INTERNATIONAL STANDARD

ISO/IEC 26554

First edition
2018-11

Information technology — Software and systems engineering — Tools and methods for product line testing

Technologies de l'information — Ingénierie des systèmes et du logiciel — Outils et méthodes pour tester une gamme de produits

Reference number
ISO/IEC 26554:2018(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

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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

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

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

For an explanation 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.

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

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.

Introduction

The main purpose of this document is to deal with the capabilities of methods and tools of software and systems product line (SSPL) testing. This document defines how methods and tools can support the software and systems product line-specific testing processes.

Product line engineering sets up a common product line platform including identified key variability and develops individual systems on top of the platform. Variability realizes flexibility among member products, and it is closely related to the purpose of the product line reuse. In addition to the verification and validation of commonalities, domain testing generates reusable test artefacts, so as to minimize test efforts in application testing. However, variability continues throughout the product line life cycle, and its resolution phase is diverse. Thus, the complexity of product line testing becomes high. Testing in product line engineering differs from testing in the single system development in the following aspects:

— there are two testing life cycles, domain testing and application testing;

— test cases generated during domain testing can be incomplete due to unresolved variability;

— integration and system testing should be executed in the absence of non-functioning components because even complete test cases can interact with components or subsystems that include unresolved variability;

— application testing should be performed by reusing domain test assets and avoid retesting what has been tested during domain testing;

— test cases including variability are executable at different stages because the binding times of variabilities differ; and

— regression testing is performed in both domain testing and application testing. In application testing, when variability bindings are conducted, regression testing is performed as necessary.

This document addresses the product line-specific testing processes with the guidance of a set of tools' and methods' capabilities for supporting testing in software and systems product lines.

This document is intended to benefit the groups of people that acquire, supply, develop, operate and maintain tools and methods of testing for software and systems product lines. This document can be used in one or more of the following modes:

— by an organization intended to implement product lines – to understand, adopt and enact the processes, tools and methods of testing for product line. This also helps the organization evaluate and select relevant tools and methods based on business and user-related criteria;

— by a tool vendor who facilitate or leverage product line engineering practices – to provide a set of tool capabilities that should be embodied in a tool for supporting testing of a product line.

The ISO/IEC 26550 family of standards addresses both engineering and management processes and capabilities of methods and tools in terms of the key characteristics of product line development. This document provides processes and capabilities of methods and tools for product line testing. Other standards in the ISO/IEC 26550 family are as follows:

ISO/IEC 26550, ISO/IEC 26551, ISO/IEC 26555, ISO/IEC 26557, ISO/IEC 26558 and ISO/IEC 26559 are published. ISO/IEC 26552, ISO/IEC 26553, ISO/IEC 26556, ISO/IEC 26560, ISO/IEC 26561, ISO/IEC 26562 and ISO/IEC 26563 are planned International Standards. The following list provides an overview of the series:

— processes and capabilities of methods and tools for domain requirements engineering and application requirements engineering are provided in ISO/IEC 26551;

— processes and capabilities of methods and tools for domain design and application design are provided in ISO/IEC 26552;

— processes and capabilities of methods and tools for domain realization and application realization are provided in ISO/IEC 26553;

— processes and capabilities of methods and tools for technical management are provided in ISO/IEC 26555;

— processes and capabilities of methods and tools for organizational management are provided in ISO/IEC 26556;

— processes and capabilities of methods and tools for variability mechanisms are provided in ISO/IEC 26557;

— processes and capabilities of methods and tools for variability modelling are provided in ISO/IEC 26558;

— processes and capabilities of methods and tools for variability traceability are provided in ISO/IEC 26559;

— processes and capabilities of methods and tools for product management are provided in ISO/IEC 26560;

— processes and capabilities of methods and tools for technical probe are provided in ISO/IEC 26561;

— processes and capabilities of methods and tools for transition management are provided in ISO/IEC 26562;

— processes and capabilities of methods and tools for configuration management of asset are provided in ISO/IEC 26563;

— others (ISO/IEC 26564 to ISO/IEC 26599) are to be developed.

Information technology — Software and systems engineering — Tools and methods for product line testing

1 Scope

This document, within the methods and tools of testing for software and systems product lines:

— provides the terms and definitions specific to testing for software and systems product lines;

— defines processes performed during product line testing (those processes are described in terms of purpose, inputs, tasks and outcomes);

— defines method capabilities to support the defined tasks of each process; and

— defines tool capabilities to automate/semi-automate tasks or defined method capabilities.

This document concerns processes and capabilities of testing methods and tools for a family of products, not for a single system.

2 Normative references

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

ISO/IEC 26555, *Software and systems engineering — Tools and methods for product line technical management*

3 Terms and definitions

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

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

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

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

3.1
absent variants
variants that are not determined or developed at the specific time

3.2
application testing
sub-process of application engineering where domain test artefacts are reused to uncover evidence of defects in the application

3.3
application test asset
application-specific test asset that has reuse potential

3.4
aspect
special consideration within product line engineering process groups and tasks to which one can associate specialized methods and tools

3.5
domain testing
domain engineering phase whose role is to test domain artefacts

Note 1 to entry: Testing of artefacts related to a product line (domain) rather than to an individual product.

3.6
domain test asset
domain test artefacts that will be reused in *application testing* (3.2)

Note 1 to entry: Domain test assets can be reused in *domain testing* (3.5), e.g., test plans, artefacts of regression testing. In this document, however, the platform role of domain asset, which can be reused for member products, is the central theme.

3.7
domain test requirements
specific elements of a domain artefact that should be covered by *domain testing* (3.5)

Note 1 to entry: Domain test requirements cover functional and non-functional commonality and variability, and they include both their normal and error conditions.

3.8
product line test strategy
scope of *domain testing* (3.5) and *application testing* (3.2)

Note 1 to entry: Many variants may not be implemented in domain engineering. To cope with the impacts of unimplemented variants to domain integration testing and system testing, the appropriate test strategy should be established.

3.9
texture
architectural texture
collection of common development rules and constraints for realizing the applications of a product line

3.10
variability binding
selection of a certain variant for a variation point

3.11
variability in test cases
variability included in domain test cases that will be bound during application testing in order to derive concrete test cases

4 Abbreviated terms

CRS	commonality and reuse strategy
SAS	sample application strategy
SSPL	software and systems product line
UML	unified modelling language

5 Reference model for product line testing

5.1 Overview

The key characteristics of product line testing are that there are two test processes: domain testing and application testing, with variability included in domain artefacts. The major difference between

SSPL testing and single product testing comes from variability in domain artefacts. Handling variability is challenging and the decisions on how to deal with it are the starting point of SSPL testing. From this view, there are two types of test assets, domain test assets (test cases, procedures and scenarios that are produced during domain engineering) and application test assets (test cases, procedures and scenarios that are produced during application engineering for a member product).

Domain test assets should have forms that can be efficiently reused in application testing; and they should address variations. In addition, testing for a member product of a product line should devise a method to reuse domain test cases and minimize regression testing for the parts that are already tested in domain testing or tested by another member product. It is unusual for all variants to be implemented during domain engineering. So, domain testing should consider tests for non-executable domain artefacts due to the undeveloped variants, i.e. absent variants. Coping with absent variants is challenging because they can complicate integration testing and system testing, making system testing in domain engineering impossible in some cases. However, because the quality of domain artefacts affects the quality of all member products in a product line, test execution of parts linked to absent variants should be carefully handled during domain testing.

Variability is bound during application engineering. The binding times of variability can be widely separate, so integration testing and system testing for a member product are much more complicated. If testing is executed whenever binding occurs, a significant amount of effort will be required for developing test stubs and drivers. Therefore, the trade-off between defect correction costs and test development effort should be considered. Application testing deals with testing for application-specific parts and with re-testing or regression testing for the domain parts already tested.

Product line testing requires specific organizational level managerial supports and technical management supports, product line test management and variability management in testing. Product line test management provides managerial supports for domain testing, application testing and asset management in testing from the product line organization level. Variability management in testing, on the other hand, manages variability in domain and application test artefacts, including variability models from the testing perspective.

The reference model specifies the structure of supporting processes and sub processes for product line testing. As shown in Figure 1, product line testing can be structured in five processes. Each process is divided into sub processes to address product line realization issues; and each sub process is described in terms of the following attributes:

— the title of the sub process;

— the purpose of the sub process;

— the outcomes of the sub process;

— the inputs to produce the outcomes;

— the tasks to achieve the outcomes; and

— the capabilities of methods and tools required for performing the tasks effectively and efficiently.

NOTE 1 Test methods in this document include technique, pattern and template as well as procedure.

NOTE 2 The ISO/IEC/IEEE 29119 series defines “test methods” as “procedure used to derive and/or select test cases”.

NOTE 3 When the process, sub process, outcomes and tasks are listed or described in a sentence they are italicized in order to increase their visibility.

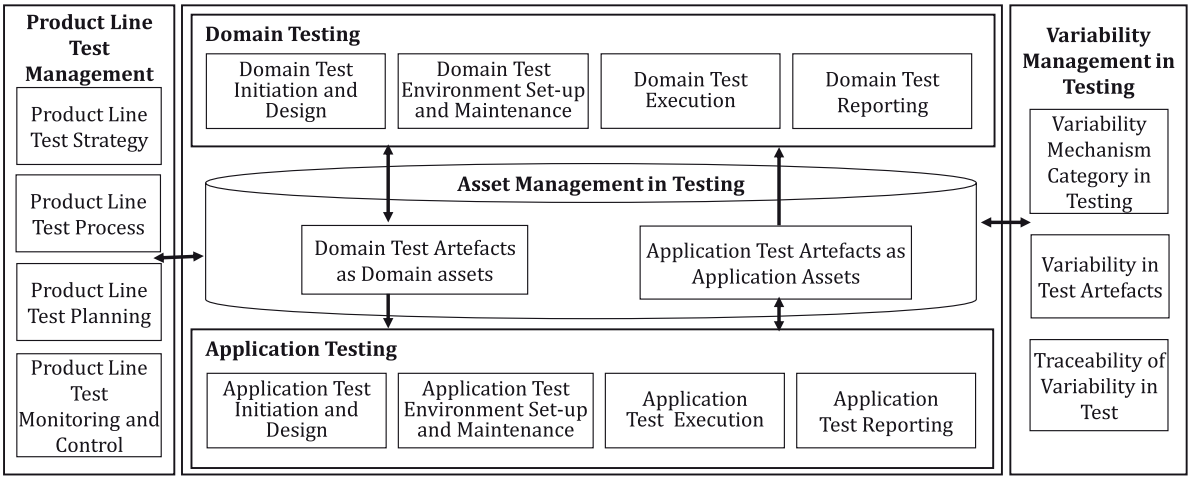

Figure 1 — Product line testing reference model

To reduce efforts and costs invested in product line testing and to manage the complexity of testing due to the variability in a product line, testing activities should be initiated as early as at the domain requirements analysis phase, in parallel with domain and application engineering. Defects in domain assets have influence on the whole product line members and dynamic testing is performed restrictedly because of variability, so static testing should be performed thoroughly. Product line testing thus covers both static and dynamic testing. Static testing for product lines includes reviews, inspections and prototyping for domain and application artefacts to fulfil verification and validation. In addition, testers responsible for domain and application testing should cooperate with domain requirements engineers, architects and developers in order to detect defects in the corresponding development phase as early as possible. To that end, these test organization should develop reasonable domain test plans and have systematic test case generation methods including tool supports at hand for each domain test levels or test types.

5.2 Product line test management

The product line test management shall be responsible for the following.

— *Product line test strategy* determines how domain and application testing will be carried out at the product line organization. The product line test strategy documents guidelines for governing product line testing operated in the organization.

— *Product line test process* is to define the process specific to a product line testing by selecting and tailoring a generic domain and application test process to solve domain and application-specific test problems.

— *Product line test planning* is to develop the product line test plan. It describes activities, schedules and resources for product line testing, and product line plans are revisited and refined as domain and application test plans for conducting the specific test types or test levels of domain and application testing.

— *Product line test monitoring and control* monitor and control the status of product line testing to ensure it is performed in line with the product line test plan. It serves to monitor, measure and control domain and application test progress. Domain and application test status are integrated into product line-wide management.

5.3 Domain testing

In domain testing, normally there is no single, executable application to be tested, since only the platform consisting of a set of loosely coupled components exists, and thus product line policies and processes reflecting this aspect are necessary. Domain testing covers the generation of domain test

artefacts and executions, and making them domain assets for reuse. Domain testing is responsible for validation and verification of domain requirements specification, domain architecture, detailed domain design artefacts and realization artefacts. However, system testing in domain engineering may be partially executed or may not be performed because there may be no complete application during domain engineering. Therefore, domain testing should have a proper strategy for these cases.

The domain testing shall serve to do the following and define the capabilities of tools and methods for supporting those services.

— *Domain test initiation and design* set up the readiness for performing domain testing, derive domain test cases and define domain test procedures.

— *Domain test environment set-up and maintenance* set up and maintain domain test environments including support tools as well as integrated environments for each lifecycle phase related to each domain test type (i.e. static test and dynamic test).

— *Domain test execution* performs dynamic and static testing for the domain on the established domain test environments. Domain test cases may include variability that cannot be executed simultaneously.

— *Domain test reporting* analyses the identified failures and locations where the failures have occurred and reports domain test incidents with how they will be managed.

5.4 Asset management in testing

The asset management in testing establishes and maintains domain test assets that have reuse potential for the right use in application engineering and application test assets for later reuse by the relevant member product or the other member products within a product line. The asset management in testing shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Domain test artefacts as domain assets* manages domain test artefacts that will be reused by the product line members such as test cases for commonalities, those for variabilities and those including interactions between common and variable modules.

— *Application test artefacts as application assets* manages application test artefacts that will be referred in regression testing due to the application evolution or as inputs in other applications.

5.5 Variability management in testing

Variability is newly defined during domain testing for handling test requirements variation among member products. Domain test assets include variability because of the variability included in test targets (i.e. domain assets). Therefore methods and tools for product line testing should have the capability for variability representation and its binding at the right time. The variability management in testing shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Variability mechanism category in testing* maintains a set of variability implementation mechanisms that can be used for expressing or realizing variability in domain test artefacts.

— *Variability in test artefacts* serves to define variability types and ways to express variability included in test artefacts such as test plans, test cases and test scenarios.

— *Traceability of variability in test* establishes and maintains trace links between variability in test artefacts and variability models in test defined separately. The trace links are required to support the reuse of test artefacts in application testing.

5.6 Application testing

Domain test assets should be correctly reused in application testing. The application testing validates the final member product against the application requirements by drawing upon test assets from domain test assets and by performing additional application-specific tests. This process shall guide to

minimize test efforts by avoiding redundant test execution and reusing domain assets. Furthermore, this process covers all test types and levels from static testing for application through acceptance testing. The application testing shall serve to do the following and to define the capabilities of tools and methods for supporting them.

— *Application test initiation and design* set up the readiness for performing application testing, derive application-specific test cases and define application-specific test procedures. Application test cases and test procedures are derived from domain test assets, so this process only deals with application-specific test cases and procedures.

— *Application test environment set-up and maintenance* set up and maintain application test environments based on the established domain test environment.

— *Application test execution* performs application dynamic and static testing on the established application test environments.

— *Application test reporting* analyses the identified failures and location where the failure has occurred. Application test reports include application-specific and domain test incidents as well as how they will be managed.

The identification and analysis of the key differentiators between single-system engineering and management and product line engineering and management can help organizations to understand the product line and to formulate a strategy for successful implementation of product line engineering and management. The key aspects are defined in ISO/IEC 26550 and Table 1 shows the category of the key aspects.

Table 1 — Key aspects for identifying product line-specific testing tasks

Category	Aspects
Reuse management	application engineering, domain assets, domain engineering, product management, platform, reusability
Variability management	binding, variability
Complexity management	collaboration, configuration, enabling technology support, reference architecture, texture, traceability
Quality management	measurement and tracking, cross functional verification and validation

The following are the descriptions for each aspect concerning product line testing. The product line tests-relevant processes and tasks shall be identified on the basis of these aspects. The concerns specific to product line testing will enable an organization to understand the product line test-relevant processes, sub processes, tasks, methods and tools' capabilities.

— **Application engineering**. Application testing is a stage of application engineering where the domain test artefacts are reused. Application-specific testing is developed from scratch.

— **Binding**. Most application test artefacts are generated by variability binding according to application variability model, so testing processes, methods and tools for product lines should contain capabilities to support binding.

— **Collaboration**. For efficient and effective testing, preliminary artefacts for testing should be developed at each development phase. Hence, test managers/engineers and the corresponding participants of each development phase should collaborate with each other for producing test artefacts and ensuring testability of artefacts.

— **Configuration**. Since the test artefacts include product variations, configurations of test artefacts with the corresponding development artefacts should be consistently managed.

— **Domain asset**. Domain test artefacts such as test plans, test cases including test data and test suites that will be reused in the application testing are the major domain assets of product line testing.

— **Domain engineering**. Testing for product lines should be prepared and executed from the initial domain engineering phase combined with the characteristics of each domain engineering process.

— **Enabling technology support**. Proper technologies for producing domain test assets and for avoiding redundant testing in the member product testing should be supported.

— **Measurement and tracking**. The status of defects in domain assets should be carefully traced, because they have significant ripple effects on member products. Effectiveness and efficiency of domain testing and application testing should be measured for improvement.

— **Platform**. Testing should help ensure that the platform fulfils the specified domain requirements and its intended use.

— **Product management**. Testing evolves in accordance with the evolution and changes of product lines.

— **Reference architecture**. The reference architecture is the major input for designing domain integration testing.

— **Reusability**. Domain testing assets are developed as available artefacts as they are (or are partly) in a generic form so that a large part of the required testing in applications should not be developed from scratch.

— **Texture**. Domain testing and application testing should help ensure both of them conform to the corresponding guidelines and rules in the texture.

— **Traceability**. For each test artefact, a trace link is established to the corresponding references. For example, when system test cases are derived from use cases, each system test case is related to the corresponding use case by a trace link.

— **Cross functional validation and verification**. Detailed processes for validation and verification are defined in this document.

— **Variability**. Test artefacts contain variability, so that each member product selects variants to verify, validate and test its artefacts.

6 Product line test management

6.1 General

The major difference between SSPL testing and single product testing comes from variability. Thus, handling variability is very challenging; and the decisions on how to deal with it are the starting point of SSPL testing. This decision should be made from the organizational level by considering the efficiency and effectiveness of testing. From this decision, test strategy, test process and necessary test environments including organizational level support can be determined. A product line test management process includes the following four key sub processes.

— *Product line test strategy* provides technical guidelines for performing domain and application testing. Product line test strategy provides the scope of domain and application testing, which means that domain artefacts will be tested during domain testing and artefacts will be deferred to application testing (see Annex A for an exemplar product line test strategy).

— *Product line test process* is used to solve product line-specific test problems. Product line test process is defined by selecting and tailoring a generic domain and application test process.

— *Product line test planning* provides test plans for performing product line tests throughout the product line lifecycle phases. Domain and application test planning is based on domain and application engineering artefacts, i.e. requirements, architecture, detailed design and implementation artefacts, and the variability model of both domain and application engineering.