DRAFT International Standard

ISO/IEC DIS 19512

Information technology — Structured patterns metamodel standard (SPMS) 1.4

ICS: 35.080

ISO/IEC JTC **1**

Secretariat: **ANSI**

Voting begins on:
2026-07-28

Voting terminates on:
2026-10-20

THIS DOCUMENT IS A DRAFT CIRCULATED FOR COMMENTS AND APPROVAL. IT IS THEREFORE SUBJECT TO CHANGE AND MAY NOT BE REFERRED TO AS AN INTERNATIONAL STANDARD UNTIL PUBLISHED AS SUCH.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

RECIPIENTS OF THIS DRAFT ARE INVITED TO SUBMIT, WITH THEIR COMMENTS, NOTIFICATION OF ANY RELEVANT PATENT RIGHTS OF WHICH THEY ARE AWARE AND TO PROVIDE SUPPORTING DOCUMENTATION.

This document has not been edited by the ISO Central Secretariat.

Reference number
ISO/IEC DIS 19512:2026(en)

© ISO/IEC 2026



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2026

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

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

Published in Switzerland

Contents

Introduction 1
1 Scope 1
2 Conformance 2
3 Normative References 2
4 Terms and Definitions 2
5 Additional Information 3
5.1 Acknowledgements 3
6 SPMS Overview (Informative) 4
7 Definitions Classes 6
7.1 Introduction 6
7.2 PatternElement (Abstract) 6
7.3 PatternDefinition 7
7.4 Role 7
7.5 PatternSection 8
8 Observations Classes 9
8.1 Introduction 9
8.2 Binding 9
8.3 PatternInstance 10
8.4 PatternObservation 10
9 Formalisms Classes 11
9.1 Introduction 11
9.2 FormalizedDefinition (Abstract) 12
9.3 Assertion 12
9.4 BooleanExpression (Abstract) 12
9.5 AndExpression 13
9.6 OrExpression 13
9.7 NotExpression 13
9.8 DefinitionTerminal 13
9.9 FreeVariable 14
9.10 FormalBinding (Abstract) 14
9.11 VariableToRole 14
9.12 PropertyToRole 14
9.13 PropertyToVar 15
10 Relationships Classes 16
10.1 Introduction 16
10.2 InterpatternRelationship (Abstract) 16
10.3 RelatedPattern 16
10.4 MemberOf 17
10.5 Perspective 17
10.6 Nature 17

© ISO/IEC 2026 – All rights reserved

10.7 Category 18
10.8 KnownUse 18
11 PIN Classes 19
11.1 Introduction 19
11.2 Overview 19
11.3 PINbox Class 20
11.3.1 Overview 20
11.3.2 Collapsed 21
11.3.3 Standard 21
11.3.4 Expanded 21
11.4 Equality Class 22
11.5 BindingGlyph Class 24
11.6 Multiplicities 26
11.6.1 Overview 26
11.6.2 Stacked PINbox 26
11.6.3 MultiBranched Annotation 26
11.7 Peeling and Coalescing 29
12 PHORML Overview (Informative) 31
13 PHORML::Core Classes (Informative) 33
13.1 Introduction 33
13.2 Entity (Abstract) 33
13.3 Model 33
13.4 NamedEntity (Abstract) 34
14 PHORML::RequiredEntitySet Classes (Informative) 35
14.1 Introduction 35
14.2 TypedEntity (Abstract) 35
14.3 MethodAndFieldContainer (Abstract) 35
14.4 Object 36
14.5 Method 36
14.6 Field 36
14.7 Type 37
15 PHORML::Reliances Classes (Informative) 38
15.1 Introduction 38
15.2 RelianceBase 39
15.3 Method Invocation 39
15.4 Field Use 40
15.5 State Change 40
15.6 Cohesion 40
Annex A: EntityExtension Examples (Informative) 41
Annex B: Procedural Language Modeling (Informative) 41
Annex C: AST-Based Pattern Metamodel Language (APML) (Informative) 43
Bibliography 47

© ISO/IEC 2026 – All rights reserved

Preface

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at https://www.omg.org/.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at: https://www.omg.org/spec

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
9C Medway Road, PMB 274
Milford, MA 01757

USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult https://www.iso.org

© ISO/IEC 2026 – All rights reserved

Introduction

Patterns are ubiquitous in software design, production, analysis, and maintenance. Numerous communities have arisen that support the authoring and curating of patterns of various kinds, including anti-patterns, security patterns, design patterns, architectural patterns, build patterns, and so on. There is a great need for a standard for the sharing of this information, both within and between these patterns communities. This document describes and defines a metamodel for use by these communities, to support several use cases of patterns in software, without specifying how communities should use the metamodel for their own purposes. This standard creates a foundation for information sharing, and leaves the details of which precise information to be stored and shared up to the communities that will be using that information.

For example, a design pattern community will be concerned with patterns of software design, while an architecture pattern community will be concerned with patterns of system design. Both communities have common needs surrounding how to organized the definitions of patterns, how to relate and categorize definitions, how to report on observed instances of patterns within their context, a need to display those instances to a user, and how to describe those patterns in an appropriate formal manner for their community. The context of those communities is independent of these needs, which are common to working with patterns regardless of the domain. This specification defines a container for sharing the above information.

1 Scope

The Structured Patterns Metamodel Standard (SPMS) specification defines a common standard for the definition and description of patterns as used in architecting, designing, and implementing software systems, working with software faults or security issues, and any situation where a pattern is appropriately applied.

SPMS has three main goals:

1) Sharing of pattern definitions in repositories or catalogs, including human-oriented specifications and machine-oriented formalisms for automated tool use.

2) Sharing of pattern instances – indicators of the existence of a pattern within a model – regardless of how that pattern was determined, with traceability back to the methodology, and traceability to the model artifacts that prove its existence. if applicable. These instances may come from manual assertion, or from the results of an automated tool.

3) A visual representation for pattern instances that augments existing modeling representations and supports both automated production of graphical diagrams, and informal “line and box” style human-generated sketching.

The first goal is supported by the **Definitions** package, which defines a metamodel for defining and storing pattern specifications, suitable for use in tooling and repositories.

The second goal is supported by the **Observations** package, which defines a metamodel for pattern instances. The classes defined here offer support for both human-oriented use cases (consulting, investigation, education) and machine-oriented use cases (automated analysis tools, automated results analysis, etc.).

Both goals are further supported by the **Relationships** package, which augments the Definitions package with metadata appropriate for a repository or catalog of patterns. This metadata offers a set of semantic relationships between pattern definitions and instances, enhancing searchability and other use cases appropriate to the domain. Again, both human-oriented and machine-oriented use cases are supported in this package.

The **Formalisms** package supports the first goal more thoroughly for automated tool use cases and research purposes. It provides a mechanism for linking to a variety of formal metamodels such as Object Constraint Language (OCL), Knowledge Domain Metamodel (KDM), Abstract Syntax Tree Metamodel (ASTM), or Pattern Hierarchical Object Relation Metamodel Language (PHORML), depending on the needs of the modeler and community.

The third goal is supported by the **Pattern Instance Notation (PIN)** metamodel, which defines a common metamodel for the graphical depiction of pattern instances. It relies on the abstractions defined in SPMS. PIN and the corresponding elements in SPMS are equivalent in their expressive power, and have a one-to-one coherence of features.

PIN was developed hand in hand with the Patterns package of SPMS and provides a simple and human-oriented approach for quickly depicting instances of patterns, how they work in concert, and how they are expressed in an implementation or further design document. Most notably, PIN can be used entirely by itself to illustrate pattern interactions independent of an implementation, or used as an annotation with the variety of other graphical notations,

© ISO/IEC 2026 – All rights reserved

such as UML diagrams.

2 Conformance

The principle goal of SPMS is the exchange of definitions, descriptions, and depictions of software patterns and related abstractions in software. To be SPMS compliant, a tool must completely support the normative SPMS model elements listed in this document as Required, which currently are contained within the Definitions package. The Observations package (Clause 9) is normative, but optional, intended to support reporting instance of patterns. The Relationships package (Clause 10) is normative, but optional, intended for use in repositories or catalogs. The Formalisms package (Clause 11) is normative, but optional, intended to support automated analysis tools. The PIN metamodel (Clause 12) is normative, a tool shall support a graphical notation. PHORML (Clauses 13-16) is informative only.

An implementation shall further provide:

- The capability to generate XMI documents based on the SPMS XMI schema capturing a tool's representation of the instance model of existing patterns within a software system.

- The capability to import pattern models via representations based on the SPMS XMI schema and to map the pattern object model into the existing model of the tool.

3 Normative References

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- OMG Specification formal/2017-12-05, Unified Modeling Language (UML), v2.5.1
- OMG Specification formal/2016-11-01, Meta Object Facility (MOF), v2.5.1
- OMG Specification formal/2015-06-01, Diagram Definition (DD), v1.1

4 Terms and Definitions

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

4.1
pattern
recurring solution to a problem within a particular context of forces and constraints

Note 1 to entry: A 'software pattern', as commonly accepted in the existing literature, such as in *Design Patterns*, Gamma et. al. This document does not distinguish between ‘design patterns’, ‘architecture patterns’ or other types of patterns, but instead focuses on the needs and requirements common to all patterns communities.

4.2
pattern specification
canonical specification of a pattern expressed as human-oriented prose

4.3
pattern implementation
embodiment of a pattern specification, as it appears in an implemented system

4.4
pattern instance
single occurrence of a pattern within a pattern implementation

© ISO/IEC 2026 – All rights reserved

Note 1 to entry: One pattern implementation may give rise to many pattern instances of the same pattern specification.

4.5
pattern description
informal expression of a pattern consisting of the name and the necessary roles

4.6
repository
collection of pattern definitions for pattern specifications intended for community sharing

4.7
catalog
collection of pattern definitions for pattern specifications not necessarily intended for community sharing

Note 1 to entry: While a repository is specifically designed for sharing within a community, a catalog is a collection that may be used for other purposes, perhaps internal to an automated tool. A repository may be considered a specialized expression of a catalog.

5 Additional Information

5.1 Acknowledgements

The following companies submitted this specification:

- The Software Revolution, Inc.
- CAST
- KDM Analytics

The following persons were members of the core team that designed and wrote this specification: Jason McC. Smith (TSRI); Razak Ellafi, Camal Tazine, Bill Curtis (CAST); Nikolai Mansourov, Djenana Campara (KDM Analytics); Alain Picard, Stéphane Vaucher (Benchmark Consulting); Bob Martin, Sean Barnum (MITRE); Yann-Gaël Guéhéneuc (École Polytechnique de Montréal) and Maged Elaasar (Crossplatform Software, Inc).

The following companies supported this specification:

- TSG Consulting, Inc
- Benchmark Consulting
- MITRE
- eCube Systems
- University of North Carolina at Chapel Hill
- École Polytechnique de Montréal

© ISO/IEC 2026 – All rights reserved