International Standard

ISO/IEC/IEEE 32430

Software engineering — Software non-functional size measurement

Ingénierie du logiciel — Norme pour la quantification des caractéristiques non fonctionnelles des logiciels

Second edition
2025-02

Reference number
ISO/IEC/IEEE 32430:2025(en)

Institute of Electrical and Electronics Engineers, Inc
3 Park Avenue, New York
NY 10016-5997, USA

Email: stds.ipr@ieee.org
Website: www.ieee.org

Published in Switzerland

Contents

Page

Foreword

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

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 document 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 or www.iec.ch/members_experts/refdocs).

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

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

For an explanation of 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 www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC/JTC 1, *Information technology,* Subcommittee SC 7, *Software and systems engineering,* in cooperation with the Systems and Software Engineering Standards Committee of the IEEE Computer Society, under the Partner Standards Development Organization cooperation agreement between ISO and IEEE.

This second edition cancels and replaces the first edition (ISO/IEC/IEEE 32430:2021), which has been technically revised.

The main changes are as follows:

— clarifications of terminology regarding software size and software non-functional requirements.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

Used in conjunction with functional size measurement (FSM), non-functional size measurement (NFSM) assists organizations in multiple ways. It provides insight into the delivery of software projects and maintenance of software and assists in estimating the effort and in the analysis of key performance indicators, such as quality and productivity.

Having both software functional size and non-functional size provides significant information for the management of software product development. The functional size is quantifiable and represents a standardized measure of the functional project/application size. Providing a quantifiable measure derived from the non-functional requirements (NFR) for the software allows organizations to build historical data repositories that can be referenced to assist in decision making for the technical or quality aspects of applications.

By learning the method as described in this document and by performing the non-functional sizing together with functional sizing, users avoid duplication of measurement effort.

Having this information enables software professionals to do the following:

a) plan and estimate projects;

b) compare projects and compare the project to benchmarks;

c) identify areas of improvement and analyze trends of improvement;

d) quantify the impacts of the current non-functional strategies;

e) assist in determining future non-functional strategies;

f) provide specific data when communicating non-functional issues to various audiences;

g) communicate the impact of non-functional requirements (NFR) on the project with users and customers;

h) help users determine the benefit of an application package to their organization by assessing portions or categories that specifically match their requirements;

i) determine the non-functional size of a purchased application package.

NFSM is independent of the way NFR are defined. Analyzing the requirements to measure the non-functional size can assist in identifying implicit requirements.

This document contains rules on how to use ISO/IEC 20926:2009 (IFPUG FSM) and NFSM together, so that there are no gaps and no overlaps between the functional size and the non-functional size. A software requirement that contains both functional and non-functional aspects can be sized using ISO/IEC 20926:2009 for its functional aspects and this document for its non-functional aspects.

FSM and NFSM together can provide a broader view of the size of the software product.

Software engineering — Software non-functional size measurement

1 Scope

1.1 Overview

This document defines a method for measuring the non-functional size of the software. It complements ISO/IEC 20926:2009, which defines a method for measuring the functional size of the software.

This document also describes the complementarity of functional and non-functional sizes, so that deriving the sizes from the functional and the non-functional requirements does not result in duplication in the distinct functional and non-functional sizes.

In general, there are many types of non-functional requirements. Moreover, non-functional requirements and their classification evolve over time as the technology advances. This document does not intend to define the type of NFR for a given context. Users can choose ISO 25010 or any other standard for the definition of NFR. It is assumed that users size the NFR based on the definitions they use.

This document covers a subset of non-functional requirements. It is expected that, with time, the state of the art can improve and that potential future versions of this document can define an extended coverage. The ultimate goal is a version that, together with ISO/IEC 20926:2009, covers every aspect that can be required of any prospective piece of software, including aspects such as process and project directives that are hard or impossible to trace to the software's algorithm or data. The combination of functional and non-functional sizes would then correspond to the total size necessary to bring the software into existence.

Estimating the cost, effort and duration of the implementation of the NFR is outside the scope of this document.

1.2 Purpose

The purpose of this document is to define a method for measuring the non-functional size of the software.

1.3 Word usage

The word *"shall"* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted *("shall"* equals is required to).[1),2)]

The word *"should"* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*"should"* equals is recommended that).

The word *"may"* is used to indicate a course of action permissible within the limits of the standard (*"may"* equals is permitted to).

The word *"can"* is used for statements of possibility and capability, whether material, physical, or causal ("can" equals is able to).

1) The use of the word *"must"* is deprecated and shall not be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.

2) The use of *"will"* is deprecated and shall not be used when stating mandatory requirements, "will" is only used in statements of fact.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14143-1:2007, *Information technology — Software measurement — Functional size measurement — Part 1: Definition of concepts*

ISO/IEC 20926:2009, *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

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

ISO, IEC and IEEE maintain terminology databases for use in standardization at the following addresses:

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

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

— IEEE Standards Dictionary Online: available at: http://dictionary.ieee.org

NOTE For additional terms and definitions in the field of systems and software engineering, see ISO/IEC/IEEE 24765, which is published periodically as a "snapshot" of the SEVOCAB (Systems and Software Engineering Vocabulary) database and is publicly accessible at www.computer.org/sevocab.

3.1.1
algorithm
finite ordered set of well-defined rules for the *solution* (3.1.41) of a problem

[SOURCE: ISO/IEC 2382:2015, 2121376, modified — Notes to entry have been removed.]

3.1.2
application
cohesive collection of automated procedures and data supporting a business objective, consisting of one or more components, modules, or subsystems

EXAMPLE Accounts payable, accounts receivable, payroll, procurement, shop production, assembly line control, air search radar, target tracking, weapons firing, flight line scheduling, and passenger reservations.

[SOURCE: ISO/IEC 20926:2009, 3.2]

3.1.3
boundary
conceptual interface between the software under study and its *users* (3.1.42)

Note 1 to entry: In this document, "application boundary" is used.

[SOURCE: ISO/IEC 14143-1:2007, 3.3, modified — Note 1 to entry has been added.]

3.1.4
code data
list data
translation data
substitutable data, which provide a list of valid values that a descriptive attribute may have

EXAMPLE Examples of code data include:

— State

  — State code

  — State name

— Payment type

  — Payment type code

  — Payment description

Note 1 to entry: Typically the attributes of the code data are code, description and/or other 'standard' attributes describing the code.

Note 2 to entry: When codes are used in the business data, it is necessary to have a means of translating to convert the code into something more recognizable to the *user* (3.1.42). In order to satisfy non-functional user requirements (e.g., usability, readability, maintainability) developers often create one or more tables containing the code data.

3.1.5
data configuration
process of adding, changing or deleting reference data or *code data* (3.1.4) information from the database or data storage with no change in software code or the database structure

3.1.6
data element type
DET
unique, *user* (3.1.42)-recognizable, non-repeated attribute

Note 1 to entry: A DET can be in business data, reference data, or *code data* (3.1.4). The number of different types of data elements of all tables is the *number of DETs* (3.1.27). Instances of control information are also counted as a DET, such as the "Enter" key to facilitate an *external input (EI)* (3.1.12).

[SOURCE: ISO/IEC 20926:2009, 3.15, modified — Note 1 to entry has been added.]

3.1.7
data function
functionality provided to the *user* (3.1.42) to meet internal or external data storage requirements

Note 1 to entry: A data function is either an *internal logical file* (3.1.21) or an *external interface file* (3.1.14).

[SOURCE: ISO/IEC 20926:2009, 3.16]

3.1.8
database view
result set of a stored query on the data, which the database *users* (3.1.42) can query just as they would in a persistent database collection object

Note 1 to entry: This stored (pre-established query) command is kept in the database dictionary. Unlike ordinary base tables in a relational database, it is a virtual table computed or collated dynamically from data in the database, when access to that view is requested. Changes applied to the data in a relevant underlying table are reflected in the data shown in subsequent invocations of the view.

3.1.9
elementary process
EP
smallest unit of activity that is meaningful to the *user* (3.1.42)

[SOURCE: ISO/IEC 20926:2009, 3.21, modified — The abbreviated term "EP" has been added.]

3.1.10
extensive mathematical operation
mathematical operation that includes one or more series of mathematical equations and calculations executed in conjunction with, or according to, logical operators, to produce results identifiable to the *user* (3.1.42)

EXAMPLE A program evaluation review technique (PERT) to calculate the expected completion date of a project.

Note 1 to entry: See also: *algorithm* (3.1.1).

3.1.11
extensive logical operation
logical operation either containing a minimum of four nesting levels, containing more than 38 *DETs* (3.1.6) required to operate the logical operation, or both

Note 1 to entry: These DETs do not necessarily have to cross the *application* (3.1.2) *boundary* (3.1.3).

3.1.12
external input
EI
elementary process (EP) (3.1.9) that processes data or control information sent from outside the *boundary* (3.1.3)

[SOURCE: ISO/IEC 20926:2009, 3.27, modified — Note 1 to entry has been removed.]

3.1.13
external inquiry
EQ
elementary process (EP) (3.1.9) that sends data or control information outside the *boundary* (3.1.3)

[SOURCE: ISO/IEC 20926:2009, 3.28, modified — Note 1 to entry has been removed.]

3.1.14
external interface file
EIF
user (3.1.42)-recognizable group of logically related data or control information, which is referenced by the *application* (3.1.2) being measured, but which is maintained within the *boundary* (3.1.3) of another application

[SOURCE: ISO/IEC 20926:2009, 3.29, modified — Note 1 to entry has been removed.]

3.1.15
external output
EO
elementary process (EP) (3.1.9) that sends data or control information outside the *boundary* (3.1.3) and includes additional processing logic beyond that of an *external inquiry* (3.1.13)

[SOURCE: ISO/IEC 20926:2009, 3.30, modified — Note 1 to entry has been removed.]

3.1.16
family of platforms
software *platforms* (3.1.29) that are serving the same purpose

EXAMPLE Operating systems; browsers.

3.1.17
file type referenced
FTR
data function (3.1.7) read or maintained by a transactional function

Note 1 to entry: A file type referenced includes an *internal logical file* (3.1.21) read or maintained by a transactional function, or an *external interface file* (3.1.14) read by a transactional function.

[SOURCE: ISO/IEC 20926:2009, 3.31, modified — Note 1 to entry has been added.]

3.1.18
function point
FP
unit of measure for functional size

[SOURCE: ISO/IEC 20926:2009, 3.35]

3.1.19
functional size measurement
FSM
process of measuring the functional size

[SOURCE: ISO/IEC 14143-1:2007, 3.7]

3.1.20
functional user requirement
FUR
sub-set of the *user* (3.1.42) requirement that describes what the software does, in terms of tasks and services

Note 1 to entry: FUR include but are not limited to:

— data transfer (for example: input customer data; send control signal);

— data transformation (for example: calculate bank interest; derive average temperature);

— data storage (for example: store customer order; record ambient temperature over time);

— data retrieval (for example: list current employees; retrieve latest aircraft position).

User requirements that are not FUR include, but are not limited to:

— quality constraints (for example: usability, reliability, efficiency and portability);

— organizational constraints (for example: locations for operation, target hardware and compliance to standards);

— environmental constraints (for example: interoperability, security, privacy, and safety);

— implementation constraints (for example: development language, delivery schedule).

[SOURCE: ISO/IEC 14143-1:2007, 3.8]

3.1.21
internal logical file
ILF
user (3.1.42)-recognizable group of logically related data or control information maintained within the *boundary* (3.1.3) of the *application* (3.1.2) being measured

[SOURCE: ISO/IEC 20926:2009, 3.39, modified — Note 1 to entry has been removed.]

3.1.22
logical file
logical group of data as seen by the *users* (3.1.42)

[SOURCE: ISO/IEC 24570:2018]

3.1.23
multiple instance approach
case where each method of delivery of the same functionality is counted separately

Note 1 to entry: See also: *single instance approach* (3.1.33).

3.1.24
non-functional size
size of the software derived by quantifying the *non-functional requirements* (3.1.25) for the software, defined by a set of rules

3.1.25
non-functional requirement
NFR
requirement for a *software-intensive system* (3.1.40) or for a *software product* (3.1.38), including how it should be developed and maintained, and how it should perform in operation, except any *functional user requirement* (3.1.20) for the software

3.1.26
NFR for software
non-functional requirements for software
portion of the *non-functional requirements (NFR)* (3.1.25) that are realized by the software

3.1.27
number of DETs
sum of all *data element types (DETs)* (3.1.6) that are part of the input/output/query of the elementary process *(EP)* (3.1.9), plus the data elements which are read or maintained internally to the *boundary* (3.1.3)

3.1.28
partition
set of software functions within an *application* (3.1.2) *boundary* (3.1.3) that share common criteria and values

EXAMPLE Partitions can be used to improve maintainability by dividing the application into two modules (within a boundary), each needing different expertise. Integrating the modules requires additional effort, which is not reflected when sizing the functional aspect of the project or product, using *function point* (3.1.18) analysis.

Note 1 to entry: Setting partitions is not based on technical or physical considerations.

Note 2 to entry: Partitions do not overlap.

3.1.29
platform
type of computer or hardware device or associated operating system, or a virtual environment, on which software can be installed or run

Note 1 to entry: Combination of an operating system and hardware that makes up the operating environment in which a program runs. A platform is distinct from the unique instances of that platform, which are typically referred to as devices or instances.

3.1.30
precautionary principle
principle that the introduction of a new product or process whose ultimate effects are unknown or disputed should be resisted until such risks are appropriately mitigated

Note 1 to entry: The precautionary principle denotes a duty to prevent harm, when it is within our power to do so, even when all the evidence is not in. This principle has been codified in several international treaties and in international law.

3.1.31
primary intent
intent that is first in importance

3.1.32
record element type
RET
user (3.1.42) recognizable sub-group of *data element types (DETs)* (3.1.6) within a *data function* (3.1.7)

[SOURCE: ISO/IEC 20926:2009, 3.46]

3.1.33
single instance approach
case where all methods of delivery of the same functionality are counted as one

Note 1 to entry: See also *multiple instance approach* (3.1.23).

3.1.34
SNAP category
software non-functional assessment process category
group of components, processes or activities that are used in order to measure the *non-functional size* (3.1.24) of the software

Note 1 to entry: Categories classify the method and are generic enough to allow for future technologies. Categories are divided into sub-categories, so that the SNAP category groups the sub-categories based on similar operations or similar types of sizing activities.

3.1.35
SNAP counting unit
SCU
software non-functional assessment process counting unit
component, process or activity, in which *non-functional size* (3.1.24) is measured

Note 1 to entry: The SCU is identified according to the nature of each sub-category. It can contain both functional and non-functional characteristics; therefore, sizing an SCU is performed for its functional sizing, using *function point* (3.1.18) analysis (FPA), and for its non-functional sizing, using SNAP.

3.1.36
SNAP point
SP
software non-functional assessment process point
unit of measurement to express the *non-functional size* (3.1.24) of the software

3.1.37
SNAP sub-category
software non-functional assessment process sub-category
component, process or activity performed within the project, to meet a *non-functional requirement* (3.1.25)

Note 1 to entry: A non-functional requirement may be sized using more than one sub-category.

3.1.38
software product
set of computer programs, procedures, and possibly associated documentation and data

Note 1 to entry: A software product is a software system viewed as the output (product) resulting from a process.

[SOURCE: ISO/IEC/IEEE 12207:2017, 3.1.54]

3.1.39
software project
set of work activities, both technical and managerial, required to satisfy the terms and conditions of a project agreement

Note 1 to entry: A software project has specific starting and ending dates, well-defined objectives and constraints, established responsibilities, a budget and a schedule. A software project may be self-contained or may be part of a larger project. In some cases, a software project may span only a portion of the software development cycle. In other cases, a software project may span many years and consist of numerous subprojects, each being a well-defined and self-contained software project.

3.1.40
software-intensive system
system for which software is a major technical challenge and is perhaps the major factor that affects system schedule, cost and risk

Note 1 to entry: In the most general case, a software-intensive system is comprised of hardware, software, people and manual procedures.

[SOURCE: IEEE 1062-2015, 3.1]

3.1.41
solution
combination of people, processes and technologies to implement a desired capability

[SOURCE: IEEE 7005-2021, 3.1]

3.1.42
user
any person or thing that communicates or interacts with the software at any time

EXAMPLE Software applications, animals, sensors, or other hardware.

[SOURCE: ISO/IEC 14143-1:2007, 3.11]

3.2 Abbreviated terms

API	application programming interface
EDI	electronic data interchange
FAQ	frequently asked questions
FPA	function point analysis
GUI	graphical user interface
HR	human resources
IATA	International Air Transport Association
IFPUG	International Function Point Users Group
NFR	non-functional requirement(s)
NFSM	non-functional size measurement
PDF	Portable Document Format
SOA	service-oriented architecture
SNAP	software non-functional assessment process
UI	user interface
WCAG	Web Content Accessibility Guidelines
XML	eXtensible Markup Language

4 Introductory information

4.1 User requirements for a system

Figure 1 illustrates how requirements can be classified.