FINAL
DRAFT

INTERNATIONAL
STANDARD

**ISO/IEC/
IEEE/FDIS
29119-2**

ISO/IEC JTC **1**/SC **7**

Secretariat: BIS

Voting begins on:
2021-06-10

Voting terminates on:
2021-08-05

Software and systems engineering — Software testing —

Part 2: Test processes

Ingénierie du logiciel et des systèmes — Essais du logiciel —

Partie 2: Processus des essais

Reference number
ISO/IEC/IEEE/FDIS 29119-2:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

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 ISO/IEC documents should be noted. This document was drafted in accordance with the rules given in the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

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) or the IEC list of patent declarations received (see https://patents.iec.ch).

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.

ISO/IEC/IEEE 29119-2 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 29119-2:2013), which has been technically revised.

The main changes compared to the previous edition are as follows:

— The definition of the test design and implementation process (8.2) has been updated. In the first edition, this process was based on the use of test conditions. Feedback on use of the standard highlighted a problem with users' understanding of 'test conditions' and their use for deriving test cases. This second edition has replaced the use of 'test conditions' with 'test models'. Annex E provides more detail on this change.

A list of all parts in the ISO/IEC/IEEE 29119 series can be found on the ISO and IEC websites.

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

The purpose of this document is to define a generic process model for software testing that can be used by any organization when performing any form of software testing. It comprises test process descriptions that define the software testing processes at the organizational level, test management level and dynamic test levels. Supporting informative diagrams describing the processes are also provided. ISO/IEC/IEEE 29119 (all parts) supports dynamic testing, functional and non-functional testing, manual and automated testing, and scripted and unscripted testing. The processes defined in this document can be used in conjunction with any software development lifecycle model. Each process is defined using the generic process template that is provided in ISO/IEC TR 24774, and covers the purpose, outcomes, activities, tasks and information items of each test process.

Testing is a key approach to risk treatment in software development. This document follows a risk-based approach to testing. Risk-based testing is a best-practice approach to strategizing and managing testing, as it allows testing to be prioritized and focused on the most important features and quality attributes.

This document uses the traditional concept of organizations and projects, but some organizations, especially those using an agile approach, do not organize their development in terms of projects; instead, they run product development based on more long-lasting product teams. Users of this document can substitute the term 'product' for 'project' where appropriate.

The concepts and definitions that support ISO/IEC/IEEE 29119 (all parts) are defined in ISO/IEC/IEEE 29119-1. Templates and examples of test documentation that are produced during the testing process are defined in ISO/IEC/IEEE 29119-3. Software test design techniques that can be used during testing are defined in ISO/IEC/IEEE 29119-4.

ISO/IEC/IEEE 29119 (all parts) aims to provide those responsible for software testing with the information required to manage and perform software testing in any organization.

Users of ISO/IEC/IEEE 12207 perform several activities and tasks which are related to software testing. Annex B provides a mapping for such users between the clauses and subclauses of ISO/IEC/IEEE 12207 and the clauses and subclauses of this document.

Users of ISO/IEC 17025 perform several activities and tasks which are related to software testing. Annex C provides a mapping for such users between the clauses and subclauses of ISO/IEC 17025 and the clauses and subclauses of this document.

Users of BS 7925-2 perform several activities and tasks which are related to software component testing. Annex D provides a mapping for such users between the clauses and subclauses of BS 7925-2 and the clauses and subclauses of this document.

Software and systems engineering — Software testing —

Part 2: Test processes

1 Scope

This document specifies test processes that can be used to govern, manage and implement software testing for any organization, project or testing activity. It comprises generic test process descriptions that define the software testing processes. Supporting informative diagrams describing the processes are also provided.

This document is applicable to testing in all software development lifecycle models.

This document is intended for, but not limited to, testers, test managers, developers and project managers, particularly those responsible for governing, managing and implementing software testing.

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/IEEE 12207, *Systems and software engineering — Software life cycle processes*

3 Terms and definitions

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

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

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

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

— IEEE Standards Dictionary Online: available at https://ieeexplore.ieee.org/xpls/dictionary.jsp

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. ISO/IEC/IEEE 29119-1 includes all terms that are used in ISO/IEC/IEEE 29119 (all parts).

3.1
actual results
set of behaviours or conditions of a *test item* (3.42), or set of conditions of associated data or the *test environment* (3.34), observed as a result of *test execution* (3.38)

EXAMPLE Outputs to screen, outputs to hardware, changes to data, reports and communication messages sent.

3.2
completion criteria
conditions under which the *testing* (3.61) activities are considered complete

3.3
dynamic testing
testing (3.61) in which a *test item* (3.42) is evaluated by executing it

3.4
expected results
observable predicted behaviour of the *test item* (3.42) under specified conditions based on its specification or another source

3.5
exploratory testing
type of unscripted experience-based *testing* (3.61) in which the tester spontaneously designs and executes tests based on the tester's existing relevant knowledge, prior exploration of the *test item* (3.42) (including the results of previous tests), and heuristic "rules of thumb" regarding common software behaviours and types of failure

3.6
incident
anomalous or unexpected event, set of events, condition, or situation at any time during the life cycle of a project, product, service, or system

3.7
incident report
documentation of the occurrence, nature, and status of an *incident* (3.6)

Note 1 to entry: Incident reports are also known as anomaly reports, bug reports, defect reports, error reports, issues, problem reports and trouble reports, amongst other terms.

3.8
organizational test practices
documentation that expresses the recommended approaches or methods for the *testing* (3.61) to be performed within an organization, providing detail on how the testing is to be performed

Note 1 to entry: The organizational test practices is aligned with the *organizational test policy* (3.52).

Note 2 to entry: An organization can have more than one organizational test practices document to cover markedly different contexts, such one for mobile apps and one for safety critical systems.

Note 3 to entry: The organizational test practices can incorporate the context of the test policy where no separate test policy is available.

3.9
organizational test process
test process (3.55) for developing and managing *organizational test specifications* (3.10)

3.10
organizational test specification
document that provides information about *testing* (3.61) for an organization, i.e. information that is not project-specific

EXAMPLE The most common examples of organizational test specifications are the *organizational test policy* (3.52) and the *organizational test practices* (3.8).

3.11
performance testing
type of *testing* (3.61) conducted to evaluate the degree to which a *test item* (3.42) accomplishes its designated functions within given constraints of time and other resources

3.12
product risk
risk that a product can be defective in some specific aspect of its function, quality, or structure

3.13
project risk
risk related to the management of a project

EXAMPLE Lack of staffing, strict deadlines, changing requirements.

3.14
regression testing
testing (3.61) to performed following modifications a *test item* (3.42) or to its operational environment, to identify whether failures in unmodified parts of the test item occur

Note 1 to entry: Regression testing differs from *retesting* (3.15) in that it does not test that the modification works correctly, but that other parts of the system have not been accidentally affected by the change.

Note 2 to entry: The adequacy of a set of regression *test cases* (3.23) depends on the item under *test* (3.21) and on the modifications to that item or its operational environment.

3.15
retesting
confirmation testing
testing (3.61) performed to check that modifications made to correct a fault have successfully removed the fault

Note 1 to entry: When retesting is performed it is often complemented by *regression testing* (3.14), to ensure that other unmodified parts of the *test item* (3.42) have not been accidentally adversely affected by the modifications.

3.16
risk-based testing
testing (3.61) in which the management, selection, prioritisation, and use of testing activities and resources are consciously based on corresponding types and levels of analysed risk

3.17
risk treatment
process to eliminate risk or reduce it to a tolerable level

Note 1 to entry: Risk treatment measures can include avoiding, optimizing, mitigating, transferring, or retaining risk.

[SOURCE: ISO/IEC/IEEE 15026-1:2019, 3.3.11, modified — Note 1 to entry has been added.]

3.18
scripted testing
testing (3.61) performed based on a documented *test script* (3.54)

Note 1 to entry: This term normally applies to manually executed testing, rather than the execution of an automated script.

3.19
security testing
test type (3.60) conducted to evaluate the degree to which a *test item* (3.42), and associated data and information, are protected so that unauthorized persons or systems cannot use, read, or modify them, and authorized persons or systems are not denied access to them

3.20
static testing
testing (3.61) in which a *test item* (3.42) is examined against a set of quality or other criteria without the test item being executed

EXAMPLE Reviews, static analysis.

3.21
test
activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component

3.22
test basis
information used as the basis for designing and implementing *test cases* (3.23)

Note 1 to entry: The test basis can take the form of documentation, such as a requirements specification, design specification, or module specification, but can also be an undocumented understanding of the required behaviour.

3.23
test case
set of preconditions, inputs and *expected results* (3.4), developed to drive the execution of a *test item* (3.42) to meet *test objectives* (3.49)

Note 1 to entry: A test case is the lowest level of test implementation documentation (i.e. test cases are not made up of test cases) for the *test level* (3.43) or *test type* (3.60) for which it is intended.

Note 2 to entry: Test case preconditions include the required state of the *test environment* (3.34), data (e.g. databases) used by the *test item* (3.42), and the test item itself.

Note 3 to entry: Inputs are the data information and actions, where applicable, used to drive *test execution* (3.38).

3.24
test case specification
documentation of a set of one or more *test cases* (3.23)

3.25
test completion process
test management process (3.45) for ensuring that useful test assets are made available for later use, *test environments* (3.34) are left in a satisfactory condition, and the results of *testing* (3.61) are recorded and communicated to relevant stakeholders

3.26
test completion report
test summary report
report that provides a summary of the *testing* (3.61) that was performed

3.27
test condition
testable aspect of a component or system, such as a function, transaction, feature, quality attribute, or structural element identified as a basis for *testing* (3.61)

Note 1 to entry: ISO/IEC/IEEE 29119 (all parts) does not use the concept of test conditions, but instead uses the concept of a *test model* (3.46) for test design. See Annex E for an explanation.

3.28
test coverage
degree, expressed as a percentage, to which specified *test coverage items* (3.29) have been exercised by a *test case* (3.23) or test cases

3.29
test coverage item
coverage item
measurable attribute of a *test item* (3.42) that is the focus of *testing* (3.61)

EXAMPLE Equivalence partitions, transitions between states, executable statements.

3.30
test data
data created or selected to satisfy the input requirements for executing one or more *test cases* (3.23)

Note 1 to entry: Test data can be stored within the *test item* (3.42) (e.g. in arrays or flat files), or can come from external sources, such as other systems, hardware devices, or human operators.

3.31
test data readiness report
document describing the status of each *test data* (3.30) requirement

3.32
test design and implementation process
test process (3.55) for deriving and specifying *test cases* (3.23) and *test procedures* (3.53)

3.33
test design technique
test technique
procedure used to create or select a *test model* (3.46), identify *test coverage items* (3.29) and derive corresponding *test cases* (3.23)

EXAMPLE Equivalence partitioning, boundary value analysis, decision table *testing* (3.61), branch testing.

Note 1 to entry: The test design technique is typically used to achieve a required level of *test coverage* (3.28).

Note 2 to entry: Some test practices, such as *exploratory testing* (3.5) or model-based testing, are sometimes referred to as "test techniques". Following the definition in ISO/IEC/IEEE 29119 (all parts)ds, they are not test design techniques as they are not themselves providing a way to create test cases, but instead use test design techniques to achieve that.

3.34
test environment
environment containing facilities, hardware, software, firmware, procedures, needed to conduct a *test* (3.21)

Note 1 to entry: A test environment can contain multiple environments to accommodate specific *test levels* (3.43) or types (e.g. a unit test environment, a performance test environment).

Note 2 to entry: A test environment can comprise several interconnected systems or virtual environments.

3.35
test environment readiness report
document that describes the status of the *test environment* (3.34)

Note 1 to entry: This can list the status of each of the *test environment requirements* (3.36).

3.36
test environment requirements
description of the necessary properties of the *test environment* (3.34)

Note 1 to entry: All or parts of the test environment requirements can reference where the information can be found, e.g. in the *organizational test practices* (3.8) document, *test plan* (3.50), and *test specification* (3.57).

3.37
test environment and data management process
test process (3.55) for establishing and maintaining a required *test environment* (3.34) and corresponding *test data* (3.30)

3.38
test execution
process of running a *test* (3.21) on the *test item* (3.42), producing *actual results* (3.1)

3.39
test execution log
record of the execution of one or more *test procedures* (3.53)

3.40
test execution process
dynamic *test process* (3.55) for executing *test procedures* (3.53) created in the *test design and implementation process* (3.32) in the prepared *test environment* (3.34) and recording the results

3.41
test incident reporting process
dynamic *test process* (3.55) for reporting *incidents* (3.6) requiring further action that were identified during the *test execution process* (3.40) to the relevant stakeholders

3.42
test item
test object
work product to be tested

EXAMPLE Software component, system, requirements document, design specification, user guide.

3.43
test level
one of a sequence of test stages, each of which is typically associated with the achievement of particular objectives and used to treat particular risks

EXAMPLE The following are common test levels, listed sequentially: unit/component testing, integration testing, system testing, system integration testing, acceptance testing.

Note 1 to entry: It is not always necessary for a *test item* (3.42) to be tested at all test levels, but the sequence of test levels generally stays the same.

Note 2 to entry: Typical objectives can include consideration of basic functionality for unit/component testing, interaction between integrated components for integration testing, acceptability to end users for acceptance testing.

3.44
test management
planning, scheduling, estimating, monitoring, reporting, control and completion of test activities

3.45
test management process
process used to coordinate, monitor and control *testing* (3.61)

EXAMPLE *Test strategy and planning process* (3.51), *test monitoring and control process* (3.48), *test completion process* (3.25).

3.46
test model
representation of the *test item* (3.42), which allows the testing to be focused on particular characteristics or qualities

EXAMPLE Requirements statements, equivalence partitions, state transition diagram, use case description, decision table, input syntax description, source code, control flow graph, parameters and values, classification tree, natural language.

Note 1 to entry: The test model and the required *test coverage* (3.28) are used to identify *test coverage items* (3.29).

Note 2 to entry: A separate test model can be required for each different type of required test coverage included in the test *completion criteria* (3.2).

Note 3 to entry: A test model can include one or more *test conditions* (3.27).

Note 4 to entry: Test models are commonly used to support test design (e.g. they are used to support test design in ISO/IEC/IEEE 29119-4, and they are used in model-based testing). Other types of models exist to support other aspects of testing, such as *test environment* (3.34) models, test maturity models and test architecture models.

3.47
test model specification
document specifying the *test model* (3.46)

3.48
test monitoring and control process
test management process (3.45) for ensuring that *testing* (3.61) is performed in line with a *test plan* (3.50) and with *organizational test specifications* (3.10)

3.49
test objective
reason for performing *testing* (3.61)

EXAMPLE Checking for correct implementation, identification of defects, measuring quality.

3.50
test plan
detailed description of *test objectives* (3.49) to be achieved and the means and schedule for achieving them, organized to coordinate *testing* (3.61) activities for some *test item* (3.42) or set of test items

Note 1 to entry: A project can have more than one test plan, for example there can be a project test plan (also known as a master test plan) that encompasses all testing activities on the project; further detail of particular *test* (3.21) activities can be defined in separate test plans (e.g. a system test plan or a performance test plan).

Note 2 to entry: A test plan is typically a written document, although other formats can be possible as defined locally within an organization or project.

Note 3 to entry: Test plans can also be written for non-project activities, for example a maintenance test plan.

3.51
test strategy and planning process
test management process (3.45) used to design the *test strategy* (3.59), complete test planning and create and maintain *test plans* (3.50)

3.52
test policy
organizational test policy
executive-level document that describes the purpose, goals, principles and scope of *testing* (3.61) within an organization

Note 1 to entry: The test policy defines what testing is performed and what it is expected to achieve but does not detail how testing is to be performed.

Note 2 to entry: The test policy can provide a framework for establishing, reviewing and continually improving the organization's testing.

3.53
test procedure
sequence of *test cases* (3.23) in execution order, with any associated actions required to set up preconditions and perform wrap-up activities post execution

3.54
test procedure specification
test script
document specifying one or more *test procedures* (3.53)

3.55
test process
set of *testing* (3.61) activities performed to achieve a *test objective* (3.49)

Note 1 to entry: The test process for a particular project can consist of multiple *test levels* (3.43) and *test types* (3.60).

3.56
test result
indication of whether a specific *test case* (3.23) has passed or failed, i.e. if the *actual results* (3.1) corresponds to the *expected results* (3.4) or if deviations were observed

3.57
test specification
complete documentation of the test design, *test cases* (3.23) and *test procedures* (3.53) for a specific *test item* (3.42)

Note 1 to entry: A test specification can be detailed in one document, in a set of documents, or in other ways, for example in a mixture of documents and database entries.

3.58
test status report
report that provides information about the status of the *testing* (3.61) that is being performed in a specified reporting period

3.59
test strategy

part of the *test plan* (3.50) that describes the approach to *testing* (3.61) for a specific project, *test level* (3.43) or *test type* (3.60)

Note 1 to entry: The test strategy usually describes some or all of the following: the test levels and test types to be implemented; the *retesting* (3.15) and *regression testing* (3.14) to be employed; the *test design techniques* (3.33) and corresponding test *completion criteria* (3.2) to be used; *test data* (3.30); *test environment* (3.34) and testing tool requirements; and expectations for test deliverables.

3.60
test type
testing (3.61) that is focused on specific quality characteristics

EXAMPLE *Security testing* (3.19), functional testing, usability testing, and *performance testing* (3.11).

Note 1 to entry: A test type can be performed at a single *test level* (3.43) or across several test levels (e.g. performance testing performed at a unit test level and at a system test level).

3.61
testing
set of activities conducted to facilitate discovery and/or evaluation of properties of one or more *test items* (3.42)

Note 1 to entry: Testing activities include planning, preparation, execution, reporting, and management activities, insofar as they are directed towards testing.

4 Conformance

4.1 Intended usage

4.1.1 General

The requirements in this document are contained in Clauses 4, 6, 7 and 8. This document provides requirements for a number of test processes suitable for use during the life cycle of a software system

or product. It is recognized that particular projects or organizations may not need to use all of the processes provided by this document. Therefore, implementation of this document typically involves selecting and declaring a set of processes suitable to the organization or project. There are two ways that an implementation can be claimed to conform to the provisions of this document – full conformance and tailored conformance.

The organization shall assert whether it is claiming full or tailored conformance to this document:

There are two criteria for claiming full conformance. Achieving either criterion suffices for conformance, although the chosen criterion (or criteria) shall be stated in the claim. Claiming "full conformance to tasks" asserts that all of the requirements of the activities and tasks of the declared set of processes are achieved. Alternatively, claiming "full conformance to outcomes" asserts that all of the required outcomes of the declared set of processes are achieved. Full conformance to outcomes permits greater freedom in the implementation of conforming processes and can be useful for implementing processes to be used in the context of an innovative life cycle model.

NOTE 1 Options for conformance are provided for needed flexibility in the application of this document. Each process has a set of objectives (phrased as "outcomes") and a set of activities and tasks that represent one way to achieve the objectives.

NOTE 2 Users who implement the activities and tasks of the declared set of processes can assert full conformance to tasks of the selected processes. Some users, however, can have innovative process variants that achieve the objectives (i.e., the outcomes) of the declared set of processes without implementing all of the activities and tasks. These users can assert full conformance to the outcomes of the declared set of processes. The two criteria — conformance to task and conformance to outcome — are necessarily not equivalent since specific performance of activities and tasks can require, in some cases, a higher level of capability than just the achievement of outcomes.

NOTE 3 When this document is used to help develop an agreement between an acquirer and a supplier, clauses of this document can be selected for incorporation in the agreement with or without modification. In this case, it is more appropriate for the acquirer and supplier to claim compliance with the agreement than conformance with this document.

NOTE 4 An organization (for example, national, industrial association, company) imposing this document, as a condition of trade, can specify and make public the minimum set of required processes, outcomes, activities, and tasks, which constitute suppliers' compliance with the conditions of trade.

NOTE 5 Requirements of this document are marked by the use of the verb "shall". Recommendations are marked by the use of the verb "should". Permissions are marked by the use of the verb "may". However, despite the verb that is used, the requirements for conformance are selected as described previously.

Organizations and projects that perform both scripted and exploratory testing can find that meeting all the requirements of some processes (e.g. Test Design and Implementation) is difficult when performing exploratory testing. In such cases, if the scripted testing meets all the requirements of the process, the organization or project can claim full conformance for the scripted testing (i.e. testers are allowed to execute additional informal tests that do not meet the full set of requirements defined in this document). Otherwise, if the organization's testing does not meet the requirements of specific clauses, then they can claim tailored conformance as per 4.1.3.

4.1.2 Full conformance

4.1.2.1 Full conformance to outcomes

A claim of full conformance declares the set of processes for which conformance is claimed. Full conformance to outcomes is achieved by demonstrating that all of the outcomes of the declared set of processes have been achieved. In this situation, the provisions for activities and tasks of the declared set of processes are guidance rather than requirements, regardless of the verb form that is used in the provision. One intended use of this document is to facilitate process assessment and improvement. For this purpose, the objectives of each process are written in the form of 'outcomes' compatible with the provisions of ISO/IEC 33002. ISO/IEC 33002 provides for the assessment of the processes of this document, providing a basis for improvement. Users intending process assessment and improvement