INTERNATIONAL STANDARD

ISO/ IEC/IEEE 29119-3

Second edition
2021-10

Software and systems engineering — Software testing —

Part 3: Test documentation

Ingénierie du logiciel et des systèmes — Essais du logiciel — Partie 3: Documentation des essais

Reference number
ISO/IEC/IEEE 29119-3:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
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 29119-3 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 29119-3:2013), which has been technically revised.

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

— the concept of test conditions has been replaced by test models, as feedback on the previous edition of this document highlighted a problem with users' understanding of 'test conditions' and their use for deriving test cases.

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 ISO/IEC/IEEE 29119 (all parts) is to define an internationally-agreed set of standards for software testing that can be used by any organization when performing any form of software testing.

ISO/IEC/IEEE 29119-1 introduces software testing concepts. This document uses the concepts of ISO/IEC/IEEE 29119-1.

ISO/IEC/IEEE 29119-2 comprises test process descriptions that define the software testing processes at the organisational level, test management level and dynamic test levels. It supports dynamic testing, functional and non-functional testing, manual and automated testing and scripted and unscripted testing, and can be utilized within any lifecycle model, including agile and traditional methodologies. Supporting diagrams describing the processes are also provided.

ISO/IEC/IEEE 29119-4 defines software test design techniques, which can be used within any lifecycle and for any product.

ISO/IEC/IEEE 29119-5 addresses the use of keyword-driven testing.

This document defines templates and provides examples of test documentation that are produced during the test process. An overview of the test documentation is provided in Figure 1. The templates are arranged within clauses reflecting the overall test process description structure in ISO/IEC/IEEE 29119-2, i.e. by the test process in which they are being produced. Annex A contains a list of all the information items identified in Clauses 6, 7 and 8 with the corresponding level of conformance (shall/should/may) from ISO/IEC/IEEE 29119-2. Annex B contains an overview of the examples. Annexes C to R contain examples of the application of the templates. Annex S provides mappings to existing standards. Annex T explains why the concept of test conditions has been replaced by test models in this document. A Bibliography is provided at the end of the document.

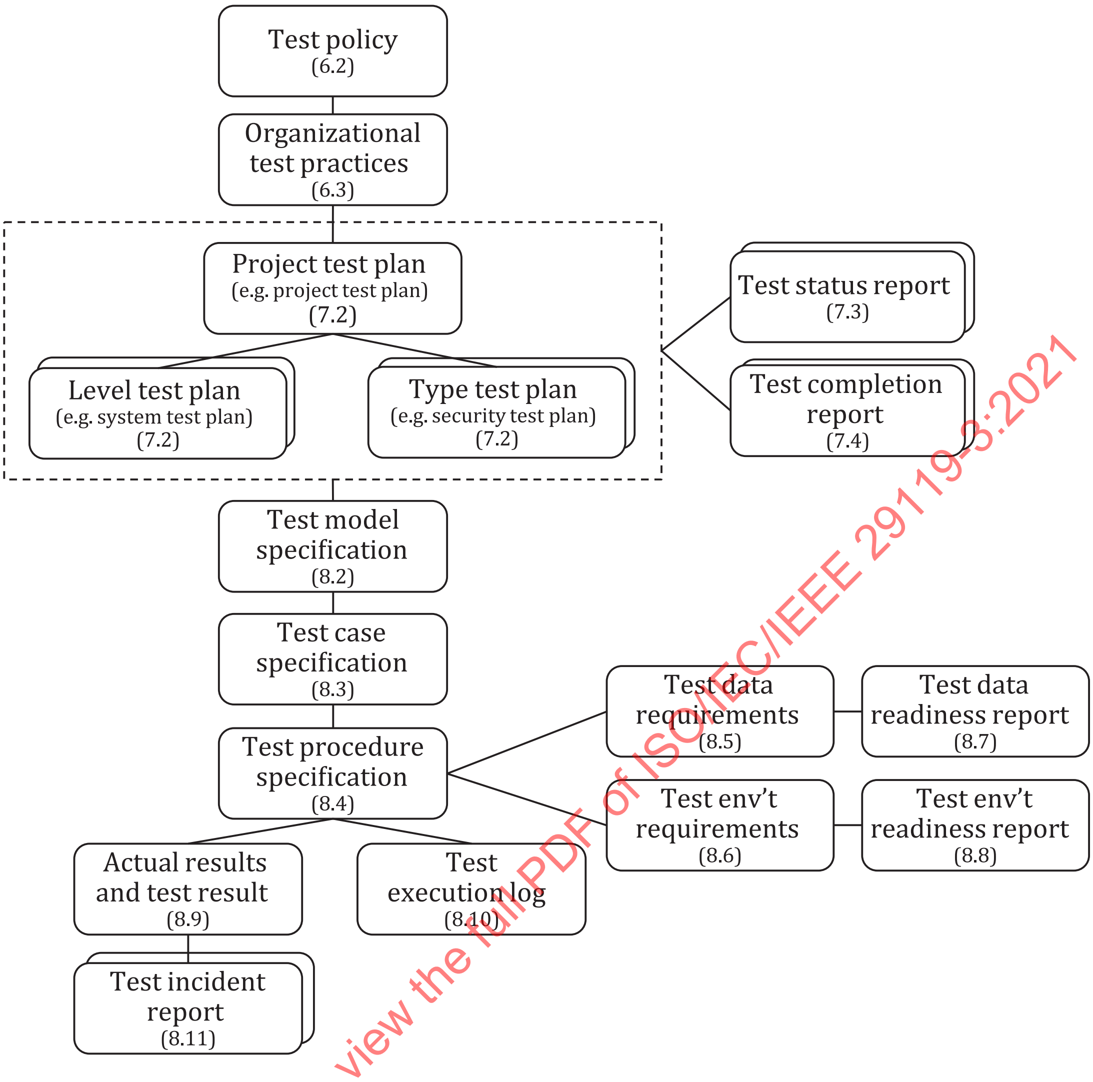


Figure 1 — Overview of test documentation

The test documentation described in this document can be on paper or in electronic form (e.g. records in test tools, spreadsheets, mind maps, white board photos).

The nomenclature of test documentation within this document (e.g. document names, section headings) and the contents of each document can be tailored to suit the unique needs of an organization, under the tailoring clause (see Clause 4).

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.

ISO/IEC/IEEE 29119 (all parts) aims to provide stakeholders with the ability to manage and perform software testing in any organization. This document can be adopted under any lifecycle methodology including traditional (e.g. waterfall, iterative), agile or DevOps.

Software and systems engineering — Software testing —

Part 3: Test documentation

1 Scope

This document specifies software test documentation templates that can be used for any organization, project or testing activity. It describes the test documentation that is an output of the processes specified in ISO/IEC/IEEE 29119-2.

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

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, 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 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 https://www.computer.org/sevocab.

3.1
actual results
set of behaviours or conditions of a test item, or set of conditions of associated data or the test environment, observed as a result of test execution

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

3.2
expected results
observable predicted behaviour of the test item under specified conditions based on its specification or another source

3.3
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

3.4
incident report
documentation of the occurrence, nature, and status of an *incident* (3.3)

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.

3.5
organizational test practices
documentation that expresses the recommended approaches or methods for the testing 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 *test policy* (3.20).

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.

3.6
organizational test specification
documentation that provides information about testing for an organization, i.e. information that is not project specific

EXAMPLE The most common examples of organizational test specifications are the *test policy* (3.20) and *organizational test practices* (3.5).

3.7
test basis
information used as the basis for designing and implementing test cases

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.

3.8
test case specification
documentation of a set of one or more test cases

3.9
test completion report
test summary report
report that provides a summary of the testing that was performed

3.10
test data readiness report
documentation describing the status of each test data requirement

3.11
test environment item
element of a test environment that can be considered separately from other parts of the test environment

EXAMPLE Hardware, software, interfaces, peripherals, tools.

3.12
test environment readiness report
documentation that describes the status of each test environment requirement

Note 1 to entry: This can list the status of each of the *test environment requirements* (3.13).

3.13
test environment requirements
documentation of the necessary properties of the test environment

Note 1 to entry: All or parts of the test environment requirements can reference where the information can be found, e.g. in the appropriate *organizational test practices* (3.5) document, *test plan* (3.19) and/or *test specification* (3.23).

3.14
test execution log
record of the execution of one or more test procedures

3.15
test incident
event occurring during the execution of a test that requires investigation

3.16
test model
representation of the test item, 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, source code, control flow graph, parameters and values, classification tree, natural language.

Note 1 to entry: The test model and the required test coverage are used to identify test coverage items.

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.

Note 3 to entry: A test model can include one or more test conditions.

Note 4 to entry: Test models are commonly used to support test design (e.g. they are used to support the 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 models, test maturity models and test architecture models.

3.17
test model specification
documentation specifying the *test model* (3.16)

3.18
test organization
management structure responsible for testing within an organization

Note 1 to entry: The test organization is typically technically, managerially and financially independent from the development organization.

3.19
test plan
detailed description of test objectives to be achieved and the means and schedule for achieving them, organized to coordinate testing activities for some test item 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 activities can be defined in one or more test level / test type 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.

3.20
test policy
organizational test policy
executive-level documentation that describes the purpose, goals, principles and scope of testing 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.

3.21
test procedure specification
test script
documentation specifying one or more test procedures

3.22
test result
indication of whether or not a specific test case has passed or failed, i.e. if the *actual results* (3.1) corresponds to the *expected results* (3.2) or if deviations were observed

3.23
test specification
complete documentation of the test design, test cases and test procedures for a specific test item

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.

3.24
test status report
report that provides information about the status of the testing that is being performed in a specified reporting period

3.25
test strategy
part of the *test plan* (3.19) that describes the approach to testing for a specific project, test level or test type

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 and regression testing to be employed; the test design techniques and corresponding test completion criteria to be used; test data; test environment and testing tool requirements; and expectations for test deliverables.

3.26
test traceability matrix
verification cross reference matrix
requirements test matrix
requirements verification table
document, spreadsheet, or other tool used to identify related items in documentation and software, such as requirements with associated tests

Note 1 to entry: Different test traceability matrices can have different information, formats, and levels of detail.

4 Conformance

4.1 Intended usage

4.1.1 General

The requirements in this document are contained in Clauses 4, 5, 6, 7, 8 and Annex A.

This document provides requirements for test documentation across the complete software testing lifecycle. Test documentation shall be considered as conforming if it is available in an electronic format (e.g. records in a test management tool or spreadsheet), divided into separate documents, or combined with other documents into one document.

In this document, for simplicity, each item of test documentation is described as if it were a separate hardcopy document. Test documentation titles, headings and layout described in this document may be modified (e.g. added to, combined or re-titled). The use of the nomenclature used in Clauses 5, 6, 7 and 8 is not mandatory.

The information items identified in Clauses 6, 7 and 8 correspond to the outputs of the test processes in ISO/IEC/IEEE 29119-2. Annex A provides an overview of the requirements for clauses in ISO/IEC/IEEE 29119-2 where the creation of the information items defined in Clauses 6, 7 and 8 are described.

It is recognized that some organizations, projects or teams may not need to use all of the test documentation defined by this document. Therefore, implementation of this document typically involves selecting a set of test documentation suitable for the organization, project or team. 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.

NOTE 1 Options for conformance are provided for needed flexibility in the application of this document.

NOTE 2 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 3 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 test documentation, which constitute suppliers' compliance with the conditions of trade.

NOTE 4 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.

4.1.2 Full conformance

Full conformance shall be achieved by providing evidence that all requirements (i.e. shall statements) of the test documentation defined in Clauses 5 to 8 and Annex A have been met.

4.1.3 Tailored conformance

When this document is used as a basis for establishing a set of test documentation that does not qualify for full conformance, the subset of test documentation for which tailored conformance is claimed, shall be recorded. Tailored conformance is achieved by demonstrating that all requirements (i.e. shall statements) for the recorded subset of test documentation defined in Clauses 5 to 8 and Annex A have been satisfied.

Where tailoring occurs, justification shall be provided (either directly or by reference), whenever the requirements (i.e. shall statements) of test documentation defined in Clauses 5 to 8 and Annex A are not met. All tailoring decisions shall be recorded with their rationale, including the consideration of any applicable risks. Tailoring decisions shall be agreed by the relevant stakeholders.

EXAMPLE Where organizations follow information item management processes in standards such as ISO 15489-1 or ISO 9001 or use similar internal organizational processes, they can decide to use information items produced by those processes in place of the test documentation defined in this document.

5 Common information for all test documentation

5.1 Overview

This clause defines the common information elements that shall be captured for all test documentation types covered in Clauses 6, 7 and 8. The common information elements listed in 5.2 shall be included in each applicable piece of test documentation.

NOTE The test documentation described in this document can be issued in several versions over time. Whilst a simple level of version control information is defined in Clause 5, the overall handling of multiple versions of test documentation (including reviewing and maintaining test documentation) is out of scope of this document, as that is a configuration management issue that is handled in other ISO standards.

5.2 Common information elements

5.2.1 Unique identifier

Uniquely identifies the test documentation or record.

EXAMPLE The unique identifier can include the title of the test documentation, the date of issue, version, and/or documentation status (e.g. draft, reviewed, corrected, final).

5.2.2 Issuing organization

Specifies the organization responsible for preparing and releasing the test documentation or record. It may also include authors.

5.2.3 Approval authority

Identifies the designated persons who have the responsibility for signing off on the test documentation or record (possibly electronically). It may also include reviewers and pertinent managers.

5.2.4 Change history

Includes a log of all of changes made to the test documentation or record since its inception.

EXAMPLE This can consist of a table that includes the present version of the test documentation and any predecessor versions, with the unique identifier of each version, a description of documentation changes with respect to the previous version, the name and role of the person making the changes and reasons for changes. Reasons for change can include addressing audit comment, team review comments or system changes. The person making the change can include the test documentation author, project manager or system owner.

5.2.5 Status

Defines the status of each item of test documentation.

EXAMPLE Test documentation status can include "not yet started", "in design", "in review" and "approved."

5.2.6 Introduction

Explains the context and structure of the test documentation or record.

5.2.7 Scope

Identifies the extent of the coverage of the subject area by the test documentation or record, and describes any inclusions, exclusions, assumptions and/or limitations.

5.2.8 References

Lists other referenced documents or records and identifies repositories for system, software, and test information. The references may be separated into “external” references that are imposed from outside the organization and “internal” references that are imposed from within the organization.

EXAMPLE Referenced documents can be policies, plans, procedures, and other source data.

5.2.9 Glossary

Provides a lexicon for the terms, abbreviations, and acronyms, if any, used in the test documentation or record.

NOTE This section can be an annex, or it can refer to another item of test documentation that provides a general glossary. All or part of the glossary and/or acronym list can be online, as a separate testing specific glossary or incorporated in a larger organizational glossary (including more terms than just those that are related to testing).

6 Organizational test process documentation

6.1 Overview

Organizational test specifications describe information about testing at the organization level and are not project dependent. Typical examples of organizational test specifications developed in the organizational test process include:

- test policy;
- organizational test practices.

The full templates with explanatory text for these items of test documentation can be found in 6.2 and 6.3. Annex A provides an abbreviated overview of each item of test documentation along with the requirements, recommendations and permissions (shall, should and may) for each section of the test documentation. Annexes C and D provide examples of a test policy and organizational test practices for example projects.

6.2 Test policy

6.2.1 Overview

The test policy defines the objectives and principles of software testing to be applied within the organization. It defines what should be accomplished by testing, but does not detail how testing is performed. The policy provides a framework for establishing, reviewing, and continually improving the organization’s test policy.

A.3 provides an outline of the organizational test policy, while C.1 and C.2 provide examples that demonstrate how Organizational Test Policies can be developed for two different example projects.

The contents of the test policy are specified in 6.2.2 to 6.2.11.

6.2.2 Objectives of testing

Describes the purpose, goals, and overall scope of the testing within the organization. States the organization’s position on why testing is performed and what they look to achieve.

6.2.3 Test process

Identifies the test process that the organization will follow. This may include reference to a specific document providing details of the test process.

EXAMPLE Such a document can be a test process in ISO/IEC/IEEE 29119-2. The details of activities in the test process can be described in more detailed test process documentation.

6.2.4 Test organization structure

Identifies the roles and structure of the test organization. A diagram to show test organization hierarchy may be used, or the information may be presented in a table.

6.2.5 Tester training

States required training and certifications for individuals performing testing in the organization.

6.2.6 Tester ethics

Identifies the organizational ethics code to be upheld by the testers.

6.2.7 Standards

States which standards are applicable to the testing performed within the organization.

6.2.8 Other relevant policies

Identifies policies that impact the testing performed within the organization.

EXAMPLE A policy statement can be that testing will conform to the quality policy.

6.2.9 Measuring the value of testing

States how the organization determines the return on investment of testing. Identifies the objectives for measuring the value of testing.

6.2.10 Test asset archiving and reuse

States the organization's position on the archiving and reuse of test assets.

6.2.11 Test process improvement

States the method for ensuring continuous improvement of the test process.

6.3 Organizational test practices

6.3.1 Overview

The organizational test practices is technical test documentation that provides guidelines on how testing should be carried out within the organization, i.e. how to achieve the objectives stated in the test policy, where one exists.

The organizational test practices provides guidelines to projects within its scope; it is not project specific.

For small or highly homogenous organizations, a single set of organizational test practices documentation may cover all testing activities. An organization may have more than one set of organizational test practices documentation if the organization performs testing in a number of significantly different ways, such as for both safety-critical products and non-safety-critical products;

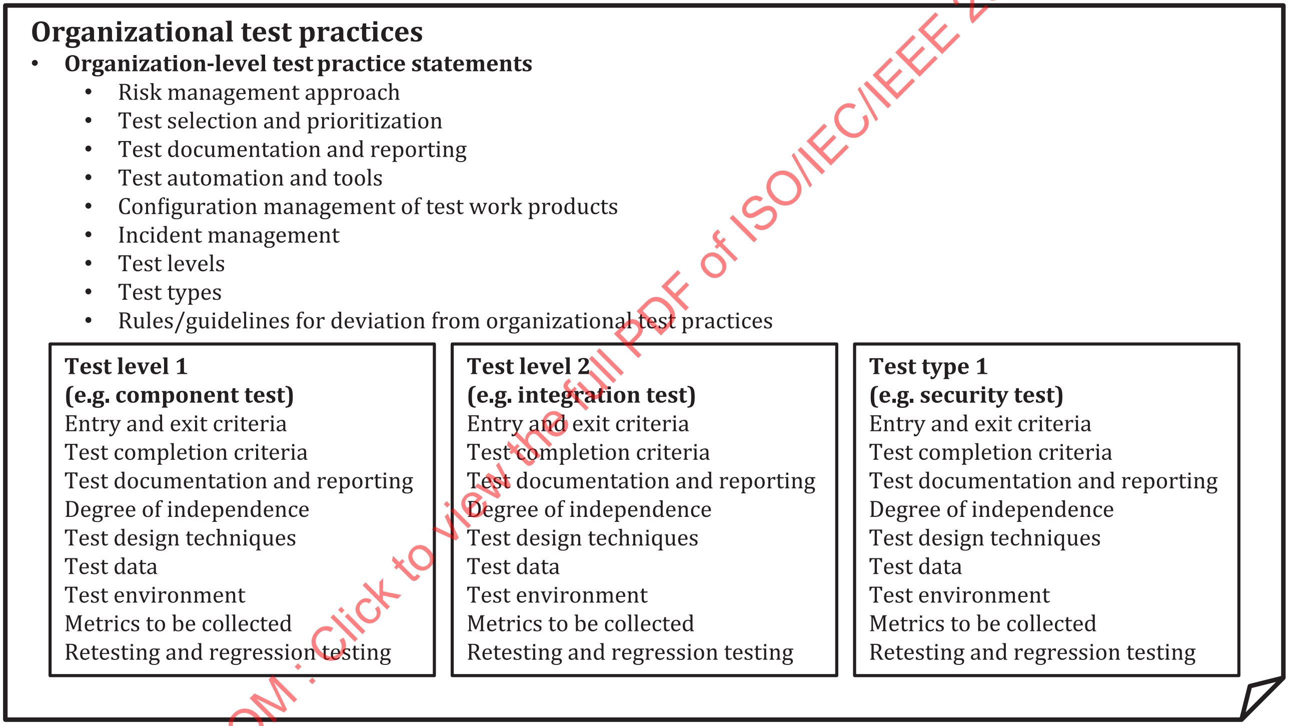
or if it is using both agile and V-model development models; or if its programmes are large enough to merit their own documented test practices.

The organizational test practices may incorporate the content of the test policy where no separate test policy is available.

The practices are defined for the specified scope – such as at the level of a programme, project or for a specific level or type of testing.

The organizational test practices includes identification of test levels expected to be used and types and practice statements for each of these. It may be partitioned with a subsection for each identified test level and test type if the test level or test type practice statements differ significantly between each test level or type; this is illustrated in Figure 2.

The contents of the organizational test practices are defined in 6.3.2 and 6.3.3. A.3 provides an outline of the organizational test practices documentation, while Annex D provides examples that demonstrate how organizational test practices can be developed for two example organizations.



Organizational test practices

- **Organization-level test practice statements**
  - Risk management approach
  - Test selection and prioritization
  - Test documentation and reporting
  - Test automation and tools
  - Configuration management of test work products
  - Incident management
  - Test levels
  - Test types
  - Rules/guidelines for deviation from organizational test practices

Test level 1
(e.g. component test)
Entry and exit criteria
Test completion criteria
Test documentation and reporting
Degree of independence
Test design techniques
Test data
Test environment
Metrics to be collected
Retesting and regression testing

Test level 2
(e.g. integration test)
Entry and exit criteria
Test completion criteria
Test documentation and reporting
Degree of independence
Test design techniques
Test data
Test environment
Metrics to be collected
Retesting and regression testing

Test type 1
(e.g. security test)
Entry and exit criteria
Test completion criteria
Test documentation and reporting
Degree of independence
Test design techniques
Test data
Test environment
Metrics to be collected
Retesting and regression testing

Figure 2 — An example structure of the organizational test practices document

6.3.2 Organization-level test practice statements

6.3.2.1 General

This section includes statements that are apply to all of the test levels and test types to be performed in projects run within the organization (or that part of the organization that is within the scope of the organizational test practices documentation). This section may include subsections from the test policy, if needed.

6.3.2.2 Risk management approach

Identifies the generic approach to risk management expected to be used to direct the testing activities.

EXAMPLE This can be to identify project and product risks that can be mitigated or reduced by testing.

6.3.2.3 Test selection and prioritization

Describes the approach to selecting and prioritizing test execution, in the form of prioritized test procedures. Test procedures consist of prioritized test cases, derived from prioritized test coverage items.

6.3.2.4 Test documentation and reporting

Identifies the test documentation expected to be produced during testing by each project being run within the organization. Describes when each documentation type is prepared and the associated approval process. This is tightly connected to the test process that is specified in the test policy.

6.3.2.5 Test automation and tools

Describes the approach to test automation. Identifies the test tools to be used throughout testing.

EXAMPLE This can include test management tools, test execution tools, performance test tools, security test tools and usability test tools.

6.3.2.6 Configuration management of test work products

Describes the configuration management to be performed for the work products from testing; describes how these work products are to be identified, traced, stored, and made available to stakeholders.

6.3.2.7 Incident management

Describes how incidents should be managed during testing or refers to a description elsewhere.

6.3.2.8 Test levels

Identifies specific test levels to be performed as part of the testing within the scope of the practices.

6.3.2.9 Test types

Identifies specific types of testing to be performed as part of testing within the scope of the practices.

6.3.2.10 Rules/guidelines for deviation from organizational test practices

Defines rules or guidelines for determining the conditions under which deviation from organizational test practices is possible/allowed within the organization.

EXAMPLE An organization can decide that individual projects or teams cannot change the defect severity and priority levels that are set out in the organizational test practices.

6.3.3 Test level/type-specific organizational test practice statements

6.3.3.1 Entry and exit criteria

Criteria that specify when test processes for a given test level or type can start and stop.

Each test level and type consists of the following test processes:

- test planning;
- test design and implementation;
- test environment and data management;
- test execution;

— test incident reporting.

Different entry and exit criteria may be defined for each of these test processes, or for selected ones, or for the entire test level or type as a whole.

EXAMPLE Entry criteria for test execution during system testing can include: the system test plan has been created and signed off, test cases have been created and approved by business stakeholders and test data and the test environment (including deployment of the test item) are prepared and ready for execution. Exit criteria for system testing can include: all high priority test cases have been executed and passed testing or that any severity 1 or 2 defects that remain open have been reviewed and approved by the business as not requiring addressing during system testing.

6.3.3.2 Test completion criteria

Criteria that specify when test execution for a given test level, test type or test technique can be considered complete. For test types and test levels, the test completion criteria may be the same as the exit criteria for that test level.

EXAMPLE The test completion criterion for state transition testing can be that test execution will continue until all states in the state model have been "visited" during testing.

6.3.3.3 Test documentation and reporting

Identifies the test documentation, including reporting, used for testing activities in the test level or type. Describes when each is prepared and the associated approval process. This is tightly connected to the test process specified in the test policy.

6.3.3.4 Degree of independence

Establishes the level of independence of those performing the testing. States how testers or groups of testers are technically, managerially, and financially independent.

6.3.3.5 Test design techniques

Identifies specific test design techniques to be used during test design and implementation within the test level or type.

6.3.3.6 Test data

Identifies the test data required for the test level or type. This may include a statement of the origin of test data, where particular types of test data are located and groups or organizations responsible for creating and maintaining test data.

NOTE The data quality model defined in ISO/IEC 25012 can support the identification of test data requirements (and potentially data measurement needs).

6.3.3.7 Test environment

Identifies the test environment for the test level or type. This may include where specific types of test should be performed and may identify groups or organizations responsible for the test environment.

6.3.3.8 Metrics to be collected

Describes the metrics for which values are to be collected during the test activities in the test level or type.

6.3.3.9 Retesting

Identifies the practices, conditions and activities for retesting in each test level or type.

6.3.3.10 Regression testing

Identifies the practices, conditions and activities for regression testing in each test level or type.

7 Test management processes documentation

7.1 Overview

The test documentation developed in the test management processes comprise the following types:

- test plan;
- test status report;
- test completion report.

The full templates with explanatory text for each of these items of test documentation are specified in 7.2 to 7.4. Annex A provides an abbreviated overview of each item of test documentation along with the requirements, recommendations and permissions (shall, should and may) for each section of the test documentation. Annexes E, F and G provide examples of test plans, test status reports, and test completion reports for example projects.

7.2 Test plan

7.2.1 Overview

The test plan is an item of test documentation that is used for test planning and test management. It describes the decisions made during initial test planning and evolves as re-planning is performed as part of the control activity. Some projects may have a single test plan, while for larger projects multiple test plans may be produced. Test plans may apply across multiple projects (at the programme level), or to a single project (i.e. project test plan/master test plan), or to a specific test level or type (e.g. system test plan, integration software test plan, subsystem test plan, subcontractor software test plan, unit software test plan, or performance test plan), or to a specific iteration of testing. If multiple software test plans are created, a mapping tree may be produced to aid the documentation of relationships and the information contained in each.

If the organization does not yet have organizational test practices defined, then there may be sections from that test documentation that need to be included in the test plan. This may include the test selection and prioritization (6.3.2.3), test automation and tools (6.3.2.5), configuration management of test work products (6.3.2.6) and incident management (6.3.2.7), as these are not repeated in the test plan template defined in this subclause.

A.3 provides an outline of the test plan, while Annex E provides examples that demonstrate how test plans can be developed for two different example projects.

The contents of the test plan are specified in 7.2.2 to 7.2.10.

7.2.2 Context of testing

7.2.2.1 Projects / test levels / test types

Identifies the projects, test levels and/or test types for which the test plan is being written and other relevant contextual information.

7.2.2.2 Test items

Identifies the test items for the testing covered by this test plan including version/revision information or a reference for where this information can be found.

This section may describe the purpose of the test item or reference where this information can be found.

NOTE This information can be defined in a system definition document, such as a concept of operations.

EXAMPLE The test item can be a software unit, a group of units or modules, a service, a subsystem, or a complete system, app or product.

7.2.2.3 Test scope

Summarizes the features of each test item to be tested. Also identifies any features of a test item that are to be specifically excluded from testing and the rationale for their exclusion.

EXAMPLE Features to be tested can be specific functions, interfaces, business processes or attributes of the software, such as usability or performance.

7.2.2.4 Test basis

Identifies the basis for designing and implementing tests.

7.2.3 Assumptions and constraints

Describes any assumptions and constraints for the testing covered by this plan. These may include regulatory standards, requirements in the test policy and the organizational test practices, contractual requirements, project time and cost constraints, and availability of appropriately skilled staff, tools and/or environments.

7.2.4 Stakeholders

Lists the stakeholders and their relevance to the testing.

EXAMPLE 1 This can include the stakeholder with the authority for resolving issues raised as a result of the testing activities and the stakeholder with the authority for approving test products and processes.

EXAMPLE 2 This information can be captured in a RACI (responsible, accountable, consulted, informed) matrix.

7.2.5 Testing communication

Describes the lines of communication between testing, other lifecycle activities, and within the organization.

This information may be represented visually.

NOTE A visual representation can include an organization chart or a figure that illustrates the flow of information.

7.2.6 Risk register

7.2.6.1 General

Lists the risks identified as part of the test planning activity. This should include any relevant risks that may be specified in the organizational test practices. Provides an exposure level for each risk based on its impact and probability. Provides recommendations to manage the risks. This section may reference a separate risk register.

EXAMPLE Recommendations to manage risk can include to eliminate, reduce, monitor or accept a risk.

NOTE A risk register can be located in a project plan or a risk management plan.

7.2.6.2 Product risks

Identifies test-related product risks.

EXAMPLE Test-related product risks can include defects in functionality or in non-functional aspects such as performance.

7.2.6.3 Project risks

Identifies test-related project risks.

EXAMPLE Test-related project risks can include risks related to schedule or resources.

7.2.7 Test strategy

7.2.7.1 General

Describes the approach to testing for the identified scope of testing of the test item, as outlined in 7.2.7.2 to 7.2.7.15. It may refer to the organizational test practices stating only its differences from it.

7.2.7.2 Test levels

For a project test plan or a type test plan, this identifies the levels of testing that will be conducted.

7.2.7.3 Test types

For a project test plan or a level test plan, this identifies the types of testing that will be conducted.

7.2.7.4 Test deliverables

Identifies all test documentation that is to be delivered from the testing activity or equivalent information to be recorded electronically, for example in spreadsheets, databases or dedicated test tools.

EXAMPLE The following test documentation can be included:

- test plan;
- test model specification;
- test case specification;
- test procedure specification;
- test data readiness report;
- test environment readiness report;
- incident reports;
- test status reports;
- test completion report.

Test input data and test output data may be identified as deliverables. Test tools created as part of the testing activity may also be included. If test documentation has been combined or eliminated, then this list will be modified accordingly.

This subsection may include information on when each item of test documentation should be delivered, and name the sender and recipient (preferably by position, not name).

7.2.7.5 Test design techniques

Specifies which test design techniques are to be applied.

7.2.7.6 Entry and exit criteria

Specifies when test processes for a given test level or type can start and stop.

Each test level and type consists of the following processes:

- test planning;
- test design and implementation;
- test environment and data management;
- test execution;
- test incident reporting.

Different entry and exit criteria may be defined for each of these test processes, or for selected ones, or for the entire test level or type as a whole.

EXAMPLE Entry criteria for test execution during system testing can include: the system test plan has been created and signed off, test cases have been created and approved by business stakeholders and test data and the test environment (including deployment of the test item) are prepared and ready for execution. Exit criteria for system testing can include: all high priority test cases have been executed and passed testing or that any severity 1 or 2 defects that remain open have been reviewed and approved by the business as not requiring addressing during system testing.

7.2.7.7 Test completion criteria

Specifies when test execution for a given test level, test type or test technique can be considered complete. For test types and test levels, the test completion criteria may be the same as the exit criteria for that test level.

EXAMPLE The test completion criterion for state transition testing can be that test execution will continue until all states in the state model have been "visited" during testing.

7.2.7.8 Degree of independence

Establishes the level of independence of those performing the testing. States how the testers are technically, managerially, and financially independent.

7.2.7.9 Metrics to be collected

Describes the metrics for which values are to be collected during the test activities.

7.2.7.10 Test data requirements

Specifies all relevant test data requirements for the project or test level or test type (as appropriate).

EXAMPLE This can identify the origin of the test data and state where specific test data is located, whether data has to be sanitised for confidentiality reasons, and the role responsible for the test data.

These test data requirements may be deferred to the test data requirements documentation (see 8.5), as applicable.

NOTE The data quality model defined in ISO/IEC 25012 can support the identification of test data requirements (and potentially data measurement needs).

7.2.7.11 Test environment requirements

Specifies the necessary and desired properties of the test environment.

EXAMPLE 1 This can include hardware, software, test tools, databases, and personnel (identifying their organizations, as appropriate).

Information regarding selection, evaluation, acquisition and support for each component part of the test environment may be included.

It may include test environment requirements for test preparation, test execution (including data capture), and any post-execution activities.

EXAMPLE 2 A post-execution activity can be data analysis.

These test environment requirements may be defined in the test environment requirements documentation (see 8.6) as applicable, but this should be stated in the test plan.

7.2.7.12 Retesting

Specifies the conditions under which retesting will be performed.

7.2.7.13 Regression testing

Specifies the conditions under which regression testing will be performed.

7.2.7.14 Suspension and resumption criteria

Specifies the criteria used to suspend and resume all or a subset of the testing activities in the test plan. Identifies who has the authority to suspend and resume testing activities. Specifies the activities that need to be completed when testing resumes, which may include repeating specific testing activities.

7.2.7.15 Deviations from the organizational test practices

Records any test plan content that deviates from the organizational test practices and the reasons for deviation. Identifies those responsible for approving deviations, where applicable.

7.2.8 Testing activities and estimates

Identifies all the necessary testing activities required to implement the test strategy. The approach and expected number of iterations for retesting and regression testing as well as any dependencies between activities should be considered.

NOTE 1 The testing activities can be described in terms of a work breakdown structure or activities on an agile activity board.

Describes estimates for each identified testing activity. Additionally, where appropriate, describes the allocated testing budget and cost estimates or references where that information can be found.

NOTE 2 Allocated budget and cost estimates can be located in the project plan.

7.2.9 Staffing

7.2.9.1 General

Describes the staffing requirements for the testing covered by this plan.

NOTE Staffing can be accomplished by internal transfer, external hiring, consultants, subcontractors, business partners, and/or outsourced resources.

7.2.9.2 Roles and responsibilities

Provides an overview of the primary (they are the activity leader) and secondary (they are not the leader but provide support) people filling the test-related roles and their corresponding responsibilities and authority for the testing activities. In addition, identifies those responsible for providing the test items. They may be participating either full-time or part-time.

For each testing person, specify the periods when the person is required.

EXAMPLE 1 The responsible parties can include the project manager, the test manager, the developers, the test analysts and executors, operations staff, user representatives, technical support staff, data administration staff, and quality support staff.

EXAMPLE 2 This information can be captured in an RACI matrix.

7.2.9.3 Hiring needs

Identifies specific requirements for additional testing staff that are needed to perform the specified testing. Specifies when the staff are needed, if they should be temporary, full-time or part-time, and their desired skill set. These may be constrained by contract and business needs.

7.2.9.4 Training needs

Specifies test training needs by skill level and identifies training options for providing the necessary skills for the staff needed.

EXAMPLE Training can take a variety of forms, such as traditional classroom training, self-paced computer-based training, training over the Internet, and mentoring by more knowledgeable staff members.

7.2.10 Schedule

Identifies test milestones defined in the project schedule and test activities from the test strategy. Summarizes the overall schedule of the testing activities, identifying where activity results feed back to the development, organizational, and supporting processes. Specifies the schedule for each testing activity and test milestones based on the activity estimates, available resources, and other constraints.

EXAMPLE Supporting processes can be quality assurance and configuration management.

7.3 Test status report

7.3.1 Overview

The test status report provides information about the status of the testing that is performed in a specific reporting period.

NOTE In agile delivery, the test status report might not be a written document. For example, test status and progress information can be shared verbally by testers at a daily stand-up meeting. When evidence of test status reporting is not recorded, it constitutes tailored conformance (see 4.1.3).

A.3 provides an outline of the test status report, while Annex F provides examples that demonstrate how test status reports can be developed for two different example projects.

The contents of the test status report are specified in 7.3.2 to 7.3.8.

7.3.2 Test status

Provides information on the status of the testing for the reporting period.

EXAMPLE Test status can be ahead of schedule, on time, behind schedule.

7.3.3 Reporting period

Specifies the time period covered by the report.

7.3.4 Progress against test plan

Describes the progress that has been made against the test plan. Any notable deviations from the plan should be highlighted, with explanations of the reasons for deviation, descriptions of any remedial actions, an account of the effects, and consideration of the implications with regard to planned project objectives.

7.3.5 Factors blocking progress

Identifies those factors that impeded progress during the reporting period and the corresponding solutions that were implemented to resolve them. Outstanding (unsolved) issues still impeding progress should be recorded and possible solutions identified.

7.3.6 Test measures

Presents the collated test measures related to the reporting period.

EXAMPLE This can include measures on test cases, defects, incidents, test coverage, activity progress and resource consumption.

7.3.7 New and changed risks

Lists the new risks that have been identified as a result of testing as well as changes to existing risks during the reporting period.

7.3.8 Planned testing

Describes the planned testing for the next reporting period.

7.4 Test completion report

7.4.1 Overview

The test completion report provides a summary of the testing that was performed. This may be for the project/programme as a whole or for the particular test level or test type.

A.3 provides an outline of the test completion report, while Annex G provides examples that demonstrate how test completion reports can be developed for two different example projects.

The contents of the test completion report are specified in 7.4.2 to 7.4.10.

7.4.2 Summary of testing performed

Summarizes the testing performed across the project and/or in the test levels and test types that are in scope for this report.

Provides details on what was tested and describes any constraints on how the testing was performed.

EXAMPLE This can include restrictions on test environment availability.

7.4.3 Deviations from planned testing

Describes deviations from the planned testing, if any. This section may also reference the section on residual risks for any risks that the deviations pose to the testing and their relevant risk treatments.

7.4.4 Test completion evaluation

Describes the extent to which the testing met the specified test completion criteria, and where necessary, explains why the criteria were not met. This section may also reference the section on residual risks for any risks that any incomplete completion criteria pose and their corresponding risk treatments.

7.4.5 Factors that blocked progress

Describes those factors that impeded progress and the corresponding solutions that were implemented to remove them.

7.4.6 Test measures

Presents the collated test measures.

EXAMPLE This can include measures for test cases, defects, incidents, test coverage, activity progress, and resource consumption.

7.4.7 Residual risks

Lists the risks that are untreated at the end of the testing; this may be risks that have not been fully treated by the testing and /or any new risks identified as a result of the final monitoring and closure of the test.

7.4.8 Test deliverables

Lists all the test deliverables produced as a result of the testing and their location.

EXAMPLE This can include the test plan, test case specifications and test procedure specifications.

7.4.9 Reusable test assets

Lists all the reusable test assets and their location.

EXAMPLE This can include test procedures and test data.

7.4.10 Lessons learned

Describes the results of the lessons learned meeting.

NOTE If the project is following an agile methodology, the lessons learned may have been collected and captured iteratively during retrospectives. Key findings that are to be brought into future test levels, types or projects will still typically be documented in the test completion report.

8 Dynamic test processes documentation

8.1 Overview

The test documentation developed in the dynamic test processes comprise the following types:

- test specification, divided into:
  - test model specification;
  - test case specification;
  - test procedure specification;

NOTE 1 These can be separate documents, can appear as chapters in a test specification, can be in the form of charters, or can be electronic records in a test tool, depending on the size and nature of the test project.

- test data requirements;
- test environment requirements;
- test data readiness report;
- test environment readiness report;
- test execution documentation, divided into:
  - actual results;
  - test results;
  - test execution log;
- incident report.

The full templates with explanatory text for each of these items of test documentation are found in 8.2 to 8.11. Annex A provides an abbreviated overview of each item of test documentation along with the requirements, recommendations and permissions (shall, should and may) for each section of the test documentation. Annexes H to R provide examples of the dynamic test process documentation for example projects.

NOTE 2 There are many test documentation styles and names, e.g. in agile, session sheets and charters with test ideas are often used. It is expected that when tailoring is required (see 4.1.3), other test documentation names or headings can be substituted for the names used in this document. A mapping can be produced as a part of that tailoring process. In Annexes B to R, examples for two different project types are presented with options for tailoring of test documentation names. The annexes are not inclusive of all test documentation names, formats, or test methodologies, but are intended to demonstrate some possible options.

8.2 Test model specification

8.2.1 Overview

A test model is a representation of an aspect of the test item, which is focused on the attributes that define required test coverage. The test model and the required test coverage are used to derive the test coverage items.

A test model will either already exist (e.g. source code or state transition diagram used for development) or can be documented in a test model specification.

A.3 provides an outline of the test model specification, while Annex H provides examples that demonstrate how test models can be described for two different example projects.

EXAMPLE Examples of test models are equivalence classes, a set of identified boundaries on the edges of each equivalence class, a decision table, a state transition diagram, source code, and a control flow graph.

The contents for a test model specification are specified in 8.2.2 to 8.2.7.

8.2.2 Unique identifier

Describes the unique identifier for the test model so that it can be distinguished from all other test models. An automated tool may control the generation of the identifiers, or it may be done manually according to an appropriate notation scheme. The unique identifier is not usually changed during the lifetime of the test model, because it is used for traceability purposes.

8.2.3 Objective

Identifies and briefly describes the particular focus of the test model.

8.2.4 Priority

Defines the priority of the testing associated with this particular test model, if needed.

8.2.5 Test strategy extract

Includes the part of the test strategy that describes the relevant test design technique and test completion criterion. This may also include whether the test will be automated or manually executed.

8.2.6 Test model

Presents the test model. The notation used for the test model will depend on the required test coverage. Thus, it can be written in natural language, formal language, expressed in a tabular format, graphical format or other format as appropriate. It may simply reference the artefact (e.g. source code) that is the test model.

8.2.7 Traceability

Describes traceability from the test model to associated requirements or artefacts (e.g. source code). This can be recorded in a test traceability matrix.

8.3 Test case specification

8.3.1 Overview

The test case specification identifies the test coverage items and the corresponding test cases derived by applying one or more test techniques to the test model.

A.3 provides an outline of the test case specification, while Annex I provides examples that demonstrate how test case specification can be described for two different example projects.

The contents for a test case specification are specified in 8.3.2 and 8.3.3.

8.3.2 Test coverage items

8.3.2.1 Overview

Summarizes the test coverage items identified from the test model. Test coverage items are derived by applying a test design technique to the test model.

EXAMPLE Test coverage items in the form of transitions between states can be derived from a state transition diagram by applying state transition testing.

NOTE The test coverage items can be described in lists or in tables in a document or using a tool, e.g. a database or a dedicated test tool. Test coverage items are not always formally documented, as they can be seen as a draft of test cases.

The contents for a test coverage item are defined in 8.3.2.2 to 8.3.2.5.

8.3.2.2 Unique identifier

Describes the unique identifier for the test coverage item so that it can be distinguished from all other test coverage items. An automated tool may control the generation of the identifiers, or it may be done manually according to an appropriate notation scheme. The unique identifier is not usually changed during the lifetime of the test coverage item because it is used for traceability purposes.

When the number or volatility of test coverage items is so high that the requirement for unique identifiers becomes impractical, they may be replaced by the use of other means, generally based on automated tools, to establish traceability between test cases and the test model.

8.3.2.3 Description

Describes the test coverage item; that is, the attribute that is expected to be covered by a test case according to the test design technique that was used during its derivation. It may also include additional information about the coverage item.

EXAMPLE Whether or not it is a valid or invalid equivalence partition.

8.3.2.4 Priority

Defines the priority for the testing of this particular test coverage item, if needed. The higher priority test coverage items will be executed before lower priority test coverage items.

8.3.2.5 Traceability

Describes traceability from the test coverage items to the test model. This may be recorded in a test traceability matrix.

8.3.3 Test cases

8.3.3.1 Overview

Defines the test cases derived from the test coverage items. A test case specifies how one or more test coverage items are exercised to help determine whether or not the test item has been implemented correctly.

The number of test cases derived from test coverage items will depend on the test coverage criterion defined in the test plan (e.g. if only 80 % test coverage is required not all test coverage items need to have an associated test case) and the risk level associated with the test coverage item (e.g. more test cases are normally generated for higher risk test coverage items).

NOTE The test cases can be described in lists or in tables in a document or using a tool, e.g. a database or a dedicated test tool.

The information for a test case is specified in 8.3.3.2 to 8.3.3.8.

8.3.3.2 Unique identifier

Describes the unique identifier for the test case so that it can be distinguished from all other test cases. An automated tool may control the generation of the identifiers or it may be done manually according to an appropriate notation scheme. The unique identifier is not usually changed during the lifetime of the test case, because it is used for traceability purposes.

8.3.3.3 Objective

Identifies and briefly describes the special focus or objective of the test case. This is typically in the form of a title.

8.3.3.4 Priority

Defines the priority for the testing of this particular test case, if needed. The higher priority test cases will be run before the lower priority test cases, where possible.

8.3.3.5 Traceability

Describes traceability to the test coverage items that the test case exercises. This may be recorded in a test traceability matrix.

8.3.3.6 Preconditions

Describes the required state of the test environment and test data and any special constraints pertaining to the execution of the test case.

EXAMPLE The required state of the test item before execution can start, including existence of specific test data and the currently active form or screen.

This can be described explicitly, or it can include references to other test cases, whose execution will set the preconditions.

The environment needed may be described collectively for one or more test cases, or it may not be described in each individual test case specification if the description in the test plan is sufficient.

8.3.3.7 Inputs

Specifies each action required to bring the test item into a state where the expected result can be compared to the actual results. The level of detail provided by the descriptions should be tailored to fit the knowledge of the test executors.

NOTE This can require provision of input data and/or events, e.g. button clicks, to the test item. Some of the input data can be specified by value, while others may be specified by name. Constant tables, transaction files, databases, files, terminal messages, memory resident areas, and values passed by the operating system can be considered.

Specifies all required relationships between input events.

EXAMPLE A relationship can be timing.

The actions may be numbered within the test case, if needed.

8.3.3.8 Expected results

Specifies the expected outputs and behaviour required of the test item in response to the inputs that are given to the test item when it is in its precondition state. Provides the expected values (with tolerances where appropriate) for each required output.

EXAMPLE Behaviour required of the test item can be response time.

The actions required to compare the expected results to the actual results may also be specified. For instance, examining the output in a field on a form that is not active when the input is provided, waiting for a batch job to run and a report to be printed out and examined, or closing down the test item and restarting it to examine stored data.

8.4 Test procedure specification

8.4.1 Overview

The test procedure specification describes the test cases in execution order, along with any associated actions that may be required to set up necessary preconditions and any post-execution wrap up activities.

The test procedures may be described in lists, in tables, in a document or stored by a tool, such as a database or a dedicated test case management tool.

NOTE The test procedures can be described in lists or in tables in a document or using a tool, e.g. a database or a dedicated test tool.

A.3 provides an outline of the test procedure specification, while Annex J provides examples that demonstrate how test procedure specifications can be described for two different example projects.

The contents of the test procedure specification are specified in 8.4.2 to 8.4.8.

8.4.2 Unique identifier

Describes the unique identifier for the test procedure so that it can be distinguished from all other test procedures. An automated tool may control the generation of the identifiers or it may be done manually according to an appropriate notation scheme. The unique identifier is not usually changed during the lifetime of the test procedure, because it is used for traceability purposes.

8.4.3 Objective

Identifies and briefly describes the focus or objective of the test procedure.

8.4.4 Priority

Defines the priority for this particular test procedure, if needed.

8.4.5 Start up

Describes the necessary actions to prepare for execution of the test cases specified in the test procedure. This will typically be the actions to set up the preconditions for the first test case to be executed.

8.4.6 Ordered test cases

Lists the test cases in the order in which they are to be executed. The test cases may be numbered sequentially within the test procedure. The degree to which the procedure can be varied may be defined.

This list may be a reference to the test cases, or it may be a copied list of the test cases.

If the execution of one or more test cases in the procedure does not set the preconditions for the following test case, actions to set the preconditions may be added between test cases.

The test procedure may include placeholders to record actual results and test results. Alternatively, actual results and test results may be recorded in the actual results and test result (see 8.9).

8.4.7 Relationship to other procedures

Describes dependencies this test procedure may have on any other test procedures.

Examples of dependencies on other test procedures include that they are executed before this one, concurrently with this one, or subsequent to this one.

8.4.8 Stop and wrap up

Describes the actions necessary to bring execution to an orderly halt and the actions necessary after the execution of the procedure has been completed.

EXAMPLE Actions can be termination of logging or resetting of test database.

8.5 Test data requirements

8.5.1 Overview

The test data requirements describe the test data needed to execute the test procedures defined in the test procedure specification.

NOTE The data quality model defined in ISO/IEC 25012 can support the identification of test data requirements. Data quality characteristics include accuracy, completeness, consistency, credibility, currentness, accessibility, compliance, confidentiality, efficiency, precision, traceability, understandability, availability, portability and recoverability. Data quality can be measured via data quality measures defined in ISO/IEC 25024.

A.3 provides an outline of the test data requirements documentation, while Annex K provides examples that demonstrate how test data requirements can be described for two different example projects.

The contents specified in 8.5.2 to 8.5.7 apply to each individual Test Data Requirement.

8.5.2 Unique identifier

Provides a unique identifier for each test data item so that it can be distinguished from all other test data items.

8.5.3 Description

Defines the specific name and required values or ranges of values for each test data element. It may also describe when data needs to be made anonymous or manipulated in other ways.

EXAMPLE "At least 10 customers must exist in the database with complete and correct CustomerID and all other mandatory customer information."

8.5.4 Responsibility

Specifies who is responsible for making the test data available.

8.5.5 Period needed

Identifies when and for how long the test data is needed. Test data may be needed for a single undivided period or for several separate periods.

8.5.6 Resetting needs

Specifies if the test data needs to be reset during testing.

8.5.7 Archiving or disposal

Identifies when and how test data may be archived or disposed of after completion of the testing.

8.6 Test environment requirements

8.6.1 Overview

The test environment requirements describe the properties of the test environment needed to execute the test procedures defined in the test procedure specification. If test environment requirements have been fully specified elsewhere, then separate test environment requirements documentation is not needed.

EXAMPLE The test environments requirements can be found in the organizational test practices, test plan or test specification.

In practice, the test environment might not be a perfect representation of the operational (production) environment. As such, the test environment requirements should mention the degree to which each test environment item needs to represent the operational environment.

A.3 provides an outline of the test environment requirements documentation, while Annex L provides examples that demonstrate how test environment requirements can be described for two different example projects.

The contents specified in 8.6.2 to 8.6.6 apply to each individual test environment item that is specified in the test environment requirements.

8.6.2 Unique identifier

Provides a unique identifier for each test environment item so that it can be distinguished from all other test environment items. Unique identifiers are used for traceability issues, so they should not usually be changed during the lifetime of the test environment.

8.6.3 Test environment item

Identifies each element of the test environment that is required for execution of the test procedures defined in the test procedure specification. Test environment items may be required to support set-up before the execution of a test procedure, for execution of test procedures and for any activities after test execution. If each test environment item is not a perfect reflection of the operational (production) environment, then the degree to which each test environment item represents or deviates from the operational environment should be stated.

EXAMPLE The test environment items can be divided into the following types, although other types can be needed depending on the specific environment requirements:

- hardware;
- middleware;
- software, e.g. specific systems or browsers;
- interfaces to other systems and interfacing systems;
- peripherals, e.g. printers;
- mobile devices, e.g. mobile phones, tablets;
- communication means, e.g. web access;
- tools;
- security;
- venue, e.g. size of rooms, desks/chairs and background noise level;
- accessories, e.g. special pre-printed paper forms.

8.6.4 Description

Identifies the environment item in sufficient detail for it to be implemented as required.

EXAMPLE This can include precisely named hardware or software in specific versions and specific configurations. It can also list required batch jobs that need to be run at certain points during testing to support the testing process, such as the requirement to run a standard set of batch jobs nightly, so as to reflect the production environment.

8.6.5 Responsibility

Specifies who is responsible for implementing each test environment item and for making each item available.

8.6.6 Period needed

Identifies when and for how long the environment item is needed. An environment item may be needed for a single undivided period or for several separate periods.

8.7 Test data readiness report

8.7.1 Overview

The test data readiness report describes the fulfilment of each test data requirement.

NOTE The data quality model defined in ISO/IEC 25012 can support the assessment of test data readiness.

A.3 provides an outline of the test data readiness report, while Annex M provides examples that demonstrate how test data readiness reports can be described for two different example projects.

The contents specified in 8.7.2 and 8.7.3 apply to each individual test data requirement specified in the test data readiness report:

8.7.2 Unique identifier

The unique identifier used in the test data requirements for each test data item.

8.7.3 Description of status

Describes the status of the required test data item. The status may include a description of how the actual test data deviates from the requirements, e.g. in terms of values or volume.

8.8 Test environment readiness report

8.8.1 Overview

The test environment readiness report describes the fulfilment of each test environment requirement.

A.3 provides an outline of the test environment readiness report, while Annex N provides examples that demonstrate how test environment readiness reports can be described for two different example projects.

The contents specified in 8.8.2 and 8.8.3 apply to each individual test environment requirement specified in the test environment readiness report:

8.8.2 Unique identifier

The unique identifier used in the test environment requirements for each test environment item.

8.8.3 Description of status

Describes the fulfilment of each required test environment item.

EXAMPLE This statement can include a description of how the actual test environment deviates from the requirements, e.g. in terms of versions or configuration.

NOTE This section can also be used to record the availability of certain test environment items (e.g. other applications that are integrated with the test item) as these can have an impact on testing if they are unavailable.

8.9 Actual results and test result

8.9.1 General

The actual results and the test result can be stored within a copy of the test case specification or test procedure specification (when test cases are embedded within a test procedure specification). The actual results and test result are not usually considered as independent documents. By storing actual results and test result in a copy of the test case specification or test procedure specification, the documents become a combined specification and results document.

The requirements for the actual results and test results are defined in 8.9.2 and 8.9.3.

8.9.2 Actual results

The actual results are a record of the result of the execution of a test case. The actual results are compared to the expected results to determine the test result.

Actual results are not always formally recorded. For some types of systems (e.g. regulated safety critical), there may be a requirement to fully document the actual results, and for some systems (e.g. those with high data integrity or reliability requirements) the testers may choose to fully record the actual results. The recording of actual results may be done by an automated tool during test execution.

Some test cases may include actions that provide results, which are not part of the actual results of executing the complete test case, but intermediate results. These may be recorded separately in the test log or with the actual results (with a clear distinction between the actual results and the intermediate results).

8.9.3 Test result

The test result is a record of whether or not a specific test case has passed or failed; i.e. if the actual results correspond to the expected results or if deviations were observed, or if planned execution of the test case was not possible.

EXAMPLE If the actual results and expected results match, then the test result can be a tick mark or the word "passed". Otherwise, if the test case execution fails, then the test result can be marked with a cross or with the word "failed" along with the unique identifier of the incident report that was raised as a result of the observation of the test failure. If the test case cannot be executed, then the test result can be marked as "blocked", typically along with information on what caused the test case to be blocked, if that information is available.

Sometimes this process is fully automated, with the tool comparing the actual results to expected results, and providing a report of which test cases passed, failed, or cannot be executed.

NOTE Test result is sometimes referred to as the "pass/fail" status of a test case.

8.10 Test execution log

8.10.1 Overview

Records details of the execution of one or more test procedures as a series of events. Each event may be described in a list, a table, in a document or stored in a tool, such as a database or a dedicated test case management tool.

EXAMPLE Events that can be recorded in the test execution log include:

- the date/time that the test execution session started and ended;
- the starting date/time at which each individual test was executed;
- the details of when a sudden drop in performance occurred of the computer on which a test was being executed;

- a failure, which makes further execution of a test impossible;
- a disruption to the test environment that causes the actual results to be unreliable.

A.3 provides an outline of the test execution log, while Annex Q provides examples that demonstrate how the test execution log can be described for two different example projects.

The contents of the test execution log are specified in 8.10.2 to 8.10.5, which apply to each individual event recorded in the log.

8.10.2 Unique identifier

Defines a unique identifier for each event in the test execution log.

8.10.3 Date/time

Defines the precise time and date at which each event was encountered.

8.10.4 Description

Describes what happened during each event. This may include a reference to the test procedure and test case being executed when the event was encountered, if relevant.

8.10.5 Impact

Describes the impact on test execution and the actual result, if relevant.

8.11 Test incident report

8.11.1 Overview

A test incident is any issue that is noticed during testing that requires investigation, such as when an actual result deviates from the expected result of a test or when the system behaves differently to how a tester expected. Test incidents are recorded in test incident reports. There will be one incident report for each unique incident (incident reports may also be known as defect reports, bug reports, fault reports, etc.).

The incident reports may be documented in lists or in tables in a document or using a tool, e.g. a database or a dedicated bug-tracking tool.

The format of an incident report may be defined elsewhere in an organization, for example as part of the Incident Management Processes, in which case that definition should be used.

An incident report in this context documents an incident recognized during testing.

NOTE Incidents can occur, and be reported, in other contexts, e.g., ambiguities in a business requirements specification discovered during software design, or a software failure occurring during production.

The test incident reporting template that is described in this clause only incudes those fields that are needed when the incident report is first raised. More information can be added to the incident report as it passes through the wider incident management process.

EXAMPLE Additional information that can be added to the test incident report as it moves through the broader incident management process includes the following: the cause of the incident, information on what was changed in the test item to fix the defect, the results of retesting the defect after fix, the root cause of the defect and the person that the defect is allocated to at any given time.

A.3 provides an outline of the test incident report, while Annex R provides examples that demonstrate how a test incident report can be described for two different example projects.

The contents of the test incident report are specified in 8.11.2 to 8.11.9.

8.11.2 Timing information

Records the date and possibly also the time when the incident was observed.

8.11.3 Originator

Specifies the name and title of the individual who reported the incident.

8.11.4 Context

Describes the context in which the incident was observed.

EXAMPLE This can include:

- test item the incident occurred in (i.e. name of the system or product being tested), including its version number;
- the test procedure and test case, including their unique identifiers, being executed when the incident was observed;
- any relevant information about the test environment or test data not included in other documents and considered significant by the tester;
- the test level/type in which the incident was observed.

8.11.5 Description of the incident

Provides a detailed description of the incident. Indicates if the incident is reproducible, and, if so, provides enough information to reproduce it.

Related information and observations that can help to isolate (and potentially help with correction) of the cause of the incident may be included.

The description may also reference where additional evidence or supporting information can be found to aid in diagnosis of the incident.

EXAMPLE Such evidence can include screenshots, system logs and output files.

It may also identify the existence of any known workarounds.

8.11.6 Originator’s assessment of severity

Indicates, from the originator’s point of view, the impact the incident can have from both technical and business perspectives. This may include an estimate of the time and effort to fix the associated defect.

EXAMPLE 1 Technical impact can include the user’s ability to perform tasks and operate the system, or the tester's ability to continue testing the system after the incident occurred.

EXAMPLE 2 Business impact can include consideration of how the organization's reputation can be negatively impacted by the incident, if it were to occur in the live production system.

8.11.7 Originator’s assessment of priority

Provides an evaluation of the urgency for the repair. Most organizations have from three to five categories.

EXAMPLE A categorization scheme can be that the most serious category, e.g. “Fix now”, means that the product is unusable, and the least serious, e.g. “Fix in support”, is a cosmetic incident.

8.11.8 Risk

Provides information on the introduction of new risks or changes to the status of existing risks, where applicable.

8.11.9 Status of the incident

Identifies the current status of the incident.

NOTE A common sequence for incidents as they progress through their life cycles can be: "Open", "Approved for resolution", "Assigned for resolution", "Fixed", "Retested with fix confirmed" and "Closed". Other possible status values can be "Rejected" or "Withdrawn".

Annex A
(normative)

ISO/IEC/IEEE 29119-3 requirements, recommendations and permissions

A.1 Overview

This annex explains how the requirements (shall), recommendations (should) and permissions (may) of ISO/IEC/IEEE 29119-2 activities and tasks that generate information items map to the test documentation templates defined in this document.

A.2 Common information contents

Table A.1 summarizes the requirements that apply to the common information (see Clause 5), which shall apply to all test documentation defined in this document. The position of this information in each item of test documentation can vary between organizations and within an organization.

Table A.1 — Summary of requirements that apply to all test documentation in this document

ISO/IEC/IEEE 29119-3 Clauses	Requirement (shall)
5.2.1 Unique identification of document	Shall
5.2.2 Issuing organization	Shall
5.2.3 Approval authority	Shall
5.2.4 Change history	Shall
5.2.5 Status	Shall
5.2.6 Introduction	Shall
5.2.7 Scope	Shall
5.2.8 References	Shall
5.2.9 Glossary	Shall

A.3 Test documentation contents

Table A.2 summarizes the requirements, recommendations and permissions for ISO/IEC/IEEE 29119-3. These requirements are based on the activities and tasks defined in ISO/IEC/IEEE 29119-2.

Table A.2 — Summary of requirements, recommendations and permissions for ISO/IEC/IEEE 29119-3

ISO/IEC/IEEE 29119-3 information items	Requirement (shall), recommendation (should), permission (may)
6.2 Test policy	Shall
6.2.2 Objectives of testing	Shall
6.2.3 Test process	May
6.2.4 Test organization structure	May
6.2.5 Tester training	May
6.2.6 Tester ethics	May

Table A.2 *(continued)*

ISO/IEC/IEEE 29119-3 information items	Requirement (shall), recommendation (should), permission (may)
6.2.7 Standards	May
6.2.8 Other relevant policies	May
6.2.9 Measuring the value of testing	May
6.2.10 Test asset archiving and reuse	May
6.2.11 Test process improvement	May
6.3 Organizational test practices	Shall
6.3.2 Organization-level test practice statements	Shall
6.3.2.2 Risk management approach	Shall
6.3.2.3 Test selection and prioritization	Shall
6.3.2.4 Test documentation and reporting	Should
6.3.2.5 Test automation and tools	Should
6.3.2.6 Configuration management of test work products	Should
6.3.2.7 Incident management	Should
6.3.2.8 Test levels	Should
6.3.2.9 Test types	Should
6.3.2.10 Rules/guidelines for deviation from organizational test practices	Should
6.3.3 Test level/type-specific organizational test practice statements	Should
6.3.3.1 Entry and exit criteria	Should
6.3.3.2 Test completion criteria	Should
6.3.3.3 Test documentation and reporting	Should
6.3.3.4 Degree of independence	Should
6.3.3.5 Test design techniques	Should
6.3.3.6 Test data	Should
6.3.3.7 Test environment	Should
6.3.3.8 Metrics to be collected	Should
6.3.3.9 Retesting	Should
6.3.3.10 Regression testing	Should
7.2 Test plan	Shall
7.2.2 Context of testing	Shall
7.2.2.1 Projects / test levels / test types	Shall
7.2.2.2 Test items	Shall
7.2.2.3 Test scope	Shall
7.2.2.4 Test basis	Shall
7.2.3 Assumptions and constraints	Shall
7.2.4 Stakeholders	Should
7.2.5 Testing communication	Should
7.2.6 Risk register:	Shall
7.2.6.2 Product risks	Shall
7.2.6.3 Project risks	Shall
7.2.7 Test strategy	Shall
7.2.7.2 Test levels	Shall
7.2.7.3 Test types	Shall

Table A.2 *(continued)*

ISO/IEC/IEEE 29119-3 information items	Requirement (shall), recommendation (should), permission (may)
7.2.7.4 Test deliverables	Shall
7.2.7.5 Test design techniques	Shall
7.2.7.6 Entry and exit criteria	Shall
7.2.7.7 Test completion criteria	Shall
7.2.7.8 Degree of independence	Shall
7.2.7.9 Metrics to be collected	Shall
7.2.7.10 Test data requirements	Shall
7.2.7.11 Test environment requirements	Shall
7.2.7.12 Retesting	Shall
7.2.7.13 Regression testing	Shall
7.2.7.14 Suspension and resumption criteria	Shall
7.2.7.15 Deviations from the organizational test practices	Should
7.2.8 Testing activities and estimates	Shall
7.2.9 Staffing	Shall
7.2.9.2 Roles and responsibilities	Shall
7.2.9.3 Hiring needs	Should
7.2.9.4 Training needs	Should
7.2.10 Schedule	Shall
7.3 Test status report	Shall
7.3.2 Test status	Shall
7.3.3 Reporting period	Shall
7.3.4 Progress against test plan	Shall
7.3.5 Factors blocking progress	Shall
7.3.6 Test measures	Shall
7.3.7 New and changed risks	Shall
7.3.8 Planned testing	Shall
7.4 Test completion report	Shall
7.4.2 Summary of testing performed	Shall
7.4.3 Deviations from planned testing	Shall
7.4.4 Test completion evaluation	Shall
7.4.5 Factors that blocked progress	Shall
7.4.6 Test measures	Shall
7.4.7 Residual risks	Shall
7.4.8 Test deliverables	Shall
7.4.9 Reusable test assets	Should
7.4.10 Lessons learned	Shall
8.2 Test model specification	Shall
8.2.2 Unique identifier	Shall
8.2.3 Objective	Shall
8.2.4 Priority	Shall
8.2.5 Test strategy extract	Shall
8.2.6 Test model	Shall
8.2.7 Traceability	Shall

Table A.2 *(continued)*

ISO/IEC/IEEE 29119-3 information items	Requirement (shall), recommendation (should), permission (may)
8.3 Test case specification	Shall
8.3.2 Test coverage items	Shall
8.3.2.2 Unique identifier	Shall
8.3.2.3 Description	Shall
8.3.2.4 Priority	Shall
8.3.2.5 Traceability	Shall
8.3.3 Test cases	Shall
8.3.3.2 Unique identifier	Shall
8.3.3.3 Objective	Shall
8.3.3.4 Priority	Shall
8.3.3.5 Traceability	Shall
8.3.3.6 Preconditions	Shall
8.3.3.7 Inputs	Shall
8.3.3.8 Expected results	Shall
8.4 Test procedure specification	Shall
8.4.2 Unique identifier	Shall
8.4.3 Objective	Shall
8.4.4 Priority	Shall
8.4.5 Start up	Shall
8.4.6 Ordered test cases	Shall
8.4.7 Relationship to other procedures	Shall
8.4.8 Stop and wrap up	Shall
8.5 Test data requirements	Shall
8.5.2 Unique identifier	Shall
8.5.3 Description	Shall
8.5.4 Responsibility	Shall
8.5.5 Period needed	Shall
8.5.6 Resetting needs	Shall
8.5.7 Archiving or disposal	Shall
8.6 Test environment requirements	Shall
8.6.2 Unique identifier	Shall
8.6.3 Test environment item	Shall
8.6.4 Description	Shall
8.6.5 Responsibility	Shall
8.6.6 Period needed	Shall
8.7 Test data readiness report	Shall
8.7.2 Unique identifier	Shall
8.7.3 Description of status	Shall
8.8 Test environment readiness report	Shall
8.8.2 Unique identifier	Shall
8.8.3 Description of status	Shall
8.9 Actual results and test result	Shall
8.9.2 Actual results	Shall

Table A.2 *(continued)*

ISO/IEC/IEEE 29119-3 information items	**Requirement (shall), recommendation (should), permission (may)**
8.9.3 Test result	Shall
8.10 Test execution log	Shall
8.10.2 Unique identifier	Shall
8.10.3 Date/time	Shall
8.10.4 Description	Shall
8.10.5 Impact	Shall
8.11 Incident report	Shall
8.11.2 Timing information	Shall
8.11.3 Originator	Shall
8.11.4 Context	Shall
8.11.5 Description of the incident	Shall
8.11.6 Originator's assessment of severity	Shall
8.11.7 Originator's assessment of priority	Shall
8.11.8 Risk	Shall
8.11.9 Status of the incident	Shall

Annex B
(informative)

Overview of examples

B.1 Overview

Annexes C to R contain examples of the application of each information item (template) for agile and traditional projects, to demonstrate the applicability of this standard to both types of projects. These are examples only, and many variations are possible and likely.

The example test documentation is based on two fictional organizations:

— **Agile Corporation** is a large publication organization producing magazines and books. Agile Corporation has an internal IT department, which is responsible for developing and supporting the organization’s IT products. Products are built by a single agile team and all development is conducted under an agile lifecycle. The organization has several years of experience working in this way and finds that it works well with their needs for new and enhanced IT systems to support their business. The organization has chosen to adopt an international standard for their testing, to demonstrate to their customers that they care about quality.

The product featured in this example is the development of a new web-based magazine subscription system that allows customers to become subscribers and allows existing subscribers to change their personal information and order new or extended subscriptions.

— **Traditional Ltd** is a small company that produces advanced analysis equipment for the farming industry. Some of their products are safety-critical, in the sense that incorrect analysis results risk causing the prescription of incorrect fertilizer doses (either too much or too little). The organization is hence required to produce the product according to an international standard that demonstrates and retains evidence of correct behaviour during testing, along with solid test planning, traceability and reporting.

The example project features the development of the PC-based portion of a product called UV/TRT-14 33a. It is an apparatus to measure fertilizer components and their concentration in earth samples. The apparatus has a user interface working on a PC with wireless connection to the measuring system.

The traditional examples are detailed, as they are designed to achieve full conformance (see 4.1.2). The agile examples are lightweight (i.e. minimal and lean), as they are designed to achieve tailored conformance (see 4.1.3), with some information communicated verbally. Agile Corporation’s adoption is an acceptable form of tailoring, due to the lower perceived development, testing, quality and contractual risks involved with their product. The more 'heavyweight' documentation prepared by Traditional Ltd relates to a much higher risk system that requires far greater quality assurance and testing across the lifecycle. Any organization can adopt the level of test documentation that suits their teams and systems, from full conformance (i.e. production of all documents and inclusion of all fields) to tailored conformance with a minimal set of test documentation, with agreement from relevant stakeholders.

Not all examples in these annexes include all of the common information (see Clause 5), as the examples focus specifically on the testing-related contents of each information item.

The examples might not be internally consistent; each section is to be regarded as an independent example of the information related to the topic (heading).

The examples are not always complete. Where paragraphs or omitted text have been left out, this is marked by ellipsis, like this "...".

The word “shall” appears in some of the examples. These “shall”s are example wordings only and are not normative.

The examples do not follow the editorial rules in the ISO/IEC Directives, Part 2. Explanations on the examples are provided in NOTEs.

Annex C
(informative)

Test policy

C.1 Example 1 – Agile Corporation

Test Policy – Agile Corporation

Objectives of testing: To provide enough information to determine the quality and release readiness of each product under test.

Test process: The test process is based on ISO/IEC/IEEE 29119-2 and is aligned with the agile development methodology. All agile delivery teams are required to utilise automated Test-Driven Development (TDD) for unit testing and Acceptance Test Driven Development (ATTD) for system and regression testing.

Test organization: Testers are allocated to agile development (delivery) teams from a central pool, with each tester reporting to their agile Delivery Lead. Central experts (working under the Head of Testing) are also available to provide test consultancy services to each product as required.

Training: All testers are required to have an appropriate university degree or a minimum foundation-level industry certification in software testing. Testers are expected to be knowledgeable in agile concepts or to become so within three months of joining an agile team.

Ethics: All testers will adhere to the Tester's Code of Ethics

Standards: Agile Corporation utilises tailored versions of ISO/IEC/IEEE 29119-2 and ISO/IEC/IEEE 29119-3, with all tailoring decisions design to suit the organization's agile practices and all requiring approval by relevant the Head of Testing, Delivery Leads and Product Owners.

Related relevant policies: Software Development Policy for Agile Corporation, version 4.3 (12/2/2019).

Test process improvement & value determination: Retrospectives at the end of each iteration will capture lessons learned and improvement concepts and will reflect on the value of testing practices. Improvement actions will be actioned during future iterations. Innovations are shared with other agile teams for testing maturity uplift across the organisation.

Test asset archiving & reuse: all session sheets, documented tests & test automation frameworks will be retained according to the Release Test Plan for each team.

C.2 Example 2 – Traditional Ltd

Test Policy – Traditional Ltd

Objectives of Testing: Testing is conducted as a means for improving user and customer confidence in our products and determining when systems are ready for release. Testing is one of many means to achieve this goal.

Test Process: All testing is conducted according to ISO/IEC/IEEE 29119-2. The test process includes all test activities in the standard, namely: organizational test practices, test planning, analysis and design of test cases and procedures, specification of test environment and test data requirements, test execution and recording of the test including registration of any incidents, and test completion and reporting. Static testing is also conducted on all projects according to ISO/IEC 20246. All software development projects must include a test project, with both projects starting at the same time.

Test Organization Structure: Each project will be allocated with testers that have specialist capabilities as required from a central pool. Each tester will report to their project manager and also independently to the Head of Testing as required (e.g. for escalation of testing-related risks or issues).

Tester Training: All testers are expected to have achieved a minimum university-level degree in computer science, computer technology, software engineering or similar course. All testers must also complete testing certifications that match their staffing level, which includes foundation level training for all staff, advanced test design for test analyst / senior test analysts and test management training for test leads and above.

Tester Ethics: All testers will follow the organizational code of ethics.

Standards: The organization follows ISO/IEC/IEEE 29119-2, ISO/IEC/IEEE 29119-3, ISO/IEC/IEEE 29119-4 and ISO/IEC 20246 (for Static Testing). The organization also follows the ISO/IEC/IEEE 12207 product development lifecycle.

Other Relevant Policies: All software development is conducted according to the Traditional Ltd Software Development Life Cycle, which is based on the above standards and can be found on the company intranet.

Measuring the Value of Testing: All projects are required to collect metrics that can be used to calculate the return-on-investment (ROI) for testing practices and new innovations. This can include calculating ROI in terms of reduced testing cost and effort and reduced time-to-market.

Test Asset Archiving & Reuse: All projects will store and maintain manual regression test suites in the centralised test management tool. All automated test scripts will be stored in the central code/version control tool. All manual and automated test scripts will be handed over to the business-as-usual (BAU) team at the completion of each project, after the warranty period is complete (using the company's standard knowledge retention and transfer process during handover).

Test Process Improvement: After each iterative release of a system to production, members of each project team, including all testers, will conduct a review of the testing and quality assurance practices conducted throughout the release, to identify improvements that can be implemented throughout the next release cycle. An annual test practice assessment will also be conducted, to determine whether existing testing practices are suitable and to identify new improvements that can be implemented within each project and as broader organizational testing improvement initiatives.

Issued by Miguel Sanchez (Head of Testing) and Ursula Myers (Head of Development)

Approved by Sarah Blacksmith (Chief Technology Officer)

Annex D
(informative)

Organizational test practices

D.1 Example 1 – Agile Corporation

Organizational Test Practices – Agile Corporation

Scope: The Organizational Test Practices are applicable to all testing conducted at Agile Corporation.

Risk management: All agile teams must conduct a product & project risk assessment based on the Testing-Related Risk Management Process (TRM56) defined on the team wiki. Risk assessments are conducted during iteration zero and revisited during subsequent iteration planning meetings.

Test selection & prioritization: All test selection and prioritization will be based on the iteration risk assessment, which will be carried out during each iteration planning session.

Test automation, tools, defect/incident management & test documentation: Tests are automated whenever possible via Agile Corporation's central automation framework, with scripts stored in the central source control system. Exploratory test sessions are captured via the approved recording tool, with execution evidence stored in the central test management tool. Manual tests (when required) are also stored in the central test management tool. All manual and automated tests must be traced to user stories. Defects are stored in the task/requirement management system, either as defects linked to user stories, or as comments on user stories, using the standard defect and story management practices and standard templates that have been configured into the tools. All agile teams must prepare a one-page Release Test Plan, Iteration Test Plans (whiteboard-based), Test Cases for high risk features, and an end of release one-page Test Summary Report.

Reporting: Test progress is reported verbally at morning stand-up. Due to client requirements, a point-in-time capture of test status is captured weekly on the team wiki, indicating numbers of tests passing, failing and blocked per sprint. In teams that are recording defects on separate defect cards, defect counts per severity level are also reported weekly on the wiki. Test and defect status are available real-time on the team's dashboards. Due to client requirements, all teams will be required to produce one-page Test Summary Reports.

Configuration management of test assets: The organization's requirement management, test management and test automation tools have in-built configuration management for all assets stored within each tool. All document-based assets will be stored on the team wiki, with manual version control applied to ensure each unique document version is captured and backed up.

Test levels, types & techniques: Types of testing that prove user stories meet their acceptance criteria and that reduce risks will be chosen during iteration zero and subsequent iteration planning sessions. Levels of testing typically applied include unit, integration, story/system, system integration, acceptance and production verification testing. Types of testing typically applied are functional, performance, accessibility, penetration and disaster/recovery testing. Entry criteria, exit criteria and test completion criteria are specified in the Definition of Done. Test techniques from ISO/IEC/IEEE 29119-4 are tailored to ensure testing remains lean.

Guidelines for deviation from Organizational Test Practices: Each team is required to use the same templates for requirement/story management, defect management and test management within the organization's suite of tools. Test documentation templates (i.e. for written documents) can be tailored as required by each delivery team.

Degree of independence: Each Agile team is assigned a tester that reports to the Delivery Lead. However, testers also have the ability to report any risks or issues to the Head of Testing (HoT), providing independence to all testing within the organization, as the HoT works in parallel to the Head of Development and reports directly to the Chief Technology Officer. The test organization is technically, managerially and financially independent from the development organization of Agile Corp, while within each agile team, assigned testers participate directly in self-organizing teams that report to their Delivery Lead.

Test environment & data: Developers conduct all unit and integration testing in the development environment. All testers conduct story/system and system integration testing in the system test environment. Users conduct acceptance testing in the pre-production environment. Performance, accessibility, penetration and disaster/recovery testing is also conducted in the pre-production environment. Production verification and penetration testing are conducted in the production environment. Obfuscated production data will be utilized during testing, ensuring all uniquely identifiable private data is removed or masked prior to testing.

Metrics:

— Test progress is measured according to team velocity, which is assessed at the end of each iteration

— Test automation coverage is measured by the percentage of acceptance criteria (from user stories) covered by automation tests, and is reported at the end of each iteration

From a product quality improvement perspective, the number of defects per severity level that slip through to production is assessed monthly, to identify opportunities to improve iteration and release testing processes.

Retesting and regression testing: Tests are automated wherever possible, reducing regression testing overheads. For all detected defects, an automated or manual regression test must be created.

D.2 Example 2 – Traditional Ltd

Organizational Test Practices – Traditional Ltd

Item	Practices
Risk management approach	All projects must conduct a product & project risk assessment based on the organizational Testing-Related Risk Management Process (TRM2019_04), which meets the requirements of ISO 31000 and utilizes quality criteria from ISO/IEC 25010 and ISO/IEC 25012. Risk assessments must be conducted at the start of each project, ideally while requirements are still being written, and revisited regularly (e.g. weekly) throughout each project. Any deviations from the specified risk management process must be approved by the Head of Testing.
Test selection and prioritization	Test cases and test procedures will be prioritized according to the risk associated with the requirements the cases are covering. If a test procedure includes test cases with different risk levels, the test case with the highest level determines the risk level for the entire procedure. Execution of test procedures must always be scheduled according to the risk, so that the higher the risk level the sooner the procedure is scheduled to be executed. Care must however be taken so that all feature sets are covered by some testing, that is no feature set must be left out of the execution schedule.
Test documentation and reporting	The test projects must be documented in such a way that an audit can establish what has been planned and what has been performed. Tracing between artefacts is essential. Test documentation requirements specified in ISO/IEC/IEEE 29119-3 must be produced throughout all projects.
Test automation and tools	The organization's central test management tool is to be used on all test projects and for all test levels and types. In the cases where more than 4 cycles of regression testing are planned, the project might consider automating testing.

Item	Practices
Configuration management of test work products	All test documentation must be stored in the central knowledge repository for each project (which is backed up and version controlled). All manual and automated test cases and execution results must be stored in the central test management tool. All automated test scripts, including for functional regression testing and performance testing, must be stored in the central source code control system.
Incident management	The company's Incident Management Process (IM2019_01) must be followed on all projects. This includes the mandatory requirement for documenting all defects or issues detected in each system.
Test levels	All projects must include the following test levels: — Static Testing: Requirement, Design & Code Inspections — Unit Testing — Integration Testing — System Testing — System Integration Testing — User Acceptance Testing — Production Verification Testing
Test types	All projects must consider applying the following types of testing, depending on the requirements and risks involved in each project: — Performance Testing — Disaster/Recovery Testing — Penetration Testing All other types of non-functional testing must be considered based on the unique requirements and risks involved in each project.
Guidelines for deviation from Organizational Test Practices	Deviations from the Organizational Test Practices will only be permitted with the express written approval of the Head of Testing and the Test Manager of the given project or program of work.

Item	Practices
Entry criteria	Entry Criteria for test execution: — Static Testing: Requirement, Design & Code Inspections — Artefact being inspected has been drafted — Unit & Integration Testing — Design & code have been implemented for a given feature — System Testing — Unit & Integration Testing are complete for the given feature — Test cases are designed, traced to requirements & signed off by stakeholders — System Integration Testing — Interface has been specified including functions & parameters — Internal or external system interface is available (either as real interface or stub) — Test cases are designed, traced to requirements & signed off by stakeholders — User Acceptance Testing — All previous stages of testing are complete and passed testing — Test cases are designed, traced to requirements & signed off by stakeholders — Production Verification Testing — All previous stages of testing are complete and passed testing — Test cases are designed, traced to requirements & signed off by stakeholders — Performance, Disaster/Recovery & Penetration Testing — Requirements for each type of testing have been specified and signed off by stakeholders — Test cases are designed, traced to requirements & signed off by stakeholders — Production-like environment is available and ready for testing — Required test data has been generated and is available For all test levels, it is recommended that a Project Test Plan (at a minimum) be signed off by the Project Manager, Test Manager and Head of Testing, prior to commencing each type of testing.

Item	Practices
Exit criteria	Exit Criteria for test execution: — Static Testing: Requirement, Design & Code Inspections — All — All Test Levels — All in scope test cases have either been executed and passed testing, or any remaining tests that have not yet passed testing have been signed off by stakeholders — All severity 1 and 2 defects have been resolved and have passed retesting, and/or any remaining defects have been signed off by stakeholders
Test completion criteria	All projects must aim to achieve 100 % requirements coverage, with all remaining test cases being signed off by stakeholders. All severity 1 and 2 defects are resolved and have passed retesting and/or all outstanding defects have been signed off by stakeholders, with a plan & schedule in place and agreed to address all outstanding defects.
Test documentation & reporting	ISO/IEC/IEEE 29119-3 requirements must be met for all types of dynamic testing. ISO/IEC 20246 requirements must be met for static testing.
Degree of independence	All testers will report to their assigned Project Manager, and also to the Head of Testing to ensure independence throughout testing.
Test design techniques	Specification-Based and Structure-Based testing techniques defined in ISO/IEC/IEEE 29119-4 must be implemented as required to cover requirements and to mitigate risks. Any tailoring of the techniques defined in the standard must be signed off by the Head of Testing.
Test data	Test data will be extracted from production but must be obfuscated (i.e. anonymized) to ensure customer privacy is maintained throughout development and testing.
Test environment	The test environment must be as representative as possible to the production environment in terms of hardware and software. In the case of embedded systems, the system test can be executed on a simulator.
Metrics to be collected	The following metrics shall be collected and reported throughout testing: — Planned vs. actual number of test cases & test procedures designed and executed, per day — Defect leakage – percentage of defects found in a test level that escaped from a previous test level — Defects density in production – the number of defects escaped from production normalized by project size (e.g. 0.4 defects per function point) — Defect age — Return on investment for automated testing (e.g. cost of manual regression testing vs. cost of automated regression testing)
Retesting	All test cases and procedures that detect defects must be rerun after defect correction to confirm that defects are fixed and have passed testing.
Regression testing	Regression testing during the system test level is at the Test Manager's discretion, based on the risks present at that time. In the final execution cycle of System Testing and User Acceptance Testing, all test cases and procedures must be executed.

Annex E
(informative)

Test plan

E.1 Example 1 – Agile Corporation

NOTE In this example, this test plan would be available on the agile team wiki, with the latest version posted in the top right corner of the story board in the development room.

Test Plan for: New Subscription System (NSS) **Iteration:** 3

Context, test items & scope: NSS Iteration 3, including all test types and levels of testing for all stories in play for the given sprint. All stories are captured in the team's requirement and task management tool.

Team & Communication: Each iteration is carried out by the agile team consisting of a Delivery Lead, Developers, Business Analyst, User Representatives, Testers and a Product Owner. The agile team reports to the Delivery Lead, while all Developers also report independently to the Head of Product Development (Stephanie McFarlan) and Testers to the Head of Testing (Shane Dolan)

Stakeholders: The Product Owner and Delivery Lead are stakeholders for approval purposes. Escalation to the Head of Product Development & Head of Testing is only required when consensus cannot be achieved within the agile team.

Risks: The risks for this iteration are listed on relevant story cards. A current general risk is that the agile team does not have access to live data in the supporting databases.

Test strategy:

- At release and iteration planning, estimate testing and development effort based on average velocity, returning to the backlog any user stories that cannot be achieved (done) by the end of the iteration, including any accumulated technical debt (defects) that cannot be resolved in the current iteration.
- Create automated unit & integration tests based on stories before coding starts (i.e. TDD). Test new code with system tests, followed by system integration tests with external systems and finally with user acceptance testing (using ATTD) before marking stories as Done. Ensure customers are involved in and ideally conduct user acceptance testing. Automate as many tests as possible.
- Regression test any feature that has changed since previous iterations and the entire system prior to the showcase meeting, via the automated test suite.
- Use test design techniques that best suit acceptance criteria, keeping in mind that higher risk stories require more thorough testing that lower risk stories.
- For low risk stories, use minimized combination coverage, for high risk stories use maximized combination coverage.
- Ensure that no defects of severity 1 or 2 remain outstanding when a story is marked as Done. All remaining defects must be converted to new user stories and placed in the backlog for future iterations.
- Outline test progress at daily stand up meetings including any blockers (e.g. environment issues) that are preventing testing from progressing as planned.
- Store all session sheets and tests in the central test management tool and all automated test scripts in the central code repository, so they are available for retesting and regression testing.

- Issue a short email-based summary reporting on the outcomes of testing at the end of each iteration.
- Entry, exit and completion criteria for testing are built into the Definition of Done.

All other strategy details are as per the Organizational Test Practices & Agile Team Charter.

E.2 Example 2 – Traditional Ltd

NOTE This annex provides an example of a project test plan.

Project Test Plan

Traditional Ltd

Project # UV/TRT-14 33a

Document approved, with approval provided by project board

Revision log

Date	Version	Changes	Init.
10.11.2020	0.1	Initial draft released for review	AMH
16.11.2020	0.2	Updated following internal test team feedback	TPR
17.11.2020	0.3	Updated following stakeholder feedback	TPR
20.11.2020	1.0	Approved by stakeholders	AMH
30.11.2020	1.1	Updating following environment design change	TPR

1 Overview

1-1 Introduction

This document describes the plan and strategy for testing on the UV/TRT-14 33a project, specifically the PC-based aspect of the system.

1-2 Scope

This document is written at the project level, covering all levels and types of testing in scope for the project.

1-3 References

- Project Plan (PP) for the UV/TRT-14 33a project
- Business Requirements Specification (BRS) for the UV/TRT-14 33a project
- Software Requirements Specification for the UV/TRT-14 33a project
- Enterprise Architecture Specification (EAS) for the UV/TRT-14 33a project
- Organizational Test Practices (OTP) for Traditional Ltd

1-4 Glossary

The glossary of terms for all documentation related to this project is provided in the team wiki.

2 Overview / Context of Testing

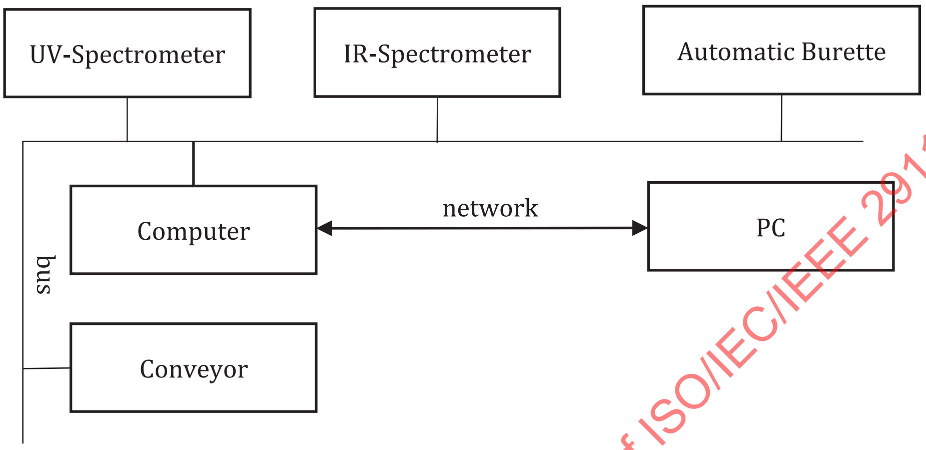
2-1 Projects, Test Types and Test Levels

The UV/TRT product consists of the following hardware modules:

- UV Spectrometer

- IR Spectrometer
- Automatic Burette
- Conveyor
- Computer (server)
- PC / Laptop

The architecture is summarised in the following figure:



The system consists of the following software modules on the computer (the server):

- UV module
- IR module
- Burette module
- Conveyor module
- Network module

The system consists of the following software modules on the PC:

- Calibration module
- Compound identification and concentration module
- Setup module
- Control and report module
- Network module

This test plan is applicable to the types and levels of testing listed in section 5-1.

2-2 Test Items

The test for this project includes testing of:

- Each module of the PC software, as listed in 2-1 above
- Each component of the PC software modules listed in 2-1 above
- The functionality of the complete software system

The version of each test item will be obtained and recorded from the configuration management system at the time of executing each test.

2-3 Test Scope

The PC system consists of software modules listed in section 2-1. The network module is a commercial-off-the-shelf product, proven in many other organizations, and does not require standalone testing. All the other modules shall be tested under the assumption that the PC's operating system and network are working correctly. Functionality related to the network connection will not be tested, except indirectly when these features are utilised in connection with other tests. Non-functional quality factors such as performance, security and disaster/recovery will be outsourced to another company that will perform this type of testing independently. A separate Test Plan will be created for each of these types of testing by the outsourcing company.

2-4 Test Basis

The requirements for the system are captured in the following documents:

- Business Requirements Specification (BRS) for the UV/TRT-14 33a project
- Software Requirements Specification for the UV/TRT-14 33a project
- Enterprise Architecture Specification (EAS) for the UV/TRT-14 33a project

2-5 Assumptions and Constraints

AS-1: it is assumed that all team members will be co-located for efficiency and effectiveness.

2-6 Stakeholders

Please refer to the stakeholder list in the PP.

3 Testing Communication

Please refer to PP for a list of teams and individuals involved in test and project communication.

4 Risk Register

The following abbreviations are used in the risk tables:

- P = probability or likelihood of the risk
- C = Consequence, impact or effect if risk materializes
- E = Exposure = Probability x Consequence

The scales for both probability and consequence will be 1 to 6, where 6 is the highest.

4-1 Product Risks

Risk ID & Name	P	C	E	Mitigation Activities
1 Risk that calibration is incorrect	2	5	10	Design and code inspection Thorough unit, system & acceptance testing Automated testing
2 Risk that compound identification is incorrect	2	6	12	Design and code inspection Thorough unit, system & acceptance testing
3 Risk that concentration calculation is incorrect	1	6	6	Design and code inspection Thorough unit, system & acceptance testing
4 Risk that calculations ‘drift’ if the machine is not turned off regularly	3	5	15	Code inspection Dynamic analysis to identify memory leaks Endurance / soak test

Risk ID & Name	P	C	E	Mitigation Activities
5 Risk that compound identification is too slow	3	1	3	Performance test under various conditions
6 Risk that reports are not accurate	2	5	10	Design and code inspection Thorough unit, system & acceptance test
7 Risk that system crashes in middle of analysis, giving unreliable results	2	2	4	Disaster/recovery testing that forces system to crash during analysis
8 Risk that user manuals are not understandable by lab technicians	4	3	12	Usability assessment & inspection of user manuals
9 Risk that setup is difficult to understand	4	2	8	Usability assessment of prototype's user interface Review of requirements & design specifications
10 Risk of errors in text of localised reports, due to translation errors	5	3	15	Review all reports in languages known to ensure correctness

4-2 Project Risks

Risk ID & Name	P	C	E	Mitigation Activities
1 Risk that staff numbers in specific roles are insufficient	2	4	8	Ensure estimates consider staff capability requirements. Track test progress closely and report any predicted budget or schedule issues.
2 Risk that available staff lack sufficient knowledge and experience	2	3	6	Perform gap analysis of needs vs. staff availability. Prepare a training plan for individual participants. Include training time in the schedule. Find mentors, if possible. Ensure time is available for reviews by senior engineers.
3 Risk that some of the people who are available will not / cannot work together	5	2	10	Try to identify what the problem is. Arrange for arbitration if necessary / practicable. Where necessary, arrange for a Belbin analysis to increase understanding between the various types. Distribute activities so that there is less contact between the individuals in question.
4 Risk that there are too few licenses available for test execution tools	3	2	6	Prepare business case for additional licences. Distribute, plan and track test activities in detail, to reduce licence waiting times as far as possible. Report related problems as early as possible.
5 Risk that testing staff are unfamiliar with the test management tool	4	1	4	Prepare business case for funding tools training. Find someone from within the company who has experience of using the tool and share knowledge. Allow for additional time in initial test design & execution. Report related problems as early as possible.

Risk ID & Name	P	C	E	Mitigation Activities
6 Risk that vendors do not supply the expected material on time and to the expected quality	3	4	12	Ensure contractual requirements for quality and delivery are stated precisely, and if not, follow up with procurement. Provide templates to vendors wherever possible. Establish specific quality criteria for deliverables. Specify the consequences of breaches of quality and scheduled delivery. Follow progress and quality closely.
7 Risk that teams use more than 30 % of test execution time on incident reporting, causing schedule slippage	4	5	20	Review unit test coverage and adequacy. Request additional until testing if gaps are identified.

5 Test Strategy

5-1 Test Levels & Types

The test for the PC-part of the UV/TRT-14 33a product shall include the following test levels:

- Unit Testing
- Integration Testing (component level)
- System Testing
- System Integration Testing
- User Acceptance Testing
- Production Verification Testing

As stated in section 2-3, non-functional testing will be outsourced to a third-party supplier, with details to be provided in their Non-Functional Test Plan.

5-2 Test Deliverables

For each test level and type of testing, the following documentation will be produced:

- Test Plan
- Test Model Specification
- Test Cases, Test Procedures, Actual Results, Test Result, Test Log, Incident Reports and Test Status Reports in the organization's test management tool
- Test Completion Report (also referred to as a Test Summary Report)

An overall Project Test Completion Report will also be produced at the end of the project, summarising the results of testing across all levels of testing.

5-3 Test Design Techniques

This is specified for each test level in the OTP.

5-4 Entry and Exit Criteria

This is specified for each test level in the OTP.

5-5 Test Completion Criteria

As per the OTP:

- All areas of high to medium risk will undergo testing that covers 100 % of the requirements and risks with the most comprehensive number of test cases available. This includes 3-value Boundary Value Analysis and coverage/execution of all potential positive, negative and alternative tests.
- Any areas of low risk will undergo positive testing, with minimal coverage of negative tests.

5-6 Degree of Independence

Unit testing will be conducted by Developers. All other levels of testing will be carried out by independent testers.

5-7 Metrics to be Collected

This is specified for each test level in the OTP.

5-8 Test Data Requirements

Sanitized data from the production environment will be provided in the system test environment.

5-9 Test Environment Requirements

The test environment will be setup to mimic production. Specific test tool requirements are:

- Ant
- Jira
- JBoss
- XRay
- Selenium Web Driver
- SoapUI or Postman (proof-of-concept implementation to determine most suitable tool)

Additional environment details will be specified in a separate Test Environment Requirements document.

5-10 Retesting

This is specified for each test level in the OTP.

5-11 Regression Testing

This is specified for each test level in the OTP.

5-12 Suspension and Resumption Criteria

Suspension criteria are listed in the project risk register.

5-13 Deviations from Organizational Test Practices

There are no deviations planned from the organizational test practices. Any deviations deemed necessary will be communicated to the Delivery Lead and Head of Testing and will be captured in the quality risk register.

6 Testing Activities and Estimates

This information is available on the team dashboard: https://mpower.traditional.com/irj/portal

For information on cost related measures and monthly tracking, refer the following link:

https://processnet.masked.com/projectdashboard/dashboardhome_new.asp

7 Staffing

7-1 Roles, Activities and Responsibilities

Detailed roles, activities and responsibilities will be documented in Test Plans of each level of testing.

7-2 Hiring Needs

An experienced system integration tester is required – interviews will commence shortly.

7-3 Training Needs

Training in the company's test case management tool will be provided for any tester that do not have existing experience in the tool, via existing in-house training materials.

8 Schedule

The test schedule is provided in the Gantt chart for the project, on the team dashboard:

https://mpower.Traditional.com/irj/portal

Annex F
(informative)

Test status report

F.1 Example 1 – Agile Corporation

NOTE 1 A status report is shared on the team's wiki at the end of each week. This documentation is required due to contractual obligations with the client.

Iteration Test Status Report for: New Subscription System (NSS) – Team ScrumPoint

Reporting Period: Iteration 3, week 1

Progress against Test Plan:

- Testing is DONE for all 10 user stories that were in play this week
- For the one high-risk story, 92 % statement coverage was achieved, and 82 % for all others on average
- There are no outstanding severity 1 or 2 defects, though there are 4 severity 3 and 3 severity 4 defects
- Test automation is being completed in-sprint for all in-scope story, SIT and UAT tests

Factors blocking progress: Exploratory testing & test automation were delayed on Tuesday & Wednesday due to changes to the UI early on Tuesday. An additional Quality Engineer and Test Automation Engineer from team Sprinters assisted with design & execution on Thursday & Friday, allowing our team to catch up.

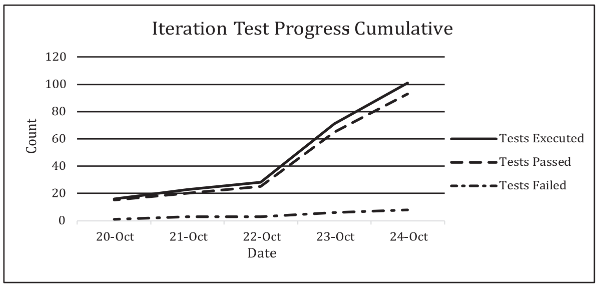
Test measures: 8 new session sheets were developed, one for each story, each containing many test ideas.

New and changed risks: Risks for stories in play this week have been mitigated. No new risks were identified.

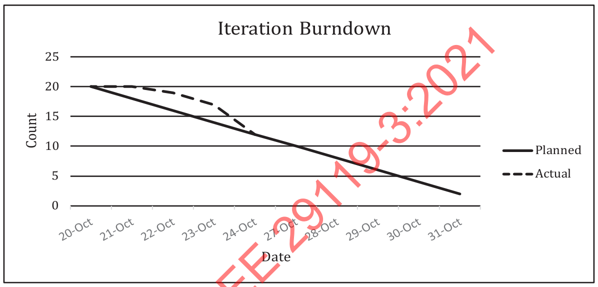
Planned testing: Story, SIT & UAT will continue next week for iteration 3.

NOTE 2 Metrics for the status report are also made available on the team's real-time dashboard.

Iteration Test Progress - Cumulative					
Date	20-Oct	21-Oct	22-Oct	23-Oct	24-Oct
Tests Executed	16	23	28	71	101
Tests Passed	15	20	25	65	93
Tests Failed	1	3	3	6	8



Iteration Burndown (Iteration 3, Week 1)					
Date	20-Oct	21-Oct	22-Oct	23-Oct	24-Oct
Planned DONE	2	2	2	2	2
Actual DONE	2	0	1	2	5



F.2 Example 2 – Traditional Ltd

Weekly Test Status Report – Project UV/TRT-14 33a

Reporting period: 27 – 31 Oct 2019 **RAG Status:** Green

Progress against test plan: System testing continued. Execution started on the Calibration module as planned. Execution of the Setup module was delayed due to delays in releasing the component to the test environment.

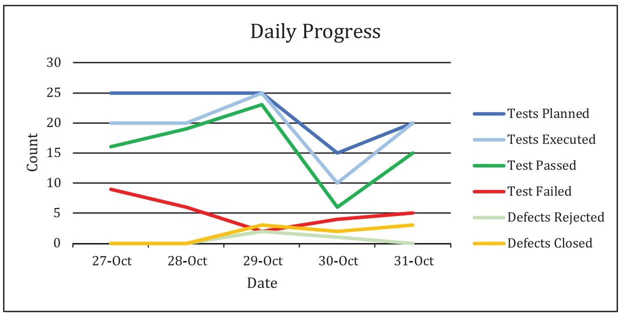
The target of executing 2/3 of test cases by October 31 was almost achieved, with only 10 tests remaining to achieve the planned milestone, with the remaining gap being due to the delayed release of the Setup module. An additional tester will be added from November 3, to allow the team to complete system testing by November 10.

32 defects are open, but almost all appear to be lower critically (severity 3 and 4). A total of 16 defects should have been detected during unit testing, rather than system testing. An increase in unit test coverage via boundary value analysis and equivalence partitioning would reduce the cost of finding and fixing defects.

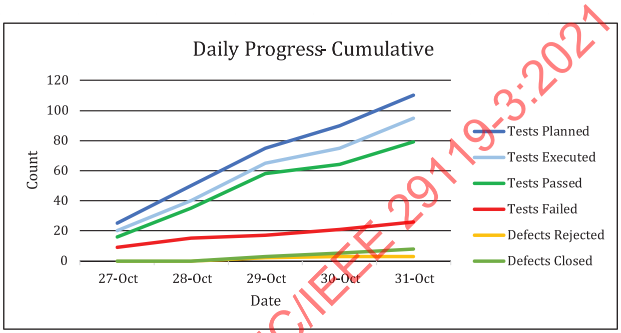
Blocking factors: Awaiting release of the Setup module, currently scheduled for November 4

Test measures:

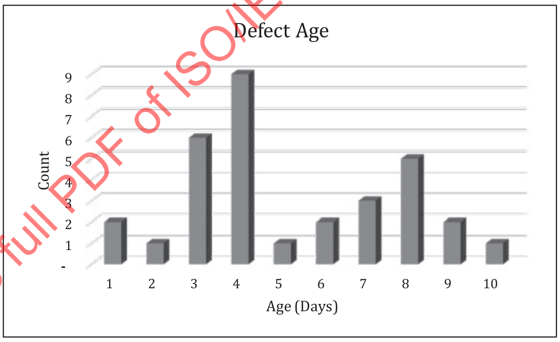
Daily Progress					
Date	27-Oct	28-Oct	29-Oct	30-Oct	31-Oct
Tests Planned	25	25	25	15	20
Tests Executed	20	20	25	10	20
Tests Passed	16	19	23	6	15
Tests Failed	9	6	2	4	5
Defects Rejected	0	0	2	1	0
Defects Closed	0	0	3	2	3



Daily Progress - Cumulative					
Date	27-Oct	28-Oct	29-Oct	30-Oct	31-Oct
Tests Planned	25	50	75	90	110
Tests Executed	20	40	65	75	95
Tests Passed	16	35	58	64	79
Tests Failed	9	15	17	21	26
Defects Rejected	0	0	2	3	3
Defects Closed	0	0	3	5	8



Defect Age (Days)	Count	Percent
1	2	6,25 %
2	1	3,13 %
3	6	18,75 %
4	9	28,13 %
5	1	3,13 %
6	2	6,25 %
7	3	9,38 %
8	5	15,63 %
9	2	6,25 %
10	1	3,13 %



Test Execution Measures – Daily

Description	#	%
Average Tests Executed Per Day	19	40

Test Execution Measures – Cumulative

Description	#	%
Total Test Cases Planned	285	100 %
Test Cases Executed to Date	190	67 %
Test Cases Passed	158	55 %
Test Cases Failed	52	18 %
Test Cases Pending (Not Yet Executed)	95	33 %

The following defect measures were also collected. P1-P6 are priorities, where P1 is highest.

Defect Summary - Daily

Description	Total	Severity 1	Severity 2	Severity 3	Severity 4
Defects Detected Today	5	0	0	4	1
Defects New	5	0	0	4	1
Defects Retested & Closed	3	0	0	1	2
Defects Re-opened	0	0	0	0	0

Defect Summary - Cumulative

Description	Total	Severity 1	Severity 2	Severity 3	Severity 4
Defects Detected	52	1	3	33	15
Defects Rejected	6	0	2	3	1
Defects Fixed	3	0	1	2	0
Defects Retested & Closed	20	0	1	10	9
Defects Open	32	1	3	16	12

New and changed risks: There is a risk that the planned end date of testing will not be met, due to the delay in the Setup module being released to the test environment. An additional tester has been added to reduce this risk. This risk is captured in detail in the project's quality risk register.

Planned testing: Over the next week, testers will aim to complete testing on the Calibration module and commence testing on the Setup module (assuming the module is released to the test environment by Friday afternoon).

Annex G
(informative)

Test completion report

G.1 Example 1 – Agile Corporation

NOTE This report is available on the team wiki at the end of each release.

Release Test Report for: New Subscription System (NSS) **Version:** Iteration 12

Date: March 12, 2021

Covers: Release 1, which is planned for release to production in release train cycle 4.

Risks: All open risks have been accepted by the product owner. Residual risks related to performance are not expected to impact on go-live as the product will be released to a reduced user base, until proven stable in production. Should additional performance testing be needed, it will be conducted in Release 2.

Test Results: the product owner has accepted this release of the product based on the following:

- All 16 in-scope user stories passed system testing, system integration testing and user acceptance testing
- 100 % statement coverage was achieved for 15 high-risk user stories, and 80 % for the remaining story
- All remaining defects were severity 3 and 4, with the Product Owner confirming that they have very little impact on end-users for product features that will be in use during the trial period. All defects have been closed and raised as new user stories, to be addressed in the first iteration of the next release
- All showcases during the release were accepted by the Product Owner and all users present, with no outstanding issues
- Accessibility requirements (WCAG 2.1 AA) were met for all features implemented during the release

Notes for future work from retrospective:

- The delivery team feels an additional tester new be required to speed up accessibility testing during the next release by ensuring that accessibility testing is carried out in every iteration.
- Severity 3 and 4 defects that have moved into the backlog as new user stories must be addressed in next release
- Obfuscated production data that was used during the second half of release 1 accelerated development and testing. This approach should be continued in all remaining releases.
- In-sprint test automation and exploratory testing accelerating well, but additional test design techniques should be considered, e.g. combinatorial testing, to ensure robust data coverage.

G.2 Example 2 – Traditional Ltd

Project PC-part of the UV/TRT-14 33a product

System Test Completion Report, V 1.0, 29/11/2019

Written by: Carlo Titlefsen (Test Manager)

Approved by: Benedicte Rytter (Project Manager)

Summary of testing performed:

- The test specification was produced; it included 600 test procedures.
- The test environment was established according to the plan.
- Test execution and recording were performed according to the plan.
- Based on the outcomes of testing, it is recommended that UAT commence.

Deviations from planned testing: The Requirement Specification was updated during the test to v5.6. This entailed rewriting of a few test cases, but this did not have an impact on the schedule.

Test completion evaluation: All test procedures were executed and passed testing, except 3. There are no remaining severity 1 or 2 defects, and all open severity 3 and 4 defects have been signed off by the business. The remaining 3 failed test cases will be rerun during UAT, once remaining severity 3 and 4 defects have been fixed. The requirements for these 3 failed tests are low risk.

Factors that blocked progress: Deployment was delayed on 5 occasions, preventing testing from starting on time. This equated to an increase in testing duration by 25 person days.

Test measures:

Three test procedures out of 605 were not passed due to defects. All test procedures that were run had passed at the end of the planned 3 weeks of testing.

During test execution, 83 defects were detected and 80 were closed.

Duration of test design and execution:

- 12,080 working hours were spent on the production of test cases and test procedures
- 10 working hours were spent on the establishment of the test environment
- 15,040 working hours were spent on test execution and defect reporting
- One hour was spent on the production of this report.

Residual risks: All the risks listed in the test plan have been eliminated, except the one with the lowest exposure, Risk # 19. This risk has been deemed acceptable by the Business Representative.

Test deliverables: All deliverables specified in the plan have been delivered to the common CM-system according to the procedure.

Reusable test assets: The test specification and the related test data and test environment requirements can be reused for maintenance testing, if and when this is needed.

Lessons learned: The test executors should have been given a general introduction to the system; it sometimes delayed them finding out how to perform a test case, but fortunately the test analyst was available throughout the test period.

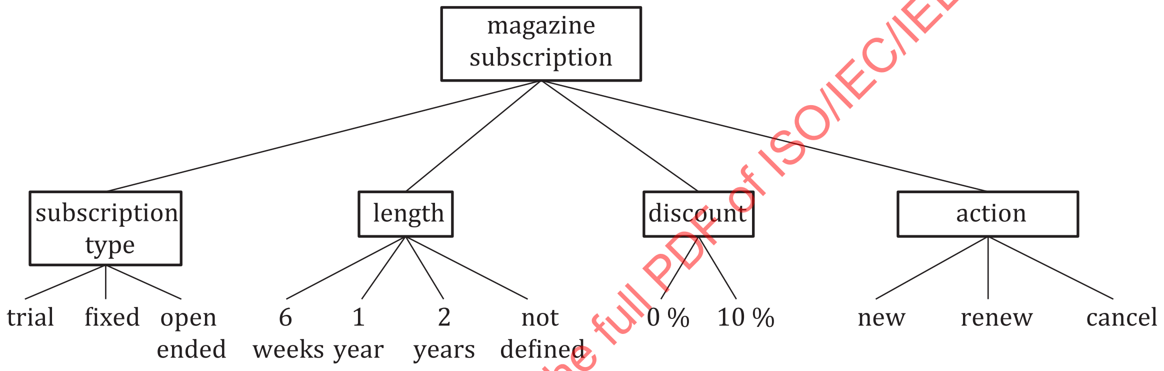
Annex H
(informative)

Test model specification

H.1 Example 1 – Agile Corporation

NOTE For Agile Corporation's new subscription system, they have created a high-level test model to show the different subscription options and actions for the users. Using this model (and keeping it up to date with each iteration) will allow the team to derive test cases for each new piece of functionality to be implemented as well as for regression testing. This test model specification is available on the product portal and the newest version is also posted in the top right corner of the story board under the test plan in the development room.

Test Model Specification



H.2 Example 2 – Traditional Ltd

Test Model Specification

Test Model: #23 – Lid Operation Model (maps to Requirement ID 12) **Priority:** 1

Objective: One of the variants of the product shall be equipped with a lid to protect the technicians performing the analyses.

Type of Testing Required (from Test Strategy): State Transition Testing (manual & automated)

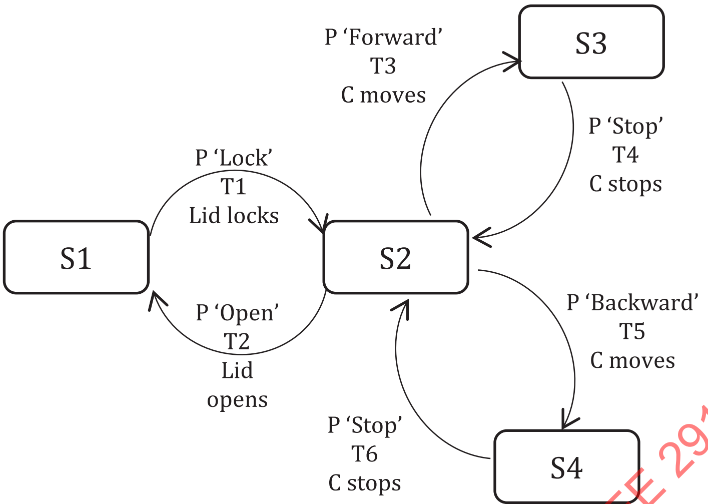
Test Model: The lid covers the carousel when it is moving. The lid has to be locked before the carousel is started and it is not possible to open it before the carousel has stopped completely. Two sensors are in place to detect if the lid is locked and if the carousel is moving.

As long as the lid is locked it is possible to start the carousel moving either forwards or backwards. To change the direction, it is necessary to stop the carousel first, but it is not necessary to open the lid.

The manoeuvring panel has the following buttons that can be pressed:

- Lock
- Open
- Forward
- Backward
- Stop

The lid operation test model can be illustrated in the following state-machine diagram, where the states and transitions are numbered, and the event "P" means "Press":



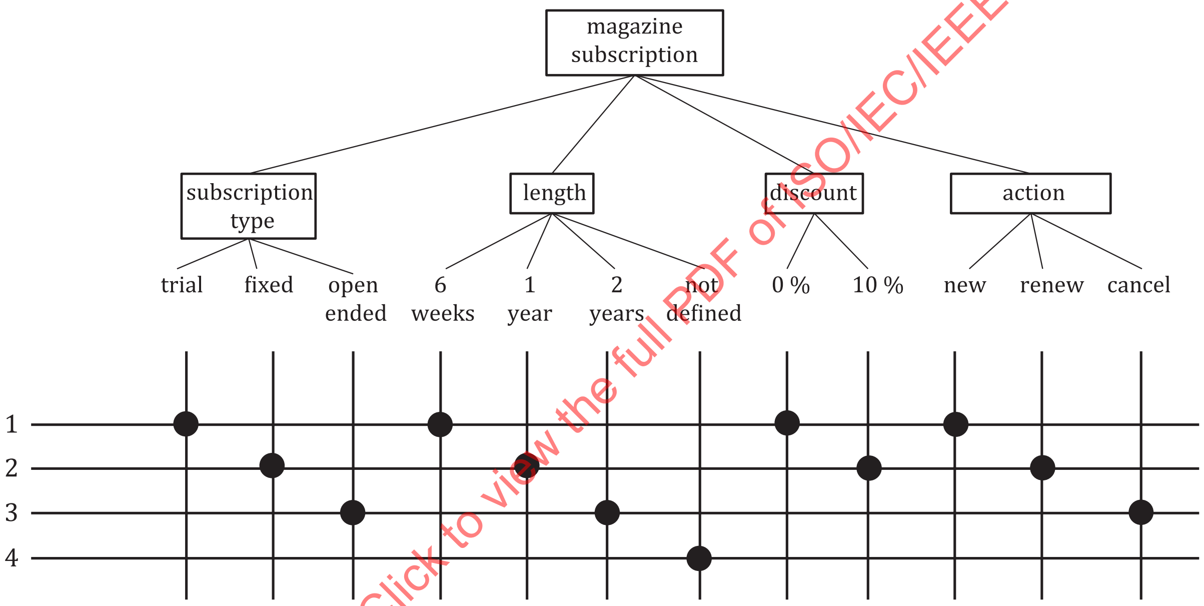
Annex I
(informative)

Test case specification

I.1 Example 1 – Agile Corporation

NOTE The test coverage items and test cases are derived from the test model specification in Annex H. The coverage method used depends on the risk of the test model items and has been outlined in the test plan. The combination table below the classification tree shows the test cases that are formed by covering (using a minimized approach) each test coverage item (marked by a black dot).

Test Case Specification



I.2 Example 2 – Traditional Ltd

Based on the test model (shown as a state transition model in Annex H – Test Model #23) the following 6 transitions can be defined as test coverage items.

	T1	T2	T3	T4	T5	T6
Starting State (TC)	S1	S2	S2	S3	S2	S4
Input	P ‘Lock’	P ‘Open’	P ‘Forward’	P ‘Stop’	P ‘Backward’	P ‘Stop’
Exp. output	Lid locked	Lid opens	Conveyor moves forward	Conveyor stops	Conveyor moves backward	Conveyor stops
End State (TC)	S2	S1	S3	S2	S4	S2

The following tables identify the null-transitions, shown in bold italic.

	Press ‘Lock’	Press ‘Forward’	Press ‘Stop’	Press ‘Backward’	Press ‘Open’
S1	S2 / Lid locked	***S1 / Null***	***S1 / Null***	***S1 / Null***	***S1 / Null***

	Press ‘Lock’	Press ‘Forward’	Press ‘Stop’	Press ‘Backward’	Press ‘Open’
S2	***S2 / Null***	S3 / Conveyor moves forward	***S2 / Null***	S4 / Conveyor moves backward	S1 / Lid opens
S3	***S3 / Null***	***S3 / Null***	S2 / Conveyor stops	***S3 / Null***	***S3 / Null***
S4	***S4 / Null***	***S4 / Null***	S2 / Conveyor stops	***S4 / Null***	***S4 / Null***

There are 6 test cases required to cover the valid transitions, and 14 to cover the null transitions, leading to a total of 20 test cases to be performed, as follows.

Test Case ID: Lid–1	**Purpose:** to test the locking of the lid when the conveyor is not moving
Priority: 1	
Test Coverage Item: T1	
Preconditions:	The apparatus must be in a stopped state with the lid open.
Inputs:	Press the lock button.
Expected results:	The lid is locked and the conveyor is still stopped.

Test Case ID: Lid–2	**Purpose:** to test the opening of the lid when the conveyor is not moving
Priority: 1	
Test Coverage Item: T2	
Preconditions:	The apparatus must be in a stopped state with the lid locked.
Inputs:	Press the open button.
Expected results:	The lid is opened and the conveyor is still stopped.

…

Annex J
(informative)

Test procedure specification

J.1 Example 1 – Agile Corporation

The following was agreed with Agile Corp's customer and team for creating new subscriptions. The team also has the option of producing documentation online with the same information. It is designed to be lightweight, covering the test case, test procedure, actual results and test result and is independent of all other session sheets.

Session Sheet ID:	9	Priority:	1
Tester Name	Michael Tester	Backlog Item #	BI_201
Sprint #	16	Date	11/12/2020
Session Duration	1 hour, 30 minutes	Build #	15.5.8
Charter	Test create magazine subscription process from perspective (persona) of new buyer / subscriber. All fields & objects on screen are in scope and should be fully operational.		
Start-up steps:	Log in as a new buyer/subscriber		
Test Ideas & Execution Results	1) Create new subscriptions with combinations of: i) All subscription types – PASSED ii) All subscription lengths – PASSED iii) No discount, full discount – FAILED (see Defects row) iv) Multiple subscriptions with combination of each type, length and discount included – PASSED 2) Start creating subscription and then cancel out of process – PASSED 3) Enter variety of positive & negative boundary values & equivalence class values in all input fields – PASSED 4) Click buttons repeatedly to check performance/reliability – NOT YET EXECUTED – deferred to next exploratory test session 5) Inspect look & feel against company design guidelines – NOT YET EXECUTED – deferred to next exploratory test session		
Defects	Defect details recorded as comments on Backlog Item BI_279		

J.2 Example 2 – Traditional Ltd

Test procedure specification

Test Proc. ID	Objective and Priority	Est. Duration
TP-3	The purpose of this test procedure is to test the way the system handles the defined measuring ranges for NCS. Priority: 2	20 minutes

<table>
<tr><th>Test Proc. ID</th><th colspan="3">Objective and Priority</th><th>Est. Duration</th></tr>
<tr><td colspan="5"><b>Start up:</b> Set apparatus ready for sampling analysis. Place NCS samples with following values in carousel:
1) Value of 1 2) Value of 2 3) Value of 56 4) Value of 315 5) Value of 316</td></tr>
<tr><td colspan="5"><b>Relationships to other procedures:</b> None</td></tr>
<tr><td colspan="5"><b>Test Log</b></td></tr>
<tr><td>Date:</td><td>Initials:</td><td colspan="2">Test item:</td><td>OK / Not OK</td></tr>
<tr><td></td><td></td><td colspan="2"></td><td></td></tr>
<tr><td colspan="5"><b>Comments:</b></td></tr>
<tr><td colspan="5"><b>Procedure</b></td></tr>
<tr><td><b>Test ID</b></td><td><b>Activities</b></td><td><b>Examination of Result</b></td><td><b>Actual Results</b></td><td><b>Test Result</b></td></tr>
<tr><td>1</td><td>Start the sampling analysis. Wait for first sample to be analysed.</td><td>Check that the display shows “Invalid sample”</td><td></td><td></td></tr>
<tr><td>2</td><td>Wait for second sample to be analysed.</td><td>Check that the sample is analysed</td><td></td><td></td></tr>
<tr><td>3</td><td>Wait for the sample to be analysed.</td><td>Check that the sample is analysed</td><td></td><td></td></tr>
<tr><td>4</td><td>Wait for fourth sample to be analysed.</td><td>Check that the sample is analysed</td><td></td><td></td></tr>
<tr><td>5</td><td>Wait for fifth sample to be analysed.</td><td>Check that the display shows “Invalid sample”</td><td></td><td></td></tr>
<tr><td colspan="5"><b>Stop and wrap up:</b> Turn off the apparatus, remove the samples, and clean up any spillage.</td></tr>
</table>

…

Annex K
(informative)

Test data requirements

K.1 Example 1 – Agile Corporation

Test Data Requirements

A modified set of live data needs to be populated, but data must not include critical customer data of credit card, address or phone number. This data will be “cleaned” by the test team and customer prior to release to production. Tests will be performed on the data used during the iterations.

K.2 Example 2 – Traditional Ltd

Test Data Requirements for PC part of UV/TRT-14 33a

Note that all data is needed for the entire system testing period, refer to [PTP].

ID	Description	Responsibility	Period	Resetting	Archive or Dispose
DBR-1	Full migration of the production database with all personal data obfuscated / scrambled	IT Department	15/3 - 30/5	Yes	A
…					
DBR-9	Sample type “1” with appropriate steps	IT Department	22/3 - 27/5	NA	D
DBR-10	Sample ID: “314”	IT Department	22/3 - 27/5	NA	D
…					

Resetting means that the IT department need to restore original database on request.

Annex L
(informative)

Test environment requirements

L.1 Example 1 – Agile Corporation

Test Environment Requirements

The test environment is an IBM compatible PC environment, with logon signature/password and "modified live" test data available on the test system configuration. Configuration testing on this environment is not planned, but functional and performance testing will be carried out on it.

L.2 Example – Traditional Ltd

Test Environment Requirements

ID	Item	Type	Description	Responsibility	Period Needed
E-1	Laptops	Hardware	6 x Windows laptops	Anne M. Hamer	15/9 – 31/10
E-2	Test Management Tool Licences	Tool	6 x tool licences	Ursula Parkinson	15/9 – 5/11
E-3	Test accounts	Accounts	6 x test accounts on system under test	Ursula Parkinson	15/9 – 5/11
E-4	SQL accounts	Accounts	6 x accounts for accessing SQL DB	Ursula Parkinson	30/9 – 31/10
E-5	Connection to test network	Network	All laptops require pre-configured connection to the test WIFI network	Anne M. Hamer	15/9 – 5/11
...					

Annex M
(informative)

Test data readiness report

M.1 Example 1 – Agile Corporation

Verbal statement made at daily stand-up: test data is now available in the test environment.

M.2 Example 2 – Traditional Ltd

Test Data Readiness Report

Summary: The test data is not ready. The data migration to the test environment will be completed by 17/3

Test data status: The following table displays the status of each test data requirement.

Requirement	Status	Comments
DBR-1	Delayed	Due to database maintenance, the data migration to the test environment will be complete by 17/3
...		
DBR-9	Ready	
DBR-10	Ready	
...		

Limitations: Upon the completion of testing, the test database will be refreshed. The database is staged specifically for this test and after testing the data will contain constraints and possess system states that require the refresh.

Conclusions and recommendations: Test data will be ready by 17/3