measurement is to count the detected faults during reviewing, testing, correcting, operating or maintaining. Severity level may be used to categorize them for taking account of impacts.

A.7.3.2 Program structural complexity number type

The measurement is to count program structural complexity. Examples are number of distinct paths or McCabe's cyclomatic number.

A.7.3.3 Number of detected inconsistency type

This measure is to count detected inconsistent items which are prepared for investigation.

a)Number of failed conforming items

Examples:

- Conformance to specified items of requirements specifications;

- Conformance to rule, regulation, or standard;

- Conformance to protocol, data format, media format, character code, or other.

b)Number of failed 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 use a questionnaires for asking tester, customer, operator, or end user which kinds of deficiencies are discovered.

The followings are example items for specified user callable function:

- Actually available or not;

- Actually operable effectively or not;

- Actually operable to user's specific intended use;

- Actually Expected/needed or not.

A.7.3.4 Number of change type

This type identifies software configuration items which are detected they have been changed. Example is number of changed places in source code.

NOTE : Changed source lines of code may be counted as changed places in source code.

The followings are belong to kinetic count type.

A.7.3.5 Number of detected failure type

The measurement is to count the detected failures during product development, testing, operating or maintaining. Severity level may be used to categorize them for taking account of impacts.

A.7.3.6 Number of attempt (trial) type

This measure is to count intended operation action of user, checking and finding action of personel, questionnaires and replies, or test cases for investigation during reviewing, testing, correcting, operating or maintaining. Examples are number of detection of design anomalies during the review, number of misunderstanding requirements at the specification level, number of actually tried test cases, number of cancellation operation and so on.

A.7.3.7 Stroke of human operating procedure type

This measure is to count strokes of user human action as kinetic steps of procedure during user is interactively operating the software. This measure quantifies an ergonomic usability as well as an effort to use. Therefore, this is used for usability metrics. Examples are number of strokes to perform a task, number of eye movements, and so on.

A.7.3.8 Score type

This type identifies the score or arithmetic calculation result. Score may includes count and calculation of checking on/off on check lists. Examples: Score of check list; score of questionnaire; Delphi method; etc.

A.8 Input to measurement

This helps to understand what kinds of information or documents are generally required to do measurement.

A.9 ISO/IEC 12207 SLCP Reference

This suggest processes of software life-cycle processes (SLCP) defined in ISO/IEC 12207, in which the metric is beneficially applicable.

A.10 Perspective/Beneficer

This helps to understand whose view and benefit are strogly related with the metric, though any other personel and party related to the software are also receive benefit.

Annex B (Informative)
Remarks for better use of metrics

B.1 Take accounts of constraints of applied metrics

The measures may not be interpret adequately and not understood sufficiently without context describing situation and condition during data are gathered and metrics are applied.

For examples, 1)"time required to learn operation" measure is often different between skillful operators in similar software systems and unskillful operators. 2) If testing is not so much intensive, measures acquired as testing result may be different so much from the true value.

Therefore, it is recommended to take accounts the following contexts to avoid to interpret inadequately and to misunderstand the measures of metrics.

a) Differences between testing environment and actual user operation one

Are there any remarkable differences between testing environment and actual user operation one?

The followings are examples:

-testing with higher / comparable / lower performance of CPU of users' computer;

-testing with higher / comparable / lower performance of network and communication;

-testing with higher / comparable / lower performance of operating system;

-testing with higher / comparable / lower performance of user interface type provided by operating system.

b) Differences between testing execution and actual user operational execution

Are there any remarkable difference between testing execution and actual user operational execution?

The followings are examples:

-coverage of functional specification or program by test cases;

-test case sampling ratio;

-high speed real time processing testing;

-over loaded data processing testing;

-non-stop operation testing;

-abnormal, exception, or fault injected data input testing;

-frequency of use, such are daily, weekly, monthly, or only at emergency;

-combination of considerable other software, devices, equipment, and systems.

c) User profile under observation

Are there any remarkable different user profile between tested and actual?

The followings are examples:

-heavy, moderate, less or temporal user;

-trained level, e.g., skillful operator or first entry operator;

-expert user or ordinal user;

-office user or home user.

d) Data validation level

Are there any problems come from what kinds of data collection procedures and level of data validation?

The followings are examples.

-procedures of data collection:

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

-data source report level:

developers' self reports / reviewers' report / inspected report by evaluator;

-data validation activity:

developers' self check / inspection by independent evaluators.

e) Balance of the extent of the investigation performance and measures

Are there remarkable problems in investigation performance?

It is important to take consider to keep balance between the extent of the investigation performance and measures (detected results) in fair range during testing;

Investigation includes reviewing and testing performance.

Most of the external metrics use measurement value derived from testing cases and results of problem detection in a validation testing or operation testing. Most of internal metrics also use derived from review. Therefore, the measured value of external metrics are affected by the extent of the testing effort, that is, investigation performance. Internal metrics also may be similarly affected the extent of the reviewing, that is, investigation performance.

The user of metrics should identify the extent of justification of the measured values depends on the extent of the investigation performance.

The extent of investigation to detect problems as input affects the extent of detected problems (failures, faults, miss-matching, etc.) and the extent of residual covered problems as outputs.

f) Balance of the extent of the specification clearance and conformance testing

Are there remarkable problems in specification clearance?

Metrics often use measurements which count problems by comparing specification with testing results and by determining whether they are consistently matching or not, that is, whether the software conform to its specification or not. However, specification may not be enough mature to do that appropriately. Therefore, it is recommended to conduct to review and to improve the specification from a view of software product fulfilling its specific intended use in actual, during specification based matching testing case are designed and tested.

B.2 Validity demonstration and use of metrics

The user of metrics should identify the methods for metrics validity demonstration, which are following 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, which is given by the square of the linear coefficient.

An evaluator can predict quality characteristics without measure directly by using these metrics which have correlation ability.

b) **Tracking**

If a metric M is directly related to a quality characteristics values 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), which is the same direction (for example, if Q increase, M increase).

An evaluator can detect movement of quality characteristics along time preriod without measure directly by using these 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 process 1, 2,..., n, have the relationship Q1 > Q2 > ..., Qn, then the correspond metric values would have the relationship M1 > M2 > ..., Mn.

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

d) **Predictability**

If a metric is used at time T1 to predict a quality characteristics values 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 with allowed prediction error.

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

e) **Discriminative**

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

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

B.3 Prediction use of metrics

Early detection and prediction of quality of the software product is a one of the most effective use of metrics.

a) Future prediction

- Future measure prediction

To estimate the future values of the same measure by using the current measured values, it is estimated based on trend along with time.

For example, the measured value trend of mean time between failures during testing can be used to estimate the value of mean time in actual operation.

- Future invisible other measure prediction

To estimate the future values of the invisible other measure by using the current measured values, it is estimated based on correlative relations.

For example, the complexity of modules during coding may be used to predict time or effort of program change and test during maintenance.

b) Current fact finding prediction

- Invisible other measure prediction

To estimate the current values of other measure which is supposed to be strongly related mutually by using the current measured values, it is estimated based on correlative relations.

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

The metrics which are designed for predicting the attributes should be documented with explanations including below:

-models for predicting the attribute;

-formula for predicting the attribute;

-experience for predicting the attribute;

-justification for predicting the attribute.

The metrics which are designed for predicting the attributes may be sophisticated with the metrics validation procedure containing below:

-identify the samples of measures of attributes which are to be predicted

-identify the metrics which are supposed to be capable for prediction

-perform a statistical analysis

-document the results

-iterate above periodically

B.4 Detect quality problem prone components

The following methods are usually employed to detect quality problem prone components:

-extremely deviated in distribution;

-exceeding boundaries of adequate pair of upper or lower limit in trend;

-extremely deviated or exceeding boundaries of adequate pair of upper or lower limits correlation.

It is recommended to use chrats such Pareto, trend along the time or histograms.

B.5 Displatying measurement results

a) Displaying quality characteristics evaluation result

For example,following graphical presentations are useful to display quality evaluation result for each quality characteristics and sub-characteristics:

Radar chart; Bar chart and so on.

b) Displaying measures

They are useful grafical presentations such Pareto chart, trend chart along the time, histograms, correlation chart and so on.

Annex C (Informative)

Example of Quality Measurement in the industry using ISO/IEC 9126 concepts

This clause presents a step by step quality approach using ISO/IEC 9126 concepts and quality metrics. This example is a summary of the steps use to put in place, execute a quality process and measure the software quality. This example is based on five-year experience and implementation in the industry. The objective of this clause is to procure to the reader an idea on how he could implement a quality process using ISO9000 and ISO/IEC 9126 concepts in the development process.

C.1 Overview of development and quality process

a) Development process view for software

These figure present an overview of the software life cycle process, quality approach steps and quality metrics (Internal, external and quality in use) at each step of the development process. At this stage, Quality criteria and internal quality and metrics are use to evaluate the software quality.

	Step #1	**Step #2**	**Step #3**	**Step #4**
Development Process (ISO/IEC 12207 Software Life-Cycle Processes)	**Software Requirements Analysis**	**Software Architectural Design**	**Software Detailed Design**	**Software Coding and Testing** **(Coding)**
Quality Approach and Quality Deliverables	**-Goal Quality (GO)** **-Required Product Quality (RPQ)** **-Quality Criteria and Weight** **-Quality Strategy Plan** **-Testing Strategy Plan** **-Configuration Plan**	**-Design Quality (DQ)** **-Development methodology** **-Standards** **-Design standards** **-Program structure**	**-Estimated / Predicted Product Quality** **-Internal Quality Characteristics and Measurement**	**-Estimated / Predicted Product Quality** **-Internal Quality Characteristics and Measurement**

Figure C.1a Quality Approach related to Development Process of Software Life Cycle Process

b) Software qualification Testing process for software

At this stage, external metric is use to evaluated the software quality.

Step #5	Step #6	Step #7	Step #8	Step #9
Software Coding and Testing **(Unit Test)**	**Software Integration,** **Software Qualification Testing (Integrated Testing)**	**System Integration,** **System Qualification Testing (System Testing)**	**Software Acceptance Support (Acceptance Testing)**	**Software Installation Support,** **Operational Testing**
-Testing Quality **-Quality Measure** **-Incident Report**	**-External Quality Characteristics and Measurement** **-Testing log** **-Test Incident Report** **-Test Summary Report** **-Pass/Fail Criteria** **-Signoff**	**-Complete Functionality** **-Testing Condition:** **1) Normal** **2) Exceptional** **3) Stress and Volume** **-Quality -External Quality Characteristics and Measurement** **-Test log** **-Test Incident Report** **-Pass/Fail Criteria** **-Signoff**	**-Complete Functionality** **-Performance Testing** **-Quality -External Quality Characteristics and Measurement** **-Test log** **-Test Incident Report** **-Pass/Fail Criteria** **-Signoff**	**-External Quality, Quality In Use Characteristics and Measurement** **-Test log** **-Test Incident Report**

Figure C.1b Quality approach related to Development process of Software Life cycle Process

C.2 Quality Approach Steps

Step #1 Goal Quality

Software Requirements analysis represent the necessary requirements from the real user. Goal Quality (GQ) represents the necessary and sufficient quality reflecting real user needs. The following relative weights for each quality characteristics have been assigned, based on the quality goal, objectives and the requirement (present at the software requirement analysis and quality requirement). Assigning relative weights will allow the developers and users to focus their efforts on the most important aspects of the software /system. The Quality Strategy Plan will describe the quality requirement, measurement and signoff (pass/fail quality criteria) process for each step of the software development process. The testing strategy plan will focus on the certification process using quality characteristic, sub-characteristic and measuring defect.

Table C.2.1 Example of overall quality characteristics and weights

CHARACTERISTIC	WEIGHT (High/Medium/Low)
Functionality	
Reliability	
Usability	
Efficiency	
Maintainability	
Portability	
Quality in Use	

Table C.2.2 Example of overview of quality sub-characteristics

Characteristic	Sub-Characteristic	User WEIGHT	Technical WEIGHT	Goal Quality Required Product Quality
Functionality	Suitability			
	Accuracy			
	Interoperability			
	Compliance			
	Security			
Reliability	Maturity (hardware/software/ data)			
	Fault tolerance			
	Recoverability (data, process, technology)			
	Redundancy			
Usability	Understandability			
	Learnability			
	Operability			
	Attractiveness			
Efficiency	Time behavior			
	Resource utilisation			
Maintainability	Analyzability			
	Changeability			
	Stability			
	Testability			
Portability	Adaptability			
	Installability			
	Replaceability			
	Co-existence			
Quality In Use	Effectiveness			
	Productivity			
	Safety			
	Satisfaction			

Step #2: Design Conformity

All the next steps are driven by the quality criteria and characteristic, sub characteristic weight.

Step #2 identify the conformity for Data, Process and Technology, using metrics. For each step, a pass or fail criteria are verify using metrics.

		QUALITY ASSURANCE FORM					**No : 1**
Design Phase (1/1)		**Process Name: _____ Date: _____** **Process # : _____** **Architect : _____**					
Charac. **Sub-Char.**		**Indicators to measure**			**Value**	**Freq.**	**Resp.**
Functionality	**Suitability and Accuracy**	FSu-1a	Process specification change ratio	**NB of change to a process due to mis-understanding of the requirements** _____ **NB Total process**		after reviews and at each change request	QA Team
		FSu-1b	Process specification change ratio	**Nb of change to process due to change in environments(legal, organizational, technological)** _____ **Nb Total Process**		after reviews and at each change request	QA Team
		FSu-1c	Process specification change ratio	**Nb of change to Process due to change in the requirements** _____ **Nb Total Process**		after reviews and at each change request	QA Team

Figure C.2.1a An example of Metrics Form Sheet

Step #3: Software Detailed Design Conformity

Step #3 identify the requirements conformity for data, function and technology, using metrics.

		QUALITY ASSURANCE FORM					**No : 1**	
Software Detailed Design Phase (1/1)		**Function Name: _____ Date: _____** **Function # : _____** **Analyst : _____**						
Charac. **Sub-Char.**		**Indicators to measure**				**Value**	**Freq.**	**Resp.**
Functionality	**Suitability and Accuracy**	FSu-1a	Functional specification change ratio	**NB of change to functions due to mis-understanding of the requirements** _____ **NB Total Function**			after reviews and at each change request	QA Team
		FSu-1b	Functional specification change ratio	**Nb of change to functions due to change in environments(legal, organizational, technological)** _____ **Nb Total Function**			after reviews and at each change request	QA Team
		FSu-1c	Functional specification change ratio	**Nb of change to functions due to change in the requirements** _____ **Nb Total Function**			after reviews and at each change request	QA Team

Figure C.2.1b An example of Metrics Form Sheet

Step #4: Software Coding Conformity

Step #4 identify the conformity for coding using internal metrics.

		QUALITY ASSURANCE FORM					**No : 1**	
Software Coding Phase (1/1)		**Program Name: _____ Date: _____** **Program # : _____** **Programmer : _____**						
Charac. **Sub-Char.**		**Indicators to measure**				**Value**	**Freq.**	**Resp.**
F u n c t i o n a l i t y	**Suitability and Accuracy**	FSu-1a	Program specification change ratio	**NB of change to program due to mis-understanding of the requirements** _____ **NB Total program**			after reviews and at each change request	QA Team
		FSu-1b	Program specification change ratio	**Nb of change to program due to change in environments(legal, organizational, technological)** _____ **Nb Total program**			after reviews and at each change request	QA Team
		FSu-1c	Program specification change ratio	**Nb of change to program due to change in the requirements** _____ **Nb Total program**			after reviews and at each change request	QA Team

Figure C.2.1c An example of Metrics Form Sheet

Step #5,6,7: Software Testing

Step # 5,6,7,8 measuring the software quality using external metrics.

		QUALITY ASSURANCE FORM					**No : 2**	
Unit Testing (1/1)		**Program Name : _____ DATE :_____** **Program # : _____** **Programmer :_____**						
Charac.		**Indicators to measure**				**Value**	**Freq.**	**Resp.**
Functionality	**Suitability**	FSu-1d	Functional Cosmetic change ratio	**Nb of change to functions due to Cosmetic change** **Nb Total Function**			after reviews and at each change request	QA Team
	Interoperability	FSu-6	Matched data format ratio	**Nb of transactions with each of the other programs** **Nb Total of transactions**			Once	Progmer

Figure C.2.1d An example of Metrics Form Sheet

	QUALITY ASSURANCE FORM					No : 3
Integrated Testing (1/2)	**Function Name: _____ DATE: _____** **Function # : _____** **Analyst resp. : _____**					
Charac. / **Sub-Charac**	**Indicators to measure**			**Value**	**Freq.**	**Resp.**
Functionality	**Suitability**	FSu-1a	Functional specification change ratio	**Nb of changes to functions due to mis-understanding of the requirements** / **Nb Total Function**		at each change request	QA Team
		FSu-1b	Functional specification change ratio	**Nb of changes to functions du to change in environments(legal, organizational, technological)** / **Nb Total Function**		at each change request	QA Team
		FSu-1c	Functional specification change ratio	**Nb of changes to functions du to change in the requirements** / **Nb Total Function**		at each change request	QA Team
		FSu-1d	Functional Cosmetic change ratio	**Nb of change to functions due to Cosmetic change** / **Nb Total Function**		after reviews and at each change request	QA Team
		FSu-1e	Ratio DataBase Change	**Nb of change request** / **Total of Data Element**		after reviews and at each change request	QA Team
	Accuracy	FAc-1	Anomalies ratio 1	**Nb of anomalies that hinders the function to give results** / **Nb of tests for the function**		Daily	Analyst (Dev. Team)
		FAc-2	Anomalies ratio 2	**Nb of anomalies where function give results but wrong** / **Nb of tests for the function**		Daily	Analyst (Dev. Team)
		FAc-4	Anomalies ratio 3	**Nb of good but unexpected results because of script incorrectness** / **Volume of tests**		Daily	Analyst (Dev. Team)
	Interoperability	FIn-1	Matched data format ratio	**Nb of matched transactions with each function** / **Nb Total of transactions matched**		Once	Analyst (Dev. Team)

Figure C.2.1e An example of Metrics Form Sheet

C.3 Measuring Quality

At each step, quality is measured using metrics for the Software. Multiple measurement could be done according to the step you are measure

Table C.3.1 An example of table representing measuring quality and step you are measure

Characteristic	Sub-Characteristic	Software Architectural Design	Software detailed Design	Software Coding and Testing	……	System Qualifica -tion Testing	Software Acceptance Suport
Functionality	Suitability						
	Accuracy						
	Interoperability						
	Security						
Reliability	Maturity (hardware/software/data)						
	Fault tolerance						
	Recoverability (data, process, technology)						
	Redundancy						
Usability	Understandability						
	Learnability						
	Operability						
	Attractiveness						
Efficiency	Time behavior						
	Resource utilisation						
Maintainability	Analyzability						
	Changeability						
	Stability						
	Testability						
Portability	Adaptability						
	Installability						
	Replaceability						
	Co-existence						