INTERNATIONAL STANDARD

ISO/IEC 20741

First edition
2017-05

Systems and software engineering — Guideline for the evaluation and selection of software engineering tools

Ingénierie des systèmes et du logiciel — Lignes directrices pour l'évaluation et le choix des outils d'ingénierie logicielle

Reference number
ISO/IEC 20741:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

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

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

Introduction

Within systems and software engineering, software engineering tools represent a major part of the supporting technologies used to develop and maintain information technology systems. Their selection is carried out with careful consideration of both the technical and management requirements.

The objective of an evaluation process is to provide quantitative and comparable results of all candidate alternatives. The final selection can then be based on these results. To be widely useful and accepted, the software engineering tool evaluation and selection processes are supposed to help both the users and the suppliers of software engineering tools. The more objective, repeatable, and impartial the evaluation and selection processes are, the more widely acceptable they are. The information and guidance outlined in this document are intended to lead to more cost-effective selections of software engineering tools and to a greater uniformity in how software engineering tool functions and features are described.

For evaluating and selecting software engineering tools, a set of processes providing a procedure for evaluation and selection, a list of capabilities providing scope of functional requirements, and a list of characteristics providing scope of non-functional requirements are needed.

Evaluation and selection of software engineering tools is usually performed within a specific, purpose-oriented tool area for practical reasons, to manage the scope of evaluation and selection. Examples of such tool areas are requirements engineering tools and configuration management tools. Lists of capabilities are tool area specific, but the list of characteristics and the set of evaluation and selection processes are more generic for all software engineering tool areas.

This document defines a set of processes and a list of characteristics which can be used by all software engineering tool areas. This document can be used together with any tool area-specific standard which defines list of capabilities for the tool area.

International standards defining lists of capabilities for specific tool areas have been published, such as ISO/IEC 30130 for “software testing tools”, ISO/IEC TR 24766 for “requirements engineering tools”, and ISO/IEC TR 18018 for “configuration management tools”. Lists of capabilities for other tool areas of software engineering can be developed as a series of standards according to their priority.

It is supposed in this document that tool area is decided before starting the evaluation and selection. It is recommended that the decision would be based on ISO/IEC 15940 which defines the software engineering service for each tool area.

This document adopts the general model of software product quality characteristics and sub-characteristics defined in ISO/IEC 25010 and gives additional guidance how to apply the model when the software product is a software engineering tool. The document follows also the software product evaluation model defined in ISO/IEC 25041.

Systems and software engineering — Guideline for the evaluation and selection of software engineering tools

1 Scope

This document gives guidelines for the evaluation and selection of software engineering tools, covering a partial or full portion of the software engineering life cycle.

It establishes processes and activities to be applied for the evaluation of software engineering tools and selecting the most appropriate software engineering tools from several candidates.

It establishes, for selected processes, the tasks and activities that can be applied for the evaluation of software engineering tools and selecting the most appropriate software engineering tools from several candidates.

It establishes processes that can be applied for the evaluation of software engineering tools and selecting the most appropriate software engineering tools from several candidates.

As these processes are generic, organizations can adapt these generic processes to meet organizational needs. The software engineering tool evaluation and selection processes can be viewed in the larger context of the organization's technology adoption process.

This document provides the following:

a) guidance on identifying organizational requirements for software engineering tools;

b) guidance on mapping those requirements to software engineering tool characteristics to be evaluated;

c) a process for selecting the most appropriate software engineering tool from several tools, based on measurements of the defined characteristics.

NOTE 1 Guidance on mapping those requirements to software engineering tool capabilities to be evaluated is not covered by this document, but is covered by a series of standards for each tool area.

Primary users of this document are organizations that intend to adopt software engineering tools to support their software life cycle processes. Software tool suppliers can also use this document to describe characteristics of their software engineering tools.

This document is not intended to apply to:

a) software engineering frameworks whose purpose is to provide mechanisms for data, control and presentation integration;

b) general purpose tools (e.g. word processors, spreadsheets) which can be used in software engineering activities, nor software engineering tools of very narrow scope or specific purpose (e.g. a compiler);

c) planning for the implementation of software engineering tools within an organization.

NOTE 2 A user of this document can make the best possible selection of a software engineering tool and yet have no guarantee of a successful implementation.

The methods described in this document are useful not only for the selection of software engineering tools, but for any project where COTS/FOSS software can be selected instead of engaging in new software development.

To follow the guidance provided in this document consists in applying the activities and tasks that are attached to the defined processes to evaluate and select software. Organizations using this document for trade purposes can specify the minimum set of processes and their related activities and tasks, suitable to their given application.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

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

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at http://www.iso.org/obp

3.1
atomic sub-characteristic
lowest level sub-characteristics

Note 1 to entry: The highest level evaluation categories are called characteristics. Characteristics are usually subdivided into sub-characteristics. At the lowest level, when no further subdivision is appropriate, the sub-characteristics are referred to as atomic sub-characteristics.

3.2
characteristic
aspect of a product by which it can be described and evaluated

Note 1 to entry: A characteristic can be refined into multiple levels of sub-characteristics that bear on its ability to satisfy stated or implied needs.

3.3
measure (noun)
variable to which a value is assigned as the result of measurement

Note 1 to entry: The term "measures" is used to refer collectively to base measures, derived measures, and indicators.

[SOURCE: ISO/IEC 15939:2007, 2.15, modified — The words "plural form" have been changed to "term".]

3.4
measure (verb)
make a measurement

[SOURCE: ISO/IEC 25040:2011, 4.39]

3.5
measurement
set of operations having the object of determining a value of a measure

Note 1 to entry: Measurement can include assigning a qualitative category such as the language of a source program (ADA, C, COBOL, etc.).

[SOURCE: ISO/IEC 15939:2007, 2.17, modified — Note 1 to entry has been changed.]

3.6
software engineering tool
software product that assists software engineers by providing automated support

3.7
rating
action of mapping the measured value to the appropriate rating level

Note 1 to entry: Used to determine the rating level associated with the software for a specific quality characteristic.

Note 2 to entry: Rating and rating levels can be applied to characteristics other than quality characteristics.

3.8
rating level
scale point on an ordinal scale which is used to categorize a measurement scale

Note 1 to entry: The rating level enables software to be classified (rated) in accordance with the stated or implied needs (see 8.2).

Note 2 to entry: Appropriate rating levels can be associated with the different views of quality, i.e. "Users", "Managers" or "Developers".

4 Abbreviated terms

BMT benchmark test

COTS commercial off-the-shelf

FOSS free/open-source software

GUI graphical user interface

5 Overview of evaluation and selection of software engineering tools

5.1 Introduction of the evaluation and selection of software engineering tools

This document defines both a set of processes and a structured set of software engineering tool characteristics for use in the technical evaluation and the ultimate selection of a software engineering tool. It follows the software product evaluation model defined in ISO/IEC 25041.

This document adopts the general model of software product quality characteristics and sub-characteristics defined in ISO/IEC 25010 and extends these when the software product is a software engineering tool; it provides product characteristics unique to software engineering tools as described in 10.2 to 10.4. This larger set of characteristics is then organized into three groups; they are characteristics related to software engineering tool usage functionality, general quality and not related to quality.

NOTE The capabilities of software engineering tools are defined in a series of standards for each tool area.

The objective of the technical evaluation process is to provide quantitative results on which the final selection can be based. Measurement assigns numbers (or other ratings) to attributes of entities; a major activity of evaluation is to obtain these measurements for use in selection. The final selection results should aim to achieve objectivity, repeatability and impartiality. These objectives and the confidence in the outcomes depend in part on the resources allocated to the overall evaluation and selection process. The user of this document is asked to deal with these issues at an early stage.

5.2 Framework of the evaluation and selection of software engineering tools

This subclause illustrates an overview of the evaluation and selection of software engineering tools discussed in this document as shown in Figure 1. Evaluation and selection of software engineering tools includes four major processes:

— preparation process;

— structuring process;

— evaluation process; and

— selection process.

In the process diagrams Figures 1 to 5, round corner rectangle is process/activity, normal rectangle is outcome of the process/activity and dashed rectangle is a plan which is referred by each process without reference. Also, a solid arrow shows data flow and a dashed arrow shows control flow of transition between processes/activities.

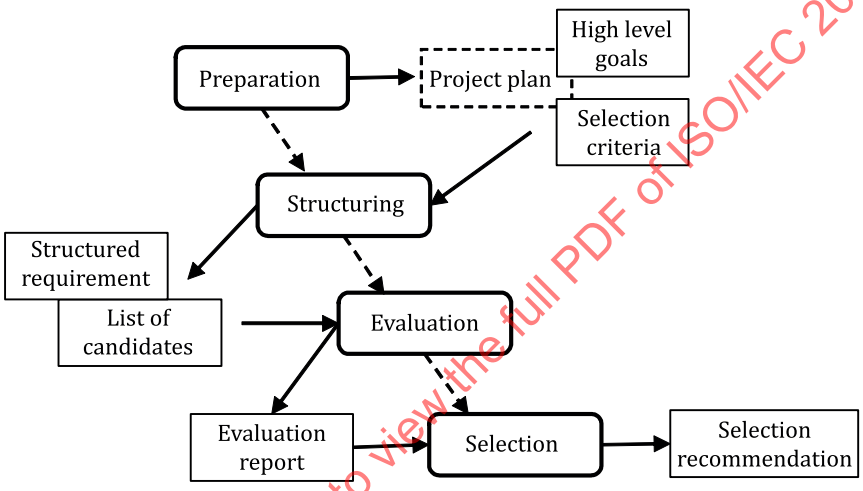

Figure 1 — Framework of evaluation and selection of software engineering tools

A key process is the structuring of a set of requirements against which candidate software engineering tools are to be evaluated and upon which selection decisions are based. The software engineering tool characteristics defined in 10.2 to 10.4 form the basis for requirements structuring and play a central role in the overall process.

5.3 General process considerations

There are several considerations that apply to the processes described in this document on a global basis. The intent is for the user of this document to tailor its application in such a way as to maximize the probability of a successful evaluation and selection process, and minimize its cost and risk.

5.3.1 Sequencing of processes

This document does not impose the sequence of process activities described above and in the following clauses. It is up to the organization to select the relevant processes and activities needed to meet its evaluation and selection goals.

The organization decides which to employ, in what sequence, and with what degree of parallelism. The sequencing of the processes' activities is then documented in an evaluation project plan.

5.3.2 Reducing cost and risk

In general, organizations which apply this document want to minimize the cost of the entire evaluation and selection process to the extent possible, while maintaining the level of effort necessary to select the most appropriate software engineering tool(s) for their use. These objectives can be addressed by minimizing the number of tools evaluated, minimizing the cost of evaluating specific tools, and ensuring that the formality of the process is appropriate to the organization.

The activities of software engineering tool information gathering and identifying final candidates for selection (see Clause 9) effectively allow the user of this document to screen the available tools against the organization's needs, and eliminate from consideration tools which do not, or are not likely to, substantially address the organization's needs.

The organization can be unable to find any tool which appears likely to sufficiently meet its needs. In such a case, the stated needs themselves should be re-examined, and if they are found to accurately reflect the organization's actual requirements for technology improvement, the overall evaluation and selection process can be abandoned. Similarly, if the final candidate tools appear to be marginal in addressing the organization's needs, the level of detail and formality of the subsequent activities should be made to reflect the risk factor, and the organization should be prepared to not select a tool if the evaluation process so indicates, as the typical cost of bringing a new tool into operational use is substantial.

Evaluations of candidate tools might have already been performed and be available to the organization. Such information can be used to reduce the cost of candidate tool evaluation.

NOTE 1 Previous evaluations which have been performed on a different version of the candidate tool can still yield useful information. Similarly, evaluations which addressed a different set of organizational needs can still provide useful information.

This document calls for the development of several plans and reports, and implicitly, for their review by various personnel within the organization. In addition, activities are required to perform the four processes outlined.

The format and level of detail of the data products are left to the discretion of the organization, as is the level of effort necessary to perform the activities.

NOTE 2 Some organizations can be required to limit the scope, detail and formality of the processes to apply this document within existing resource constraints.

6 Preparation process

6.1 Purpose

In this process, the objective and criteria for tool selection are clarified, and the project plan is defined for tool evaluation and selection.

A set of software engineering tool selection guidelines is identified and a project plan developed. The process is shown in Figure 2.

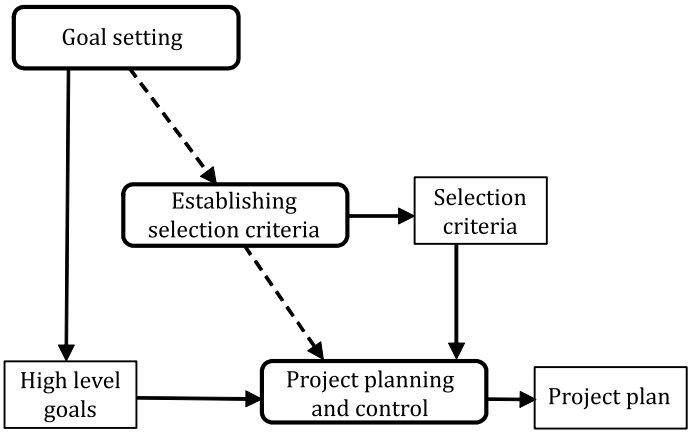

Figure 2 — Overview of preparation process

6.2 Outcomes

Outcomes resulting from the successful implementation of the preparation process:

a) objective for tool selection;

b) list of selection criteria; and

c) project plan for evaluation and selection tools.

6.3 Activities and tasks

6.3.1 Goal setting

The development of a set of realistic goals is a necessary first activity. In developing goals, both a rationale for acquisition (why acquire a software engineering tool) and a general policy for acquisition (what type of tool to acquire and how to do it) should be developed.

NOTE Goal setting activities, including possibly the identification of selection criteria, might have already been performed as a part of other efforts prior to formally entering the preparation process of evaluation and selection of software engineering tools.

The following tasks should be performed:

a) develop rationale for acquisition:

   1) review the organization's current software development process, determining its maturity and areas of concern;

   2) review the current state of software engineering tool technology and observe trends for consideration as future reference technology;

   3) compare the organization's current practices to possible future practices if software engineering tools are adopted and identify areas of potential benefit;

b) define goals and expectations:

   1) set overall goals (e.g. productivity improvement, quality improvement, enhanced process manageability);

   2) define evaluation and selection constraints (e.g. cost, schedule, resources);

3) quantify and classify expectations (based upon goals);

c) set general policy for acquisition:

1) identify constraints on tool acquisition (e.g. implementation cost, schedule, and other resources);

2) develop alternate approaches to introducing/augmenting software engineering tool (e.g. buy a tool, modify an existing tool or develop a new tool);

3) assess the feasibility of the various alternatives in light of organizational readiness, technical considerations, performance specifications, and resources;

4) identify probable impacts of software engineering tools on the organization, e.g. areas where training and education, procedure guides, and technical support are needed to effectively deploy software engineering tool; and

5) the goals and expectations established here are used to guide subsequent activities in the overall process and, finally, to validate the selection decision.

6.3.2 Establishing selection criteria

Based upon the goals and expectations developed above, selection criteria should be established.

a) Decompose the high level goals into a set of selection criteria to make the (go/no go) selection decision.

The selection criteria should be objective and quantitative. Each selection criterion should include some defined threshold specified on which the major go/no go decision is made during selection.

b) Define the relative importance of the selection criteria.

NOTE 1 The relative importance of the selection criteria is used to determine the weights assigned to tool characteristics, sub-characteristics and/or capabilities for evaluation.

c) Define the level of detail and the nature of the evaluation activities to be performed.

NOTE 2 The nature of the evaluation activities covers the methods used in collecting the data, e.g. reference, for example, how the data are measured, collected with predefined criteria, or based upon subjective observation.

d) Define the evaluation/selection scenario to be performed.

NOTE 3 The evaluation/selection scenario is defined as a method such as evaluating in a small team of a real project, evaluating in an experimental pilot project, or evaluating with catalogue of tools.

6.3.3 Project planning and control

Based upon the goals and selection criteria which have been established for the overall evaluation and selection process, a project plan should be created and a control mechanism implemented. The plan and control mechanism should be developed in accordance with the organization's normal planning and control process, and it should contain the following.

a) A project team organization with assigned responsibilities.

The skill of the evaluators has an impact on the results of the evaluation and its applicability to the organization. The evaluation personnel should be selected with this in mind, and the skill level of evaluators should be a factor in assessing evaluation results. The evaluation team should be representative of the intended tool user group.

b) A set of operational goals obtained by decomposing the overall goals previously established.

NOTE For instance, overall goal as "No need to install software tools in each client" is decomposed operational goals as "Need to run on specified web browser" and "Server side of the tool and its data should be installed on the server set in own intranet".

c) A set of selection guidelines: weighted selection criteria, definition of level of detail and special features, and an evaluation and selection scenario.

d) A schedule of activities and their tasks, along with an estimate of resource requirements and a cost estimate.

The project plan and the evaluation strategy and cost should be proportionate to the life-cycle cost and expected benefit of the tool being evaluated.

e) A means of monitoring and controlling the execution of the plan.

f) If developed, the project plan and control mechanism should be updated as the project evolves.

7 Structuring process

7.1 Purpose

In the structuring process, requirements for tools are gathered from tool user organization and the details of requirements are structured, as the framework for tool selection and evaluation was created in the preparation process.

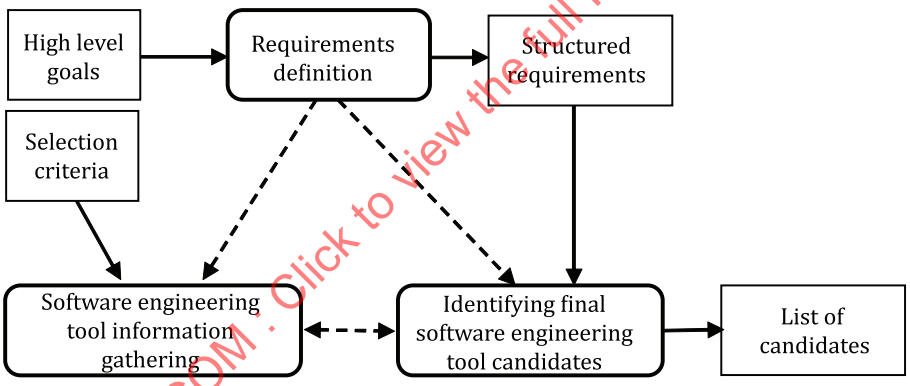

Figure 3 — Overview of structuring process

7.2 Outcomes

Outcomes resulting from the successful implementation of the structuring process:

a) organizational information;

b) tool requirements; and

c) structured tool requirements.

7.3 Activities and tasks

7.3.1 Requirements definition

7.3.1.1 Overview

During requirements definition, the requirements for the software engineering tool are collected and organized into the software engineering tool characteristics. 10.2 identifies characteristics related to software engineering tool usage functionality, 10.3 identifies general software quality characteristics, 10.4 identifies a set of characteristics not related to quality, and a series of standards for each tool area identifies the major software engineering tool specific capabilities.

7.3.1.2 Organizational information gathering

To be able to define a set of detailed requirements to be satisfied by the software engineering tool, information about the organization should be gathered, including:

a) commitment of the organization to fully fund and implement software engineering tool use;

b) current software engineering environment within the organization, including data describing current hardware, operating software, tool use, and cloud environment (e.g. IaaS, PaaS and SaaS);

c) types of software development projects undertaken by the organization including size, domain of application;

d) characteristics and constraints of the target systems for which software is developed;

e) specific expected impacts and improvements of software engineering tool on the organization;

f) requirements from potential tool users and end users;

g) current organizational procurement policies; and

h) current information security policies for software adoption.

This information is necessary to ensure the tool or tools are appropriate for use within the organization, and that they address organizational needs and needs perceived by their future users.

NOTE This information can be gathered in a number of ways, including surveys and focus groups.

7.3.1.3 Requirements identification

The tool user's requirements should deal with the question of what the software engineering tool should do as well as its impact on the existing environment. The following tasks should be performed in building the list of requirements:

a) analyze the requirements and adjust the level of detail to which requirements are defined and measured;

b) evaluate the current need for software engineering tools while taking into consideration those projects where the software engineering tool might initially be used;

c) identify desired methodology;

d) identify portions of the life cycle to be supported (e.g. planning, analysis, design);

e) identify required capabilities of the software engineering tool;

f) identify required characteristics related to software engineering tool usage functionality of the software engineering tool;

g) identify required general software quality characteristics of the software engineering tool;

h) identify required characteristics not related to quality of the software engineering tool; and

i) check the consistency of the requirements with the previously established goals.

NOTE 1 These requirements represent the total set of organizational requirements. It is possible that no single software engineering tool can satisfy all of the requirements, but that individual software engineering tools can satisfy a sufficient number to justify their use by the organization, which can continue to search for tools to support remaining requirements.

NOTE 2 When identifying required capabilities, extracting capabilities from a series of standards for each tool area is suggested. If there are no proper standards, the evaluation and selection project is needed to define the capabilities by itself.

7.3.1.4 Requirements structuring

The applicability of the user needs identified in 10.2 to 10.4 and a series of standards for each tool area, and any others which the organization may wish to add, should be defined. The purpose of this structuring is to organize the requirements in such a way that the evaluation can be proceeded more effectively. The tasks include:

a) categorize the user requirements in terms of the organization of 10.2 to 10.4 and a series of standards for each tool area, and decompose them into detailed specifications;

b) select characteristics and capabilities from 10.2 to 10.4 and a series of standards for each tool area which might be evaluated to determine the extent to which the software engineering tool meets the detailed specifications;

c) identify weights for the characteristics and/or capabilities.

NOTE The weights are applied to the ratings determined during the evaluation as part of the selection process and reflect the relative importance of the related selection criterion as determined during the preparation process.

The assignment of weights is a subjective task which has a fundamental impact on the outcome of the entire evaluation and selection process. The assignment of weights should reflect both the organization's actual requirements and the ability of the organization to evaluate the characteristic. See Annex B for further discussion.

ISO/IEC 25051 addresses quality requirements applicable to software engineering tools when considered as software packages and should be consulted as part of the requirements structuring task. It provides additional guidance on a subset of the quality requirements of ISO/IEC 25010.

7.3.2 Software engineering tool information gathering

A general search of potential software engineering tools to be evaluated is undertaken based upon the requirements and selection criteria established. The activities of gathering information and identifying the candidate software engineering tools may require several iterations to quickly and efficiently identify the most promising tools for further evaluation. For the software engineering tools that appear most promising for further evaluation, additional and more detailed data that deal with their potential acquisition are obtained. This additional information can help to quickly eliminate many tools, allowing attention to be focused on the remaining candidates. Information to be obtained includes:

a) supplier general information (e.g. business history, available support, plans and strategies);

NOTE The word "Supplier" is used for provider of commercial tools and open source tools or cloud environment services (e.g. PaaS and SaaS).

b) supplier's specific product development strategy;

c) the tool's cost (e.g. price, maintenance, modifications, training);

d) the hardware and software required to support tool use;

e) the hardware and software required to support final application/product use;

f) the training required for efficient tool use;

g) the tool's functional capabilities;

h) the tool's methodology and life-cycle support;

i) how the tool interfaces to external systems;

j) the number of users, existence of a user's group, the users' response to the tool; and

k) the tool's licence mechanism (e.g. floating licence, multi-user licences, cross platform licences).

7.3.3 Identifying final candidate software engineering tools

When the set of potential candidate tools has been identified, the final candidates for selection (those to be evaluated) may be chosen. This is accomplished through the following tasks:

a) establish a set of high-priority or critical requirements to be met by software engineering tools;

b) compare the user's functional requirements with the software engineering tool's functional capabilities, supporting methodology, system environment;

c) compare the managerial requirements with the software engineering tool's cost, available training and support;

d) analyze the tool suppliers' user base, user response, support, and business history; and

e) identify tools satisfying a sufficient number of high-priority or critical requirements which then become the final candidates for formal evaluation. The results of the previous tasks provide the justification for the list of candidates.

NOTE The tasks described in this subclause represent a "screening" of possible candidates to allow the organization to identify the candidates most likely to be acceptable, given the organization's requirements or supplier's abilities. The identification of final candidates can be performed in parallel with software engineering tool information gathering, or the two activities can be iterated. The goal is to reduce the cost of tool evaluation by only considering a screened set of final candidates during the evaluation process.

8 Evaluation process

8.1 Purpose

Candidate tools are collected in structuring process. They are evaluated with evaluation plan, and the evaluation result is reported. The selection process is carried out with this report.

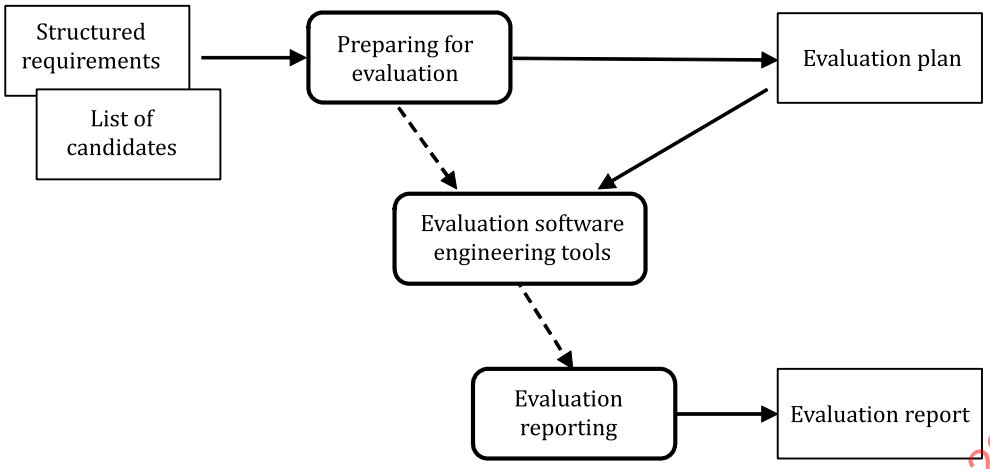

Figure 4 — Overview of evaluation process

8.2 Outcomes

Outcomes resulting from the successful implementation of the evaluation process:

a) measures and rates for evaluation;

b) evaluation report.

8.3 Activities and tasks

8.3.1 Preparing for evaluation

To define the necessary level of detail prior to beginning evaluation activities, final preparations are necessary. Based upon the list of candidate software engineering tools and the structured requirements, the following tasks should be accomplished:

a) for each atomic sub-characteristic and/or capability, define or select one or more measures and define the details of their use;

b) set the rating levels and identify the means by which the levels are generated or computed;

NOTE 1 A measured value (e.g. average lines of code per module = 274) then is assigned a rating value (e.g. based on four grades: A, B, C and D). The means by which rating levels are obtained from measurements are identified.

c) define the assessment characteristics and/or capabilities for evaluation, establishing what is acceptable, taking into consideration the rating levels previously defined and the context of use of the product; and

d) identify and schedule all activities that are performed as part of the evaluation process.

NOTE 2 Activities include preparing any data sets necessary for the evaluation, obtaining tool documentation and an instance of the tool to be evaluated, providing evaluators any necessary training in tool use, hands-on tool use, recording of tool outputs, and analysis of results.

In some cases, a benchmark test (BMT) might be a part of the evaluation process. The recommended approach for a BMT includes:

— identifying the required critical tool functions;

— identifying a test project or sample program to be the basis for the BMT;

— developing a BMT scenario, defining inputs and expected outputs.

To focus evaluation activities and provide for traceability of the evaluation process, develop an evaluation plan that includes the information above.

8.3.2 Evaluating software engineering tools

8.3.2.1 Overview

The software is evaluated in comparison with each of the chosen characteristics and/or capabilities. Evaluation is a process of measurement, rating and assessment.

8.3.2.2 Measurement

Measurements can be made based upon information obtained by examining the software engineering tool itself, or information about it, through the following types of tasks:

a) examining the supplier's documentation;

b) examining the source code and other intermediate products, if available;

c) interviewing actual users of the software;

d) viewing demonstrations and interviewing demonstrators;

e) executing test cases;

f) applying to test projects;

g) examining results of previous evaluations (whether in-house, third party, or other evaluations); and

h) performing a BMT on the candidate tools and analyzing the results.

Measurement values may be binary, based on a continuous scale (quantifiable), or textual. There are both objective and subjective characteristics.

NOTE Objective characteristics are those which permit independent and repeatable test or measure. Subjective characteristics are those for which no independent and repeatable test or measure exists (e.g. fitness of the user interface to the culture of the user).

For objective characteristics, the evaluation should be made by a repeatable procedure such that another evaluator would be able to produce the same results. During evaluation, if test cases are used, a uniform, predefined, and documented set of cases should be used.

For subjective characteristics, the evaluation should be performed independently by different evaluation personnel, who discusses and agrees upon results.

The evaluation results should be recorded in a quantified manner, where possible, together with textual justification, where applicable.

8.3.2.3 Rating

In the rating task, each measured value is rated against the scale of values defined in the evaluation plan. Rating levels are either directly generated or computed according to previously defined algorithms.

NOTE When requirements are revised during the evaluation, rating scales can be revised.

8.3.2.4 Assessment

Based upon the resulting ratings and the previously defined assessment criteria, assess the sub-characteristics, characteristics and/or capabilities. In accordance with the selection guidelines and the evaluation plan, ratings should be aggregated up to the top level.

8.3.3 Evaluation reporting

The end result of the evaluation activities is an evaluation report. One single evaluation report may address all tools which were evaluated; alternatively, several evaluation reports may be written, each reporting on a subset of the tools evaluated.

Evaluation results should be provided in terms of the lowest level of sub-characteristic and/or capability decomposition (normally an atomic sub-characteristic and/or capability). For each sub-characteristic and/or capability, the value measured should be given in terms of the rating level for that measure.

Based upon the lowest level results, any aggregation should be shown so as to make clear the method of aggregation: any weights used, the elements aggregated, and the level to which aggregation is performed. The result is a profile describing the results of the evaluation in terms of scores for the characteristics and/or capabilities of Clause 7, or in terms of scores for sub-characteristics and/or capabilities, depending on the level of aggregation.

In cases where the report covers multiple tools, or where the results of this evaluation are compared to those of other evaluations, care should be taken to ensure the results are provided in a uniform format which facilitates comparison (e.g. by using templates). Objective results should be provided with minimal accompanying text. Subjective results should be supported by text describing the specific reasons for the values assigned.

9 Software engineering tool selection process

9.1 Purpose

In the selection process, selection method is determined and selection is carried out with the values of evaluation report. Validity of selected tool is checked out in the goal and the guideline.

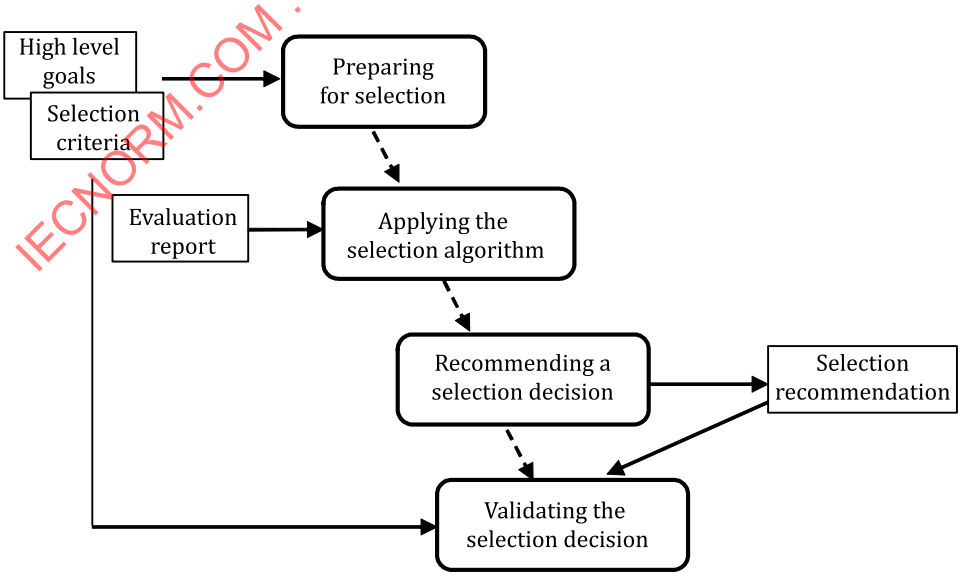

Figure 5 — Overview of selection process

9.2 Outcomes

Outcomes resulting from the successful implementation of the selection process:

a) selection algorithm; and

b) selection decision.

9.3 Activities and tasks

9.3.1 Preparing for selection

The selection algorithm determines how the data generated during the various evaluations are combined and compared to result in ratings for each candidate.

Based upon the original goals and selection guidelines, a final set of selection criteria is identified and the basis upon which these criteria are to be assessed is defined. This definition is based upon the aggregated evaluation assessments described in 8.3.3.

When selection criteria and/or rating is inappropriate, selection criteria and/or rating should be changed.

The algorithm for further aggregating the results, comparing the candidates, and arriving at a decision is then defined. A discussion of selection algorithms is provided in Annex A.

9.3.2 Applying the selection algorithm

The evaluation results are used as inputs to the selection algorithm. Information relating the candidate tools is output. Each tool's evaluation results provide a technical summary of each tool's characteristics, aggregated up to the level specified in the selection algorithm (usually the characteristic level). The selection algorithm combines the results of the evaluations of the various tools, providing a comparison for use by decision makers.

9.3.3 Recommending a selection decision

When the selection algorithm has been applied, a decision may be made to acquire a tool or set of tools. This is a management decision based upon the technical comparison provided above and additional management criteria.

Such a decision would indicate that the most appropriate of the candidates has been identified for selection. Alternatively, the assessment of evaluation results may show a need for additional information. In such case, some previous activities might be iterated.

The selection decision should be justified with a rationale that summarizes the information and logic that led to the selection.

9.3.4 Validating the selection decision

The final activity in the process should be the validation of the recommended selection. The original goals and selection guidelines should be reviewed and compared to the evaluation results and other data relating to the recommended selection. A check should be made to ensure that if the recommendation is accepted, the high level goals (or a sufficient number of them) are met. It might be found that no adequate tool exists, in which case a choice may be made between the development of a new tool or the modification of an existing tool (within the user organization or outside), or abandoning the entire evaluation and selection process.

10 General software tool characteristics

10.1 Overview

In the structuring process, evaluation and rating are defined by using characteristics and functionality which are mentioned in this clause.

The user needs which drive any evaluation and/or selection process are based upon the characteristics and sub-characteristics described below and the capabilities described in any tool area specific standard which defines list of capabilities for the tool area. By defining user needs in the terms used here and in the tool area specific standard, assessments and comparisons are made based upon a broad, common, and nearly complete set of characteristics and/or capabilities. As discussed above, a structuring activity is required to transform the set of needs initially identified by the user into the terms provided here and in the tool area specific standard.

The top-level evaluation categories are called characteristics and/or capabilities. Each characteristic is subdivided into sub-characteristics. Sub-characteristics may be further subdivided into lower level sub-characteristics. At the lowest-level, sub-characteristics are referred to as atomic sub-characteristics. 10.2 to 10.4 define atomic sub-characteristics in terms of their attributes; each of these is assigned a value during the evaluation process based upon one or more measures (see 8.2). Each capability which is defined in subsequent part of this document series is also assigned a value during the evaluation process based upon one or more measures (see 8.2).

It is unlikely that any user of this document needs to use all of the atomic sub-characteristics given below; users should select only those sub-characteristics that have significant weight with respect to their organization's requirements. There are cases where additional needs or characteristics, specific to a particular evaluation or selection, have to be added to those listed below; in that sense, the atomic sub-characteristics listed below might be considered as a partial list, to be augmented as necessary.

Non-atomic sub-characteristics are assigned values by aggregating the values of their component sub-characteristics, weighted as called for in the evaluation plan. This aggregation task is continued until the levels of aggregation called for in the evaluation plan have been reached. The selection algorithm is then used to combine the evaluation results of the various tools for comparison and decision.

10.2 Characteristics related to software engineering tool usage functionality

10.2.1 Overview

The following characteristics relate the tool to its environment and the projects it is used to support.

10.2.2 Software engineering tool operation environment characteristics

A set of atomic sub-characteristics which bear on the relationship between the software engineering tool and its operational (host) environment (see Table 1).

Table 1 — Atomic sub-characteristics of the software engineering tool operation environment

No.	Atomic sub-characteristics	Description
1)	Required hardware characteristics of tool	attributes relating to any hardware requirements for its use[a] NOTE Typical hardware items to be listed include processors (including co-processors), main memory type, bus type, type and size of peripheral storage, extension or graphics cards, input and output equipment.
2)	Required software environment of tool	attributes relating to any software items required for its use NOTE Typical software items to be listed include operating systems, database management systems languages, character sets and character codes, and communications/network packages.
3)	Physical environment of tool	attributes relating to any geographical aspects of the development environment which impacts tool use NOTE Considerations include physical and temporal separation of users and the related issues of networking considerations, online/offline considerations, and repository updating/mirroring at multiple sites.

[a] The user of this document should identify hardware items for which the minimal requirements cannot provide adequate performance, e.g. main memory. Hardware necessary to provide acceptable performance should be identified.

The user of this document should identify hardware items which are supported by the tool as options, e.g. input and output devices.

10.2.3 Software engineering tool integrability characteristics

A set of atomic sub-characteristics which bear on the ability of the software engineering tool to integrate and interoperate with other items in its operational environment (see Table 2). The evaluation and selection of software engineering tools is performed in the context of the software engineering environment in which the tool is used.

NOTE Examples of other environmental items include those given in the hardware and software environment of the tool above.

Table 2 — Atomic sub-characteristics of the software engineering tool integrability

No.	Atomic sub-characteristics	Description
1)	Compatibility with environment elements	attributes relating to its ability to interoperate with and/or directly exchange data with hardware/software environments[a] NOTE 1 Examples of other software tools include word processors and other documentation tools, databases, repositories, and other software engineering tools. NOTE 2 The extent to which the tool conforms to standards for "openness" including data interchange formats can be evaluated in terms of a number of existing standards, including, for example, ISO/IEC 15476-1.
2)	Data integration	attributes relating to its ability to use, process, and deliver information shared by other tools or part of a repository[b]
3)	Control integration	attributes relating to its ability to interact with other tools NOTE 1 Invocation of the tool can be controlled by rules, e.g. security policies, pre- or post-invocation of other tools, allowable concurrency and synchronization. The rules might be defined by the supported method. The level of compatibility of rules might be addressed. NOTE 2 Different communication mechanisms exist in different frameworks. Communication between tools can be handled by sharing data within a repository, by message queues between tools, by client-server approaches, or by remote procedure calls.
4)	Presentation integration	attributes relating to the level of the homogeneity, compatibility, and consistency of its user interface with other tools
5)	Metadata access	attributes relating to the access provided to the tool's metadata

[a] If the tool contains an interfacing capability (e.g. an application programming interface) which allows the tool to be used independently of environment elements, that interface should be described.

Process management, project management and system development functions can be provided by separate, specialized tools. In such a case, the connectivity between the different tools should be considered under this sub-characteristic.

[b] The software engineering tool should be evaluated against:

metadata: if the framework is based on a specific data model (e.g. entity/relationship or object oriented), the ability of the software engineering tool to generate, manipulate or access compatible structures of the relevant data model (e.g. compatible type, constraint, attribute or relationship);

product data engineering: if the repository holds a formal data description for software engineering, the ability of the software engineering tool to generate, manipulate or access relevant data according to its functional scope.

10.2.4 Software engineering tool application characteristics

A set of atomic sub-characteristics which bear on the relationship between the software engineering tool and the projects to which it is applied, including the environment of its products and characteristics of those products (see Table 3).

Table 3 — Atomic sub-characteristics of the software engineering tool application

No.	Atomic sub-characteristics	Description
1)	Hardware and software environment of tool products	attributes relating to the set of hardware and software items on which or with which the products of the tool can be used NOTE The level of platform support in the target environment might be a consideration, e.g. does the software engineering tool generate screens and does it generate calls to an external (platform or environment) service to generate the screens.
2)	Conformance to standards of tool products	attributes relating to conformance of the products resulting from its use to standards NOTE Examples include language, database, repository, communication, graphical user interface (GUI), documentation, development, configuration management, security, portability, and information interchange standards.
3)	Domain of application	attributes relating to the application domains which the software engineering tool is designed to support NOTE Examples of application domains include transaction processing, real-time, information management, safety critical, and mobile.
4)	Size of application supported	attributes which would result in size limitations of the application and therefore limit the tool's applicability NOTE Such parameters might include lines of code, levels of nesting, size of database, number of data elements, and number of configuration items.
5)	Languages supported	attributes relating to its ability to support specific languages NOTE Examples of such languages include programming languages, data and query languages, graphical languages, operating system interfaces such as job control languages, scripts, and XML.
6)	Databases supported	attributes relating to its ability to support specific databases
7)	Software repository (information base)	attributes relating to its ability to house and manage all relevant software engineering process information NOTE 1 This includes its ability to make information developed in one life-cycle activity available for use during other activities, as well as its ability to provide access to this information to other environment elements. NOTE 2 Examples of such information include requirements and design documentation, code, and test data. NOTE 3 Includes the ability to handle relevant data by variant.
8)	Methodology support	attributes relating to the set of methods or methodologies which it can support NOTE 1 Examples of methods or methodologies include various object-oriented approaches, structured (top-down) approaches, data-driven approaches, and real-time extensions. NOTE 2 A software engineering tool's support for a method or methodology can be evaluated based upon the extent that the tool provides the specific capabilities necessary to implement the methodology.
9)	Internationalization	attributes relating to its ability to be used in different cultures and to generate products in terms of different countries or cultures NOTE 1 Examples include using natural different languages, character sets, character and graphic presentation modes (left-right, top-bottom), and different date formats. NOTE 2 This sub-characteristic might have an influence on other hardware or software environment elements.

10.3 General quality characteristics

10.3.1 Overview

The following characteristics describe the quality of the tool in the terms of ISO/IEC 25010.

NOTE Further guidance on evaluating the sub-characteristics in this clause is found in ISO/IEC 25051.

10.3.2 Functional suitability characteristics

Functional suitability is a degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions. Atomic sub-characteristics of functional suitability are listed in Table 4.

Table 4 — Atomic sub-characteristics of the functional suitability

No.	Atomic sub-characteristics	Description
1)	Functional completeness	degree to which the set of functions covers all the specified tasks and user objectives
2)	Functional correctness	degree to which a product or system provides the correct results with the needed degree of precision
3)	Functional appropriateness	degree to which the functions facilitate the accomplishment of specified tasks and objectives EXAMPLE A user is only presented with the necessary steps to complete a task, excluding any unnecessary steps. NOTE Functional appropriateness corresponds to suitability for the task in ISO 9241-110.

10.3.3 Performance efficiency characteristics

Performance efficiency is a performance relative to the amount of resources used under stated conditions. Atomic sub-characteristics of performance efficiency are listed in Table 5.

NOTE Resources can include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, storage media).

Table 5 — Atomic sub-characteristics of the performance efficiency

No.	Atomic sub-characteristics	Description
1)	Time behaviour	degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements
2)	Resource utilization	degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements NOTE Human resources are included as part of efficiency.
3)	Capacity	degree to which the maximum limits of a product or system parameter meet requirements NOTE Parameters might include the number of items that may be stored, the number of concurrent users, the communication bandwidth, throughput of transactions, and size of database.

10.3.4 Compatibility characteristics

Compatibility is a degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment. Atomic sub-characteristics of compatibility are listed in Table 6.

NOTE Adapted from ISO/IEC/IEEE 24765.

Table 6 — Atomic sub-characteristics of the compatibility

No.	Atomic sub-characteristics	Description
1)	Co-existence	degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product
2)	Interoperability	degree to which two or more systems, products or components can exchange information and use the information that has been exchanged NOTE Based on ISO/IEC/IEEE 24765.

10.3.5 Usability characteristics

Usability is a degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. Atomic sub-characteristics of usability are listed in Table 7.

NOTE 1 Adapted from ISO 9241-210.

NOTE 2 Usability is either specified or measured as a product quality characteristic in terms of its sub-characteristics, or specified or measured directly by measures that are a subset of quality in use.

Table 7 — Atomic sub-characteristics of the usability

No.	Atomic sub-characteristics	Description
1)	Appropriateness recognizability	degree to which users can recognize whether a product or system is appropriate for their needs cf. functional appropriateness NOTE 1 Appropriateness recognizability depends on the ability to recognize the appropriateness of the product or system's functions from initial impressions of the product or system and/or any associated documentation. NOTE 2 The information provided by the product or system can include demonstrations, tutorials, documentation or, for a web site, the information on the home page.
2)	Learnability	degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use NOTE Can be specified or measured either as the extent to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by product properties corresponding to suitability for learning as defined in ISO 9241-110.
3)	Operability	degree to which a product or system has attributes that make it easy to operate and control NOTE Operability corresponds to controllability, (operator) error tolerance and conformity with user expectations as defined in ISO 9241-110.

Table 7 *(continued)*

No.	Atomic sub-characteristics	Description
4)	User error protection	degree to which a system protects users against making errors
5)	User interface aesthetics	degree to which a user interface enables pleasing and satisfying interaction for the user NOTE This refers to properties of the product or system that increase the pleasure and satisfaction of the user, such as the use of colour and the nature of the graphical design.
6)	Accessibility	degree to which a product or system can be used by people with the widest range of characteristics and/or capabilities to achieve a specified goal in a specified context of use NOTE 1 The range of capabilities includes disabilities associated with age. NOTE 2 Accessibility for people with disabilities can be specified or measured either as the extent to which a product or system can be used by users with specified disabilities to achieve specified goals with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by the presence of product properties that support accessibility.

10.3.6 Reliability characteristics

Reliability is a degree to which a system, product or component performs specified functions under specified conditions for a specified period of time. Atomic sub-characteristics of reliability are listed in Table 8.

NOTE 1 Adapted from ISO/IEC/IEEE 24765.

NOTE 2 Wear does not occur in software. Limitations in reliability are due to faults in requirements, design and implementation, or due to contextual changes.

NOTE 3 Dependability characteristics include availability and its inherent or external influencing factors, such as availability, reliability (including fault tolerance and recoverability), security (including confidentiality and integrity), maintainability, durability, and maintenance support.

Table 8 — Atomic sub-characteristics of the reliability

No.	Atomic sub-characteristics	Description
1)	Maturity	degree to which a system, product or component meets needs for reliability under normal operation NOTE The concept of maturity can also be applied to other quality characteristics to indicate the degree to which they meet required needs under normal operation.
2)	Availability	degree to which a system, product or component is operational and accessible when required for use NOTE 1 Adapted from ISO/IEC/IEEE 24765. NOTE 2 Externally, availability can be assessed by the proportion of total time during which the system, product or component is in an up state. Availability is therefore a combination of maturity (which governs the frequency of failure), fault tolerance and recoverability (which governs the length of down time following each failure).
3)	Fault tolerance	degree to which a system, product or component operates as intended despite the presence of hardware or software faults NOTE Adapted from ISO/IEC/IEEE 24765.
4)	Recoverability	degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system NOTE Following a failure, a computer system is sometimes down for a period of time, the length of which is determined by its recoverability.

10.3.7 Security characteristics

Security is a degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. Atomic sub-characteristics of security are listed in Table 9.

NOTE 1 As well as data stored in or by a product or system, security also applies to data in transmission.

NOTE 2 Survivability (the degree to which a product or system continues to fulfil its mission by providing essential services in a timely manner in spite of the presence of attacks) is covered by recoverability.

NOTE 3 Immunity (the degree to which a product or system is resistant to attack) is covered by integrity.

NOTE 4 Security contributes to trust.

Table 9 — Atomic sub-characteristics of the security

No.	Atomic sub-characteristics	Description
1)	Confidentiality	degree to which a product or system ensures that data are accessible only to those authorized to have access
2)	Integrity	degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data NOTE Adapted from ISO/IEC/IEEE 24765.
3)	Non-repudiation	degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later NOTE Adapted from ISO 7498-2.
4)	Accountability	degree to which the actions of an entity can be traced uniquely to the entity NOTE Adapted from ISO 7498-2.
5)	Authenticity	degree to which the identity of a subject or resource can be proved to be the one claimed NOTE Adapted from ISO/IEC 13335-1.

10.3.8 Maintainability characteristics

Maintainability is a degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. Atomic sub-characteristics of maintainability are listed in Table 10.

NOTE 1 Modifications can include corrections, improvements or adaptation of the software to changes in environment and in requirements and functional specifications. Modifications include those carried out by specialized support staff and those carried out by business or operational staff, or end users.

NOTE 2 Maintainability includes installation of updates and upgrades.

NOTE 3 Maintainability can be interpreted as either an inherent capability of the product or system to facilitate maintenance activities or the quality in use experienced by the maintainers for the goal of maintaining the product or system.

Table 10 — Atomic sub-characteristics of the maintainability

No.	Atomic sub-characteristics	Description
1)	Modularity	degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components NOTE Adapted from ISO/IEC/IEEE 24765.
2)	Reusability	degree to which an asset can be used in more than one system, or in building other assets NOTE Adapted from IEEE 1517-2004.
3)	Analysability	degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified NOTE Implementation can include providing mechanisms for the product or system to analyse its own faults and provide reports prior to a failure or other event.
4)	Modifiability	degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality NOTE 1 Implementation includes coding, designing, documenting and verifying changes. NOTE 2 Modularity and analysability can influence modifiability. NOTE 3 Modifiability is a combination of changeability and stability.
5)	Testability	degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met NOTE Adapted from ISO/IEC/IEEE 24765.

10.3.9 Portability characteristics

Portability is a degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. Atomic sub-characteristics of portability are listed in Table 11.

NOTE 1 Adapted from ISO/IEC/IEEE 24765.

NOTE 2 Portability can be interpreted as either an inherent capability of the product or system to facilitate porting activities or the quality in use experienced for the goal of porting the product or system.

Table 11 — Atomic sub-characteristics of the portability

No.	Atomic sub-characteristics	Description
1)	Adaptability	degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments NOTE 1 Adaptability includes the scalability of internal capacity (e.g. screen fields, tables, transaction volumes, report formats, etc.). NOTE 2 Adaptations include those carried out by specialized support staff and those carried out by business or operational staff, or end users. NOTE 3 If the system is to be adapted by the end user, adaptability corresponds to suitability for individualization as defined in ISO 9241-110.
2)	Installability	degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment NOTE If the product or system is to be installed by an end user, installability can affect the resulting functional appropriateness and operability.
3)	Replaceability	degree to which a product can replace another specified software product for the same purpose in the same environment NOTE 1 Replaceability of a new version of a software product is important to the user when upgrading. NOTE 2 Replaceability can include attributes of both installability and adaptability. The concept has been introduced as a sub-characteristic of its own because of its importance. NOTE 3 Replaceability reduces lock-in risk so that other software products can be used in place of the present one, for example, by the use of standardized file formats.

10.4 General characteristics not related to quality

10.4.1 Overview

The following characteristics are general in nature and address both the tool itself, the tool developer and/or supplier.

10.4.2 Acquisition process characteristics

A set of atomic sub-characteristics that bear on the acquisition process necessary if the software engineering tool is selected for adoption (see Table 12).

Table 12 — Atomic sub-characteristics of the acquisition process

No.	Atomic sub-characteristics	Description
1)	Cost of tool implementation	attributes relating to the cost of implementing the tool[a]
2)	Licensing policies	attributes relating to the supplier's licensing policies NOTE These include available licence options, the right to copy (media and documentation), and any restrictions and/or fees for secondary usage. That is, the tool user sells products which include some element or aspect of a tool used to develop the product. These also include any terms and conditions, including product guarantees, which apply to the tool.
3)	Export restrictions	attributes relating to the identification of any restrictions on the export of the tool, or of any secondary usage of the tool

[a] All aspects relevant to the specific instance should be considered, not only tool purchase price, but also the costs for installation, initial maintenance, hardware/software revision or upgrades, and training. Price data on all relevant configurations should be considered, including single copy, multiple copies, site licence, corporate licence, and network licence.

10.4.3 Implementation characteristics

A set of atomic sub-characteristics that bear on the tool's delivery, installation and operation (see Table 13).

Table 13 — Atomic sub-characteristics of the implementation

No.	Atomic sub-characteristics	Description
1)	Cost effectiveness	attributes relating to the cost of tool operation NOTE A cost/benefit analysis can be performed or some consideration given to the expected level of productivity of the software engineering tool.
2)	Development delivery constraints	attributes relating to any schedule constraints involving further product development and/or delivery[a]
3)	Workarounds required for user organization	attributes relating to any workarounds which would be required to implement the software engineering tool in the user's environment NOTE An example of such a workaround is finding a way to use a centralized tool (single common database) in a distributed environment.
4)	Infrastructure needs	attributes relating to infrastructure requirements for tool use NOTE Examples include floor space, table space, furniture, electricity and other physical requirements generated by new tool-related hardware or tool use considerations.

[a] In addition, the time required for the tool's users to become productive with the tool (learning curve) should also be considered.

10.4.4 Support indicators characteristics

A set of atomic sub-characteristics that bear on the supplier's ability to provide tool support (see Table 14).