INTERNATIONAL STANDARD

ISO/IEC/ IEEE 29119-4

First edition
2015-12-01

Software and systems engineering — Software testing —

Part 4: Test techniques

Ingénierie du logiciel et des systèmes — Essais du logiciel —

Partie 4: Techniques d'essai

Reference number
ISO/IEC/IEEE 29119-4:2015(E)

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 on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, SC 7, *Software and Systems Engineering*.

ISO/IEC/IEEE 29119 consists of the following standards, under the general title *Software and Systems Engineering — Software Testing*:

— *Part 1: Concepts and definitions*

— *Part 2: Test processes*

— *Part 3: Test documentation*

— *Part 4: Test techniques*

The following parts are under preparation:

— *Part 5: Keyword-driven testing*

Introduction

The purpose of this part of ISO/IEC/IEEE 29119 is to provide an International Standard that defines software test design techniques (also known as test case design techniques or test methods) that can be used within the test design and implementation process that is defined in ISO/IEC/IEEE 29119-2. This part of ISO/IEC/IEEE 29119 does not prescribe a process for test design and implementation; instead, it describes a set of techniques that can be used within ISO/IEC/IEEE 29119-2. The intent is to describe a series of techniques that have wide acceptance in the software testing industry.

The test design techniques presented in this part of ISO/IEC/IEEE 29119 can be used to derive test cases that, when executed, generate evidence that test item requirements have been met and/or that defects are present in a test item (i.e. that requirements have not been met). Risk-based testing could be used to determine the set of techniques that are applicable in specific situations (risk-based testing is covered in ISO/IEC/IEEE 29119-1 and ISO/IEC/IEEE 29119-2).

NOTE A "test item" is a work product that is being tested (see ISO/IEC/IEEE 29119-1).

EXAMPLE 1 "Test items" include systems, software items, objects, classes, requirements documents, design specifications, and user guides.

Each technique follows the test design and implementation process that is defined in ISO/IEC/IEEE 29119-2 and shown in Figure 1.

Of the activities in this process, ISO/IEC/IEEE 29119-4 provides guidance on how to implement the following activities in detail for each technique that is described:

— Derive Test Conditions (TD2);

— Derive Test Coverage Items (TD3);

— Derive Test Cases (TD4).

A test condition is a testable aspect of a test item, such as a function, transaction, feature, quality attribute, or structural element identified as a basis for testing. This determination can be achieved by agreeing with stakeholders which attributes are to be tested or by applying one or more test design techniques.

EXAMPLE 2 If a test completion criterion for state transition testing was identified that required coverage of all states then the test conditions could be the states the test item can be in. Other examples of test conditions are equivalence classes and boundaries between them.

Test coverage items are attributes of each test condition that can be covered during testing. A single test condition may be the basis for one or more test coverage items.

EXAMPLE 3 If a specific boundary is identified as a test condition, then the corresponding test coverage items could be the boundary itself and immediately either side of the boundary.

A test case is a set of preconditions, inputs (including actions, where applicable), and expected results, developed to determine whether or not the covered part of the test item has been implemented correctly.

Specific (normative) guidance on how to implement the other activities in the test design & implementation process of ISO/IEC/IEEE 29119-2, including activities TD1 (Identify Feature Sets), TD5 (Assemble Test Sets), and TD6 (Derive Test Procedures), is not included in Clauses 5 or 6 of this part of ISO/IEC/IEEE 29119 because the process is the same for all techniques.

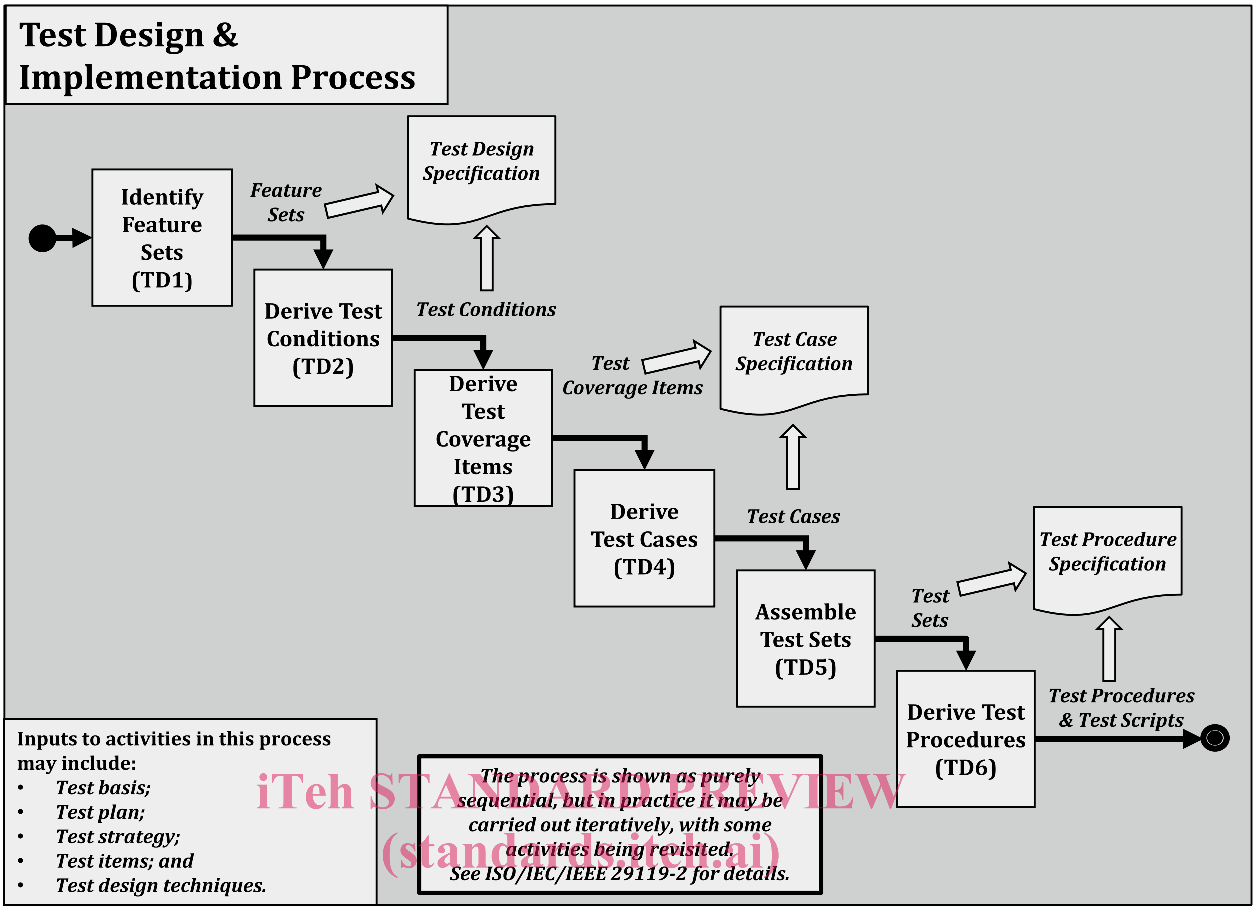

Figure 1 — ISO/IEC/IEEE 29119-2 Test Design and Implementation Process

ISO/IEC/TR 19759 (SWEBOK) defines two types of requirements: functional requirements and quality requirements. ISO/IEC 25010 defines eight quality characteristics (including functionality) that can be used to identify types of testing that may be applicable for testing a specific test item. Annex A provides example mappings of test design techniques that apply to testing quality characteristics defined in ISO/IEC 25010.

Experience-based testing practices like exploratory testing and other test practices such as model-based testing are not defined in this part of ISO/IEC/IEEE 29119 because this part of ISO/IEC/IEEE 29119 only describes techniques for designing test cases. Test practices such as exploratory testing are described in ISO/IEC/IEEE 29119-1.

Templates and examples of test documentation that are produced during the testing process are defined in ISO/IEC/IEEE 29119-3 Test Documentation. The test techniques in this part of ISO/IEC/IEEE 29119 do not describe how test cases should be documented (e.g. they do not include information or guidance on assigning unique identifiers, test case descriptions, priorities, traceability, or pre-conditions). Information on how to document test cases can be found in ISO/IEC/IEEE 29119-3.

This part of ISO/IEC/IEEE 29119 aims to provide stakeholders with the ability to design test cases for the testing of software in any organization.

Software and systems engineering — Software testing —

Part 4: **Test techniques**

1 Scope

This part of ISO/IEC/IEEE 29119 defines test design techniques that can be used during the test design and implementation process that is defined in ISO/IEC/IEEE 29119-2.

This part of ISO/IEC/IEEE 29119 is intended for, but not limited to, testers, test managers, and developers, particularly those responsible for managing and implementing software testing.

2 Conformance

2.1 Intended Usage

The normative requirements in this part of ISO/IEC/IEEE 29119 are contained in Clauses 5 and 6. It is recognised that particular projects or organizations may not need to use all of the techniques defined by this standard. Therefore, implementation of this standard typically involves selecting a set of techniques suitable for the project or organization. There are two ways that an organization or individual can claim conformance to the provisions of this standard – full conformance or tailored conformance. The organization or individual shall assert whether full or tailored conformance to this standard is claimed.

2.2 Full Conformance

Full conformance is achieved by demonstrating that all of the requirements (i.e. 'shall' statements) of the chosen (non-empty) set of techniques in Clause 5 and/or the corresponding test coverage measurement approaches in Clause 6 have been satisfied.

EXAMPLE An organization could choose to conform only to one technique, such as boundary value analysis. In this scenario, the organization would only be required to provide evidence that they have met the requirements of that one technique in order to claim conformance to this part of ISO/IEC/IEEE 29119.

2.3 Tailored Conformance

Tailored conformance is achieved by demonstrating that the chosen subset of requirements from the chosen (non-empty) set of techniques and/or corresponding test coverage measurement approaches have been satisfied. Where tailoring occurs, justification shall be provided (either directly or by reference) whenever the normative requirements of a technique defined in Clause 5 or measure defined in Clause 6 are not followed completely. All tailoring decisions shall be recorded with their rationale, including the consideration of any applicable risks. Tailoring shall be agreed by the relevant stakeholders.

3 Normative References

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC/IEEE 29119-1, *Software and systems engineering — Software testing — Part 1: Concepts and definitions*

ISO/IEC/IEEE 29119-2, *Software and systems engineering — Software testing — Part 2: Test processes*

ISO/IEC/IEEE 29119-3, *Software and systems engineering — Software testing — Part 3: Test documentation*

NOTE Other International Standards useful for the implementation and interpretation of this part of ISO/IEC/IEEE 29119 are listed in the bibliography.

4 Terms and Definitions

For the purposes of this document, the terms and definitions given in ISO/IEC/IEEE 24765 and the following apply.

NOTE Use of the terminology in this part of ISO/IEC/IEEE 29119 is for ease of reference and is not mandatory for conformance with the standard. The following terms and definitions are provided to assist with the understanding and readability of this part of ISO/IEC/IEEE 29119. Only terms critical to the understanding of this part of ISO/IEC/IEEE 29119 are included. This clause is not intended to provide a complete list of testing terms. The systems and software engineering vocabulary ISO/IEC/IEEE 24765 can be referenced for terms not defined in this clause.

4.1
Backus-Naur Form
formal meta-language used for defining the syntax of a language in a textual format

4.2
base choice
see base value

4.3
base value
input parameter value used in 'base choice testing' that is normally selected based on being a representative or typical value for the parameter. Also called base choice

4.4
c-use
see computation data use

4.5
computation data use
use of the value of a variable in any type of statement

4.6
condition
Boolean expression containing no Boolean operators

EXAMPLE "A < B" is a condition but "A and B" is not.

4.7
control flow
sequence in which operations are performed during the execution of a test item

4.8
control flow sub-path
sequence of executable statements within a test item

4.9
data definition
statement where a variable is assigned a value. Also called variable definition

4.10
data definition c-use pair
data definition and subsequent computation data use, where the data use uses the value defined in the data definition

4.11
data definition p-use pair
data definition and subsequent predicate data use, where the data use uses the value defined in the data definition

4.12
data definition-use pair
data definition and subsequent data use, where the data use uses the value defined in the data definition

4.13
data use
executable statement where the value of a variable is accessed

4.14
decision outcome
result of a decision (which therefore determines the control flow alternative taken)

4.15
decision rule
combination of conditions (also known as causes) and actions (also known as effects) that produce a specific outcome in decision table testing and cause-effect graphing

4.16
definition-use pair
data definition and subsequent predicate or computational data use, where the data use uses the value defined in the data definition

4.17
definition-use path
control flow sub-path from a variable definition to a predicate-use (p-use) or computational-use (c-use) of that variable

4.18
entry point
point in a test item at which execution of the test item can begin

Note 1 to entry: An entry point is an executable statement within a test item that may be selected by an external process as the starting point for one or more paths through the test item. It is most commonly the first executable statement within the test item.

4.19
executable statement
statement which, when compiled, is translated into object code, which will be executed procedurally when the test item is running and may perform an action on program data

4.20
exit point
last executable statement within a test item

Note 1 to entry: An exit point is a terminal point of a path through a test item, being an executable statement within the test item which either terminates the test item, or returns control to an external process. This is most commonly the last executable statement within the test item.

4.21
p-use
see predicate data use

4.22
P-V pair
combination of a test item parameter with a value assigned to that parameter, used as a test condition and coverage item in combinatorial test design techniques

4.23
path
sequence of executable statements of a test item

4.24
predicate
logical expression which evaluates to TRUE or FALSE, normally to direct the execution path in code

4.25
predicate data use
data use associated with the decision outcome of the predicate portion of a decision statement

4.26
sub-path
path that is part of a larger path

4.27
test model
representation of a test item that is used during the test case design process

4.28
variable definition
see data definition

5 Test Design Techniques

5.1 Overview

ISO/IEC/IEEE 29119-4 defines test design techniques for specification-based testing (5.2), structure-based testing (5.3) and experience-based testing (5.4). In specification-based testing, the test basis (e.g. requirements, specifications, models or user needs) is used as the main source of information to design test cases. In structure-based testing, the structure of the test item (e.g. source code or the structure of a model) is used as the primary source of information to design test cases. In experience-based testing, the knowledge and experience of the tester is used as the primary source of information during test case design. For specification-based testing, structure-based testing and experience-based testing, the test basis is used to generate the expected results. These classes of test design techniques are complementary and their combined application typically results in more effective testing.

Although the techniques presented in ISO/IEC/IEEE 29119-4 are classified as structure-based, specification-based or experience-based, in practice some of them can be used interchangeably (e.g. branch testing could be used to design test cases for testing logical paths through the graphical user interface of an Internet-based system). This is demonstrated in Annex E. In addition, although each technique is defined independently of all others, in practice they can be used in combination with other techniques.

EXAMPLE The test coverage items derived by applying equivalence partitioning could be used to populate input parameters of test cases derived using scenario testing.

ISO/IEC/IEEE 29119-4 uses the terms specification-based testing and structure-based testing; these categories of techniques are also known as “black-box testing” and “white-box testing” (or “clear-box testing”) respectively. The terms “black-box” and “white-box” refer to the visibility of the internal structure of the test item. In black-box testing the internal structure of the test item is not visible (hence the black box), whereas for white-box testing the internal structure of the test item is visible. When a technique is applied while utilising a combination of knowledge from the test item’s specification and structure, this is often called “grey-box testing”.

ISO/IEC/IEEE 29119-4 defines how the generic test design and implementation process steps TD2 (derive test conditions), TD3 (derive test coverage items), and TD4 (derive test cases) from ISO/IEC/IEEE 29119-2 (see Introduction) shall be used by each technique. It does not provide context-specific definitions of the techniques that describe how each technique should be used in all situations. Users of ISO/IEC/IEEE 29119-4 may refer to Annex B, Annex C, Annex D and Annex E for detailed examples that demonstrate how to apply the techniques.

The techniques that are defined in ISO/IEC/IEEE 29119-4 are shown in Figure 2. This set of techniques is not exhaustive. There are techniques that are used by testing practitioners or researchers that are not included in ISO/IEC/IEEE 29119-4.

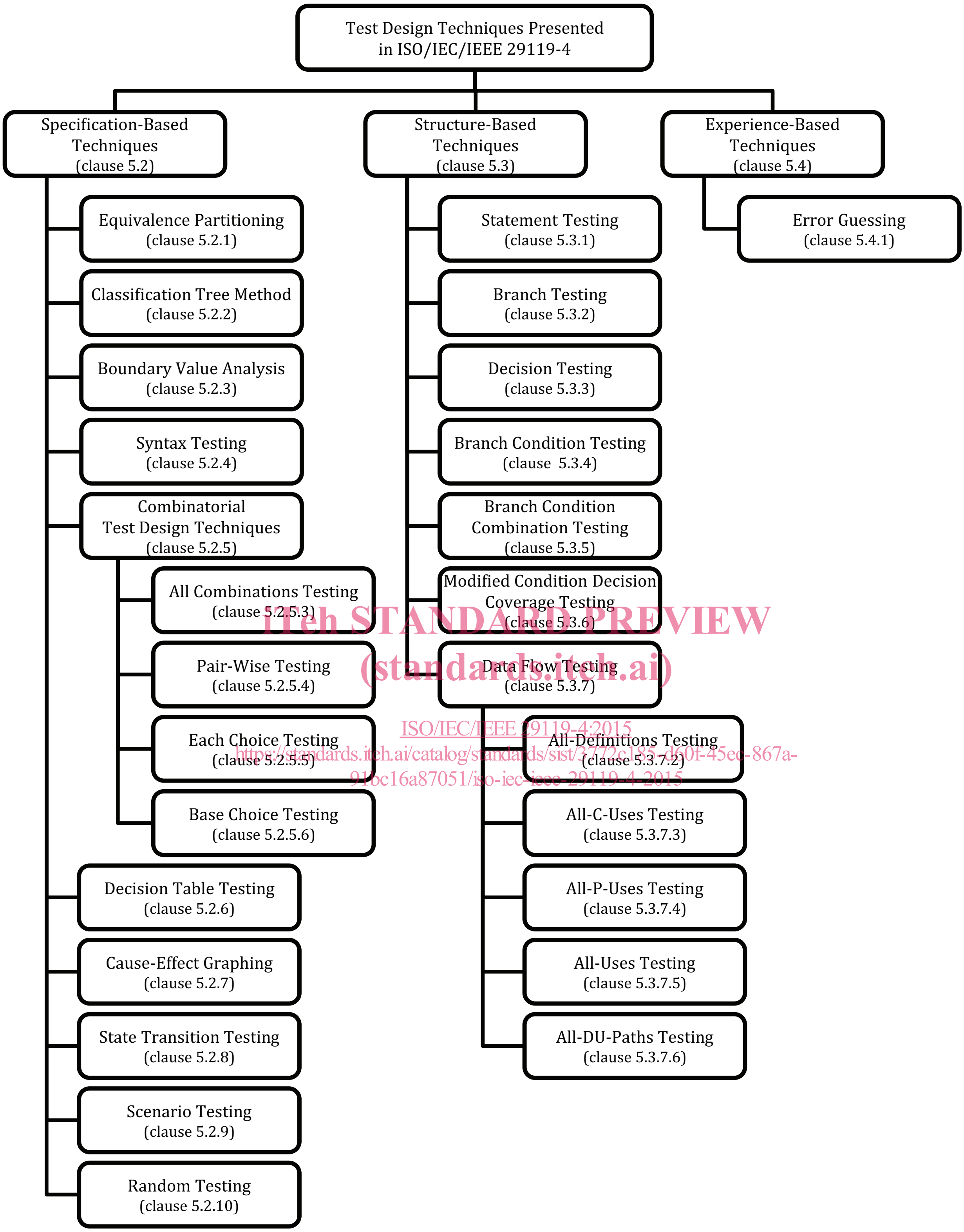

Figure 2 — The set of test design techniques presented in ISO/IEC/IEEE 29119-4

Of the six activities in the test design and implementation process (see Figure 1), test design techniques provide unique and specific guidance on the derivation of test conditions (TD2), test coverage items (TD3) and test cases (TD4). Therefore, each technique is defined in terms of these three activities.

There are varying levels of granularity within steps TD2 (derive test conditions), TD3 (derive test coverage items) and TD4 (derive test cases). Within each technique, the term "model" is used to describe the concept of preparing a logical representation of the test item for the purposes of deriving test conditions in step TD2 (e.g. a control flow model is required for deriving test conditions for all structural techniques). Some situations may require the entire model to be a test condition, whereas in other situations, one part of the model may be a test condition.

EXAMPLE 1 In state transition testing, if there is a requirement to cover all states then the entire state model could be the test condition. Alternatively, if there is a requirement to cover specific transitions between states, then each transition could be a test condition.

In addition, since some techniques share underlying concepts, their definitions contain similar text.

EXAMPLE 2 Both equivalence partitioning and boundary value analysis are based on equivalence classes.

In the test case design step (TD4) of each technique, test cases that are created may be “valid” (i.e. they contain input values that the test item should accept as correct) or “invalid” (i.e. they contain at least one input value that the test item should reject as incorrect, ideally with an appropriate error message). In some techniques, such as equivalence partitioning and boundary value analysis, invalid test cases are usually derived using the “one-to-one” approach as it avoids fault masking by ensuring that each test case only includes one invalid input value, while valid test cases are typically derived using the “minimized” approach, as this reduces the number of test cases required to cover valid test coverage items (see 5.2.1.3 and 5.2.3.3).

NOTE Invalid cases are also known as “negative test cases”.

Although the techniques defined in ISO/IEC/IEEE 29119-4 are each described in a separate clause (as if they were mutually exclusive), in practice they could be applied in a blended way.

EXAMPLE 3 Boundary value analysis could be used to select test input values, after which pair-wise testing could be used to design test cases from the test input values. Equivalence partitioning could be used to select the classifications and classes for the classification tree method and then each choice testing could be used to construct test cases from the classes.

The techniques presented in ISO/IEC/IEEE 29119-4 could also be used in conjunction with the test types that are presented in Annex A. For example, equivalence partitioning could be used to identify user groups (test conditions) and representative users (test coverage items) from those groups in test cases that are to be tested during usability testing.

The normative definitions of the techniques are provided in Clause 5. The corresponding normative coverage measures for each technique are presented in Clause 6. This is supported by informative examples of each technique in Annexes B, C, D and E. Although the examples of each technique demonstrate manual application of the technique, in practice, automation can be used to support some types of test design and execution (e.g. statement coverage analyzers can be used to support structure-based testing). Annex A provides examples of how the test design techniques defined in this standard can be applied to testing the quality characteristics that are defined in ISO/IEC 25010.

5.2 Specification-Based Test Design Techniques

5.2.1 Equivalence Partitioning

5.2.1.1 Derive Test Conditions (TD2)

Equivalence partitioning (BS 7925-2:1998; Myers 1979) uses a model of the test item that partitions the inputs and outputs of the test item into equivalence partitions (also called “partitions” or “equivalence classes”), where each equivalence partition shall be defined as a test condition. These equivalence partitions shall be derived from the test basis, where each partition is chosen such that all values within the equivalence partition can reasonably be expected to be treated similarly (i.e. they may be