INTERNATIONAL STANDARD

ISO/IEC 19507

First edition
2012-04-15

Information technology — Object Management Group Object Constraint Language (OCL)

Technologies de l'information — Langage de contraintes orienté-objet (OCL) de l'OMG

Reference number
ISO/IEC 19507:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Table of Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19507 was prepared by Technical Committee ISO/IEC JTC1, Information technology, in collaboration with the Object Management Group (OMG), following the submission and processing as a Publicly Available Specification (PAS) of the OMG Object Constraint Language specification Version 2.3.1.

ISO/IEC 19507, under the general title *Information technology - Open distributed processing - Object Constraint Language specification (OCL)*, apart from this introductory material is identical with that for the OMG specification for Object Constraint Language, v2.3.1.

Introduction

The rapid growth of distributed processing has led to a need for a coordinating framework for this standardization and ITU-T Recommendations X.901-904 | ISO/IEC 10746, the Reference Model of Open Distributed Processing (RM-ODP) provides such a framework. It defines an architecture within which support of distribution, interoperability, and portability can be integrated.

RM-ODP Part 2 (ISO/IEC 10746-2) defines the foundational concepts and modeling framework for describing distributed systems. The scopes and objectives of the RM-ODP Part 2 and the UML, while related, are not the same and, in a number of cases, the RM-ODP Part 2 and the UML specification use the same term for concepts that are related but not identical (e.g., interface). Nevertheless, a specification using the Part 2 modeling concepts can be expressed using UML with appropriate extensions (using stereotypes, tags, and constraints).

RM-ODP Part 3 (ISO/IEC 10746-3) specifies a generic architecture of open distributed systems, expressed using the foundational concepts and framework defined in Part 2. Given the relation between UML as a modeling language and Part 2 of the RM ODP standard, it is easy to show that UML is suitable as a notation for the individual viewpoint specifications defined by the RM-ODP.

OCL Language

OCL is a pure specification language; therefore, an OCL expression is guaranteed to be without side effects. When an OCL expression is evaluated, it simply returns a value. It cannot change anything in the model. This means that the state of the system will never change because of the evaluation of an OCL expression, even though an OCL expression can be used to *specify* a state change (e.g., in a post-condition).

OCL is not a programming language; therefore, it is not possible to write program logic or flow control in OCL. You cannot invoke processes or activate non-query operations within OCL. Because OCL is a modeling language in the first place, OCL expressions are not by definition directly executable.

OCL is a typed language so that each OCL expression has a type. To be well formed, an OCL expression must conform to the type conformance rules of the language. For example, you cannot compare an Integer with a String. Each Classifier defined within a UML model represents a distinct OCL type. In addition, OCL includes a set of supplementary predefined types. These are described in Clause 11 (“The OCL Standard Library”).

Information technology - Object Management Group Object Constraint Language (OCL)

1 Scope

This International Standard defines the Object Constraint Language (OCL), version 2.3.1. OCL version 2.3.1 is the version of OCL that is aligned with UML 2.3 and MOF 2.0.

2 Conformance

The UML 2.0 Infrastructure and the MOF 2.0 Core specifications that were developed in parallel with this OCL 2.3.1 specification share a common core. The OCL specification contains a well-defined and named subset of OCL that is defined purely based on the common core of UML and MOF. This allows this subset of OCL to be used with both the MOF and the UML, while the full specification can be used with the UML only.

The following compliance points are distinguished for both parts.

1. Syntax compliance: The tool can read and write OCL expressions in accordance with the grammar, including validating its type conformance and conformance of well-formedness rules against a model.
2. XMI compliance: The tool can exchange OCL expressions using XMI.
3. Evaluation compliance: The tool evaluates OCL expressions in accordance with the semantics clause. The following additional compliance points are optional for OCL evaluators, as they are dependent on the technical platform on which they are evaluated:
   - allInstances()
   - pre-values and oclIsNew() in postconditions
   - OclMessage
   - navigating across non-navigable associations
   - accessing private and protected features of an object

The following table shows the possible compliance points. Each tool is expected to fill in this table to specify which compliance points are supported.

Table 2.1 - Overview of OCL Compliance Points

	OCL-MOF subset	**Full OCL**
Syntax		
XMI		
Evaluation		
- allInstances		
- @pre in postcondtions		
- OclMessage		
- navigating non-navigable associations		
- accessing private and protected features		

3 References

3.1 Normative References

The following referenced documents are indispensable for the application 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 639 Codes for the representation of names of languages
- ISO 3166 Codes for the representation of names of countries and their subdivisions
- ISO/IEC 10646:2003 Information technology -- Universal Multiple-Octet Coded Character Set (UCS)
- UML 2.3 Superstructure Specification: http://www.omg.org/spec/UML/2.3/Superstructure/PDF/
- UML 2.3 Infrastructure Specification: http://www.omg.org/soec/UML/2.3/Infrastructure/PDF/
- MOF 2.0 Core Specification: http://www.omg.org/spec/MOF/2.0/PDF/
- UNICODE 5.1 Standard: http://www.unicode.org/versions/Unicode5.1.0/
- Unicode Technical Standard#10: http://www.unicode.org/reports/tr10/

3.2 Informative References

The following specifications are referenced in informative text:

- ISO/IEC 19501:2005 Information technology - Open Distributed Processing -- Unified Modeling Language (UML) Version 1.4.2

4 Terms and Definitions

There are no formal definitions that are taken from other documents.

5 Notational Conventions

There are no symbols defined.

6 Additional Information

6.1 Changes to Adopted OMG Specifications

This International Standard replaces the specification of OCL given in OCL 2.2.

The version of OCL specified in ISO/IEC 19501:2005 in intended for use in models based on UML 1.4.1 and UML 1.5. However, use of the OCL specified by ISO/IEC 19501:2005 is not prescribed by this specification.

The version of OCL specified in this International Standard is not directly applicable to models based on ISO/IEC 19501:2005.

6.2 Structure of the Specification

This International Standard is divided into several clauses.

- The OCL Language Description clause gives an informal description of OCL. This clause is not normative, but meant to be explanatory.
- Clause 8 (“Abstract Syntax”) describes the abstract syntax of OCL using a MOF 2.0 compliant metamodel. This is the same approach as used in the UML specifications. The metamodel is MOF compliant in the sense that it only uses constructs that are defined in the MOF.
- Clause 9 (“Concrete Syntax”) describes the canonical concrete syntax using an attributed EBNF grammar. This syntax is mapped onto the abstract syntax, achieving a complete separation between concrete and abstract syntax.
- Clause 10 (“Semantics Described using UML”) describes the semantics for OCL using UML.
- In Clause 11 (“The OCL Standard Library”) the OCL Standard Library is described. This defines type like Integer, Boolean, etc. and all the collection types. OCL is not a stand-alone language, but an integral part of the UML. An OCL expression needs to be placed within the context of a UML model.
- Clause 12 (“The Use of Ocl Expressions in UML Models”) describes a number of places within the UML where OCL expressions can be used.
- Clause 13 (“Basic OCL and Essential OCL”) defines the adaptation of the OCL metamodel when used in particular context of Core::Basic infrastructure library package and in the context of EMOF.
- Annex A (Semantics), Annex B (Bibliography), and Annex C (Legal Information)

6.3 Acknowledgements

The following companies submitted and/or supported parts of this specification:

- Adaptive Ltd.
- BoldSoft
- Borland Software Corporation
- Compuware Corporation
- Dresden University of Technology
- France Telecom
- International Business Machines
- IONA
- Kabira Technologies Inc.
- Kings College
- Classe Objecten
- Open Canarias, SL
- Oracle
- Project Technology Inc.
- Rational Software Corporation
- SAP AG
- Softeam
- Syntropy Ltd.
- Telelogic
- Thales
- University of Bremen
- University of Kent
- University of York
- Zeligsoft, Inc.

7 OCL Language Description

7.1 General

This clause introduces the Object Constraint Language (OCL), a formal language used to describe expressions on UML models. These expressions typically specify invariant conditions that must hold for the system being modeled or queries over objects described in a model. Note that when the OCL expressions are evaluated, they do not have side effects (i.e., their evaluation cannot alter the state of the corresponding executing system).

OCL expressions can be used to specify operations / actions that, when executed, do alter the state of the system. UML modelers can use OCL to specify application-specific constraints in their models. UML modelers can also use OCL to specify queries on the UML model, which are completely programming language independent.

Note - This clause is informative only and not normative.

7.2 Why OCL?

A UML diagram, such as a class diagram, is typically not refined enough to provide all the relevant aspects of a specification. There is, among other things, a need to describe additional constraints about the objects in the model. Such constraints are often described in natural language. Practice has shown that this will always result in ambiguities. In order to write unambiguous constraints, so-called formal languages have been developed. The disadvantage of traditional formal languages is that they are usable to persons with a strong mathematical background, but difficult for the average business or system modeler to use.

OCL has been developed to fill this gap. It is a formal language that remains easy to read and write. It has been developed as a business modeling language within the IBM Insurance division, and has its roots in the Syntropy method.

OCL is a pure specification language; therefore, an OCL expression is guaranteed to be without side effects. When an OCL expression is evaluated, it simply returns a value. It cannot change anything in the model. This means that the state of the system will never change because of the evaluation of an OCL expression, even though an OCL expression can be used to *specify* a state change (e.g., in a post-condition).

OCL is not a programming language; therefore, it is not possible to write program logic or flow control in OCL. You cannot invoke processes or activate non-query operations within OCL. Because OCL is a modeling language in the first place, OCL expressions are not by definition directly executable.

OCL is a typed language so that each OCL expression has a type. To be well formed, an OCL expression must conform to the type conformance rules of the language. For example, you cannot compare an Integer with a String. Each Classifier defined within a UML model represents a distinct OCL type. In addition, OCL includes a set of supplementary predefined types. These are described in Clause 11 (“The OCL Standard Library”).

As a specification language, all implementation issues are out of scope and cannot be expressed in OCL.

The evaluation of an OCL expression is instantaneous. This means that the states of objects in a model cannot change during evaluation.

7.2.1 Where to Use OCL

OCL can be used for a number of different purposes:

- as a query language,