
**Information technology — XML Metadata
Interchange (XMI)**

Technologies de l'information — Échange de métadonnées XML (XMI)

iTeh STANDARD PREVIEW
(standards.iteh.ai)

[ISO/IEC 19503:2005](https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005)

<https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005>

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

ISO/IEC 19503:2005

<https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005>

© ISO/IEC 2005

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

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

Contents

Foreword	viii
Introduction	ix
1 Scope	1
2 Normative references	1
2.1 Identical Recommendations International Standards	1
2.2 International Standards	1
3 Abbreviations	2
4 XMI Schema Design Principles	3
4.1 Purpose	3
4.2 Use of XML Schemas	3
4.2.1 XML Validation of XMI documents	4
4.2.2 Requirements for XMI Schemas	4
4.3 Basic Principles	4
4.3.1 Required XML Declarations	4
4.3.2 Metamodel Class Representation	5
4.3.3 Metamodel Extension Mechanism	5
4.4 XMI Schema and Document Structure	5
4.5 XMI Model	6
4.5.1 XML Schema for the XMI Model	6
4.5.2 XMI Model Classes	6
4.5.3 XMI	8
4.5.4 Extension	9
4.5.5 Documentation	10
4.5.6 Add, Replace, and Delete	10
4.6 XMI Attributes	12
4.6.1 Element Identification Attributes	12
4.6.2 Linking Attributes	13
4.6.3 Version Attribute	14
4.6.4 Type Attribute	14
4.7 XMI Type	14
4.8 Metamodel Class Specification	15
4.8.1 Namespace Qualified XML Element Names	15
4.8.2 Metamodel Multiplicities	15
4.8.3 Class Specification	16
4.8.4 Attribute Specification	16
4.8.5 Reference Specification	18
4.8.6 Containment Specification	18
4.8.7 Inheritance Specification	18
4.8.8 Derived Information	18
4.9 Transmitting Incomplete Metadata	18
4.9.1 Interchange of Model Fragments	19
4.9.2 XMI Encoding	19
4.9.3 Example	19

4.10	Linking	19
4.10.1	Design Principles	19
4.10.2	Linking	20
4.10.3	Example from UML	22
4.11	Tailoring Schema Production	23
4.11.1	XMI Tag Values.....	23
4.11.2	Tag Value Constraints	24
4.11.3	Scope	25
4.11.4	XML element vs XML attribute	25
4.11.5	UML profile for XML and XMI	25
4.11.6	Effects on Document Production	26
4.11.7	Summary of XMI Tag Scope and Affect	27
4.12	Transmitting Metadata Differences	28
4.12.1	Definitions	29
4.12.2	Differences	29
4.12.3	XMI Encoding	29
4.12.4	Example	30
4.13	Document Exchange with Multiple Tools	31
4.13.1	Definitions	31
4.13.2	Procedures	32
4.13.3	Example	32
4.14	General Datatype Mechanism	33
5	XML Schema Production.....	35
5.1	Purpose	35
5.2	XMI Version 2 Schemas.....	35
5.2.1	EBNF	35
5.2.2	Fixed Schema Declarations	43
5.2.3	Schema Production Rules for Non-Primitive Data	48
6	XML Document Production	49
6.1	Purpose	49
6.2	Introduction	49
6.3	EBNF Rules Representation	49
6.3.1	Overall Document Structure	50
6.3.2	Overall Content Structure	51
6.3.3	Object Structure	52
6.3.4	References.....	55
6.3.5	Object Contents	57
6.3.6	Packages	59
6.3.7	Attributes	59
6.3.8	Other Types of Links	60
6.4	Additional Examples	60
6.4.1	Inheritance	60
6.4.2	Nested Packages	61
6.4.3	Derived Types and References	62
6.5	Document Production Rules for Non-Primitive Data	63
6.5.1	Structure Type	63
6.5.2	Enumeration Type.....	63
6.5.3	Alias Type	63

6.5.4 Collection Type	63
7 Production of MOF from XML.....	65
7.1 Introduction	65
7.2 DTD to MOF	65
7.3 XML to MOF	66
7.4 XML Schema to MOF	67
8 XML Schema Model	71
8.1 Introduction	71
8.2 XML Schema Structures	71
8.2.1 XSDSchema	83
8.2.2 XSDAttribute	83
8.2.3 XSDElementRef	83
8.2.4 XSDAttributeGroup	83
8.2.5 XSDAttributeGroupRef	83
8.2.6 XSDType	83
8.2.7 XSDBuiltInType	84
8.2.8 XSDComplexType	84
8.2.9 XSDComplexTypeContent	84
8.2.10 XSDSchemaContent	84
8.2.11 XSDElement	84
8.2.12 XSDSimpleBase	85
8.2.13 XSDPattern	85
8.2.14 XSDEnumeration	85
8.2.15 XSDInclude	85
8.2.16 XSDImport	85
8.2.17 XSDGroup	85
8.2.18 XSDGroupKind	86
8.2.19 XSDGroupScope	86
8.2.20 XSDGroupContent	86
8.2.21 XSDGroupRef	86
8.2.22 XSDKey	86
8.2.23 XSDKeyRef	86
8.2.24 XSDUnique	86
8.2.25 XSDUniqueContent	87
8.2.26 XSDSelector	87
8.2.27 XSDField	87
8.2.28 XSDOBJECT	87
8.2.29 XSDAnnotatedElement	87
8.2.30 XSDDocumentation	87
8.2.31 XSDAppInfo	87
8.2.32 XSDAnnotation	87
8.2.33 XSDSimpleContent	88
8.2.34 XSDComplexContent	88
8.2.35 XSDSimpleComplex	88
8.2.36 XSDSimpleTypeContent	88
8.2.37 XSDSimpleRestrict	88
8.2.38 XSDSimpleList	88
8.2.39 XSDSimpleUnion	88

8.2.40	XSDSimpleType	89
8.2.41	XSDFacet	89
8.2.42	XSDLength	89
8.2.43	XSDMinLength	89
8.2.44	XSDMaxLength	89
8.2.45	XSDMinInclusive	89
8.2.46	XSDMaxInclusive	89
8.2.47	XSDMinExclusive	90
8.2.48	XSDMaxExclusive	90
8.2.49	XSDTotalDigits	90
8.2.50	XSDFractionDigits	90
8.2.51	XSDWhiteSpace	90
8.2.52	XSDAny	90
8.2.53	XSDAnyAttribute	90
8.2.54	XSDAttributeRef	91
8.2.55	XSDNamedElement	91
8.2.56	XSDOccurs	91
8.2.57	XSDTopLevelAttribute	91
8.2.58	XSDTopLevelElement	91
8.3	XML Schema Simple Datatypes	91
8.3.1	XSDDate	95
8.3.2	XSDDecimal	95
8.3.3	XSDDecimalType	95
8.3.4	XSDDouble	96
8.3.5	XSDCentury	96
8.3.6	XSDBinary	96
8.3.7	XSDBinaryType	96
8.3.8	XSDBooleanType	96
8.3.9	XSDBoolean	96
8.3.10	XSDByte	96
8.3.11	XSDDoubleType	97
8.3.12	XSDFloat	97
8.3.13	XSDFloatType	97
8.3.14	XSDInt	97
8.3.15	XSDInteger	97
8.3.16	XSDCDATA	97
8.3.17	XSDID	97
8.3.18	XSDIDREF	97
8.3.19	XSDIDREFS	98
8.3.20	XSDListType	98
8.3.21	XSDList	98
8.3.22	XSDLong	98
8.3.23	XSDMonth	98
8.3.24	XSDName	98
8.3.25	XSDNCName	98
8.3.26	XSDNegativeInteger	98
8.3.27	XSDNMTOKEN	99
8.3.28	XSDNonNegativeInteger	99
8.3.29	XSDNonPositiveInteger	99
8.3.30	XSDPositiveInteger	99
8.3.31	XSDQName	99

8.3.32 XSD QNameType	99
8.3.33 XSD RecurringDate	99
8.3.34 XSD RecurringDay	100
8.3.35 XSD RecurringDuration	100
8.3.36 XSD RecurringDurationType	100
8.3.37 XSD Short	100
8.3.38 XSD Token	100
8.3.39 XSD String	100
8.3.40 XSD StringType	100
8.3.41 XSD Time	101
8.3.42 XSD TimeDuration	101
8.3.43 XSD TimeDurationType	101
8.3.44 XSD TimeInstant	101
8.3.45 XSD TimePeriod	101
8.3.46 XSD UnionType	101
8.3.47 XSD UnsignedByte	101
8.3.48 XSD UnsignedInt	102
8.3.49 XSD UnsignedLong	102
8.3.50 XSD UnsignedShort	102
8.3.51 XSD URIReference	102
8.3.52 XSD URIRestrictedType	102
8.3.53 XSD ValueConstraint	102
8.3.54 XSD Year	102
8.3.55 XSD DecimalRange	103
8.3.56 XSD IntegerRange	103
8.3.57 XSD Patterned	103
Annex A (normative) Conformance Issues	105
Annex B (normative) References	107
Annex C (informative) Legal Information	109
Index	113

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.

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

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national 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 and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19503 was prepared by the Object Management Group (OMG) and was adopted, under the PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

ISO/IEC 19503 is related to

- ISO/IEC 19501, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*
- ISO/IEC 19502, *Information technology — Meta Object Facility (MOF)*

Introduction

The main purpose of this International Standard (XML) is to enable easy interchange of metadata between application development lifecycle tools (such as modeling tools based on the Unified Modeling Language (UML), ISO/IEC 19501, and metadata repositories/frameworks based on the Meta Object Facility (MOF), ISO/IEC 19502) in distributed heterogeneous environments. This International Standard integrates three key industry standards:

- XML - eXtensible Markup Language, a W3C standard.
- UML - Unified Modeling Language, an OMG modeling specification, which is now ISO/IEC 19501.
- MOF - Meta Object Facility (ISO/IEC 19502).

The OMG adopted the XMI (version 1.0) in February 1999. It was developed as a response to a request for proposal, issued by the OMG Analysis and Design Task Force, for a model and metadata interchange facility. The purpose of the facility was to support the interchange of metadata (such as ODP UML models). The most recent revision of XMI, 2.0, was submitted by the XMI Revision Task Force in October, 2002, and includes corrections and clarifications to the original specification, and changes to accommodate revisions to the 1.4 version of MOF.

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, *Open Distributed Processing — Reference Model (RM-ODP)* provides such a framework. It defines an architecture in 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. 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.

While not limited to this context, this International Standard is relevant to work on the standardization of Open Distributed Processing (ODP).

iTeh STANDARD PREVIEW **(standards.iteh.ai)**

ISO/IEC 19503:2005

<https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005>

Information technology — XML Metadata Interchange (XMI)

1 Scope

This International Standard provides specifications for:

- a. A set of XML Schema Definitions (XSD) production rules for transforming MOF based metamodels into XML Schemas.
- b. A set of XML Document production rules for encoding and decoding MOF based metadata.
- c. Design principles for XMI based Schemas and XML documents.
- d. A set of production rules for importing XML DTDs to a MOF based metamodel.

This International Standard enhances metadata management and metadata interoperability in distributed object environments in general and in distributed development environments in particular. While this International Standard addresses stream based metadata interoperability in the object analysis and design domain, XMI (in part because it is MOF based) is equally applicable to metadata in many other domains.

2 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.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation X.902 (1996) | ISO/IEC 10746-2:1996, *Information technology — Open Distributed Processing — Reference Model: Foundations*
- ITU-T Recommendation X.903 (1996) | ISO/IEC 10746-3:1996, *Information technology — Open Distributed Processing — Reference Model: Architecture*

2.2 International Standards

- ISO/IEC 10646:2003, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)*
- ISO/IEC 19501, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*
- ISO/IEC 19502, *Information technology — Meta Object Facility (MOF)*
- W3C XML 1.0 : <http://www.w3.org/TR/REC-xml> – February, 2004
- W3C XSD 1.0 <http://www.w3.org/TR/xmlschema-0/>, [xmlschema-1](http://www.w3.org/TR/xmlschema-1/), [xmlschema-2](http://www.w3.org/TR/xmlschema-2/)

3 Abbreviations

DTD	Document Type Definition
MOF	Meta Object Facility
UML	Unified Modeling Language
XMI	XML Metadata Interchange
XSD	XML Schema Definition

iTeh STANDARD PREVIEW
(standards.iteh.ai)

[ISO/IEC 19503:2005](https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005)

<https://standards.iteh.ai/catalog/standards/sist/f8184423-dc58-4cb4-bea5-8318b887d861/iso-iec-19503-2005>

4 XMI Schema Design Principles

4.1 Purpose

This Clause contains a description of the XML Schemas that may be used with the XMI specification to allow some metamodel information to be verified through XML validation. The use of schemas in XMI is described first, followed by a brief description of some basic principles, which includes a short description of each XML attribute and XML element defined by XMI. Those descriptions are followed by more complete descriptions that provide examples illustrating the motivation for the XMI schema design in the areas of metamodel class specification, transmitting incomplete metadata, linking, tailoring schema production, transmitting metadata differences, and exchanging documents between tools.

It is possible to define how to automatically generate a schema from the MOF metamodel to represent any MOF-compliant metamodel. That definition is presented in Clause 5.

This Clause describes XMI 2.0 schemas; Clause 5 describes how to create XMI 2.0 schemas.

You may specify tag value pairs as part of the MOF metamodel to tailor the schemas that are generated, but you are not required to do so. Using these tag value pairs requires some knowledge of XML schemas, but the schemas that are produced might perform more validation than the default schemas. See Clause 7 for a complete description of how to generate XML schemas using these tag value pairs. Sub clause 4.11, "Tailoring Schema Production," on page 23 describes the tag values, their affect on schema production, and their impact on document serialization.

4.2 Use of XML Schemas

An XML schema provides a means by which an XML processor can validate the syntax and some of the semantics of an XML document. This specification provides rules by which a schema can be generated for any valid XMI-transmissible MOF-based metamodel. However, the use of schemas is optional; an XML document need not reference a schema, even if one exists. The resulting document can be processed more quickly, at the cost of some loss of confidence in the quality of the document.

It can be advantageous to perform XML validation on the XML document containing MOF metamodel data. If XML validation is performed, any XML processor can perform some verification, relieving import/export programs of the burden of performing these checks. It is expected that the software program that performs verification will not be able to rely solely on XML validation for all of the verification, however, since XML validation does not perform all of the verification that could be done.

Each XML document that contains metamodel data conforming to this specification contains: XML elements that are required by this specification, XML elements that contain data that conform to a metamodel, and, optionally, XML elements that contain metadata that represent extensions of the metamodel. Metamodels are explicitly identified in XML elements required by this specification. Some metamodel information can also be encoded in an XML schema. Performing XML validation provides useful checking of the XML elements that contain metadata about the information transferred, the transfer information itself, and any extensions to the metamodel.

The XML Namespace specification has been adopted by the W3C, allowing XMI to use multiple metamodels at the same time. XML schema validation works with XML namespaces, so you can choose your own namespace prefixes in an XML document and use a schema to validate it. The namespace URIs, not the namespace prefixes, are used to identify which schemas to use to validate an XML document.

4.2.1 XML Validation of XMI documents

XML validation can determine whether the XML elements required by this specification are present in the XML document containing metamodel data, whether XML attributes that are required in these XML elements have values for them, and whether some of the values are correct.

XML validation can also perform some verification that the metamodel data conforms to a metamodel. Although some checking can be done, it is impossible to rely solely on XML validation to verify that the information transferred satisfies all of a metamodel's semantic constraints. Complete verification cannot be done through XML validation because it is not currently possible to specify all of the semantic constraints for a metamodel in an XML schema, and the rules for automatic generation of a schema preclude the use of semantic constraints that could be encoded in a schema manually, but cannot be automatically encoded.

Finally, XML validation can be used to validate extensions to the metamodel, because extensions must be represented as elements; if those elements are defined in a schema, the schema can be used to verify the elements.

4.2.2 Requirements for XMI Schemas

Each schema used by XMI must satisfy the following requirements:

- All XML elements and attributes defined by the XMI specification must be imported in the schema. They cannot be put directly in the schema itself, since there is only one target namespace per schema.
- Metamodel constructs have corresponding element declarations, and may have an XML attribute declaration, as described below. In addition, some constructs also have a complexType declaration. The declarations may utilize groups, attribute groups, and types, as described below.
- Any XML elements that represent extensions to the metamodel may be declared in a schema, although it is not necessary to do so.

By default, XMI schemas allow incomplete metadata to be transmitted, but you can enforce the lower bound of multiplicities if you wish. See 4.9, "Transmitting Incomplete Metadata," on page 18 for further details.

4.3 Basic Principles

This sub clause discusses the basic organization of an XML schema for XMI. Detailed information about each of these topics is included later in this Clause.

4.3.1 Required XML Declarations

This specification requires that XML element declarations, types, attributes, and attribute groups be included in schemas to enable XML validation of metadata that conforms to this specification. Some of these XML elements contain metadata about the metadata to be transferred. For example, the identity of the metamodel associated with the metadata, the tool that generated the metadata, whether the metadata has been verified, etc.

All XML elements defined by this specification are in the namespace "http://www.omg.org/XMI." The XML namespace mechanism can be used to avoid name conflicts between the XMI elements and the XML elements from your MOF models.

In addition to required XML element declarations, there are some attributes that must be defined according to this specification. Every XML element that corresponds to a metamodel class must have XML attributes that enable the XML element to act as a proxy for a local or remote XML element. These attributes are used to associate an XML element with another XML element. There are also other required attributes to let you put data in XML attributes rather than XML elements. You may customize the declarations using MOF tag values.

4.3.2 Metamodel Class Representation

Every metamodel class is represented in the schema by an XML element whose name is the class name, as well as a `complexType` whose name is the class name. The declaration of the type lists the attributes of the class; references to association ends relating to the class; and the classes that this class contains, either explicitly or through composition associations. By default, the content models of XML elements corresponding to metamodel classes do not impose an order on the attributes and references.

By default, XMI allows you to serialize features using either XML elements or XML attributes; however, XMI allows you to specify how to serialize them if you wish. Containment references and multivalued attributes always are serialized using XML elements.

4.3.3 Metamodel Extension Mechanism

Every XMI schema contains a mechanism for extending a metamodel class. Zero or more **extension** elements are included in the content model of each class. These extension elements have a content model of ANY, allowing considerable freedom in the nature of the extensions. The `processContents` attribute is `lax`, which means that processors will validate the elements in the extension if a schema is available for them, but will not report an error if there is no schema for them. In addition, the top level XMI element may contain zero or more **extension** elements, which provides for the inclusion of any new information. One use of the extension mechanism might be to associate display information for a particular tool with the metamodel class represented by the XML element. Another use might be to transmit data that represents extensions to a metamodel.

Tools that rely on XMI are expected to store the extension information and export it again to enable round trip engineering, even though it is unlikely they will be able to process it further. XML elements that are put in the **extension** elements may be declared in schemas, but are not required to be.

4.4 XMI Schema and Document Structure

Every XMI schema consists of the following declarations:

- An XML version processing instruction. Example: `<? XML version="1.0" ?>`
- An optional encoding declaration that specifies the character set, which follows the ISO-10646 (also called extended Unicode) standard.
Example: `<? XML version="1.0" ENCODING="UCS-2" ?>`.
- Any other valid XML processing instructions.
- A schema XML element.
- An import XML element for the XMI namespace.
- Declarations for a specific metamodel.

Every XMI document consists of the following declarations, unless the XMI is embedded in another XML document: