
**Information technology — Meta Object
Facility (MOF)**

Technologies de l'information — Facilité d'objet «méta» (MOF)

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.

© 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 and Conventions	2
4 List of Documents	2
5 MOF Usage Scenarios	3
5.1 Overview	3
5.2 Software Development Scenarios	4
5.3 Type Management Scenarios	5
5.4 Information Management Scenarios	6
5.5 Data Warehouse Management Scenarios	7
6 MOF Conceptual Overview	9
6.1 Overview	9
6.2 Metadata Architectures	9
6.2.1 Four Layer Metadata Architectures	9
6.2.2 The MOF Metadata Architecture	10
6.2.3 MOF Metamodeling Terminology	12
6.3 The MOF Model - Metamodeling Constructs	13
6.3.1 Classes	13
6.3.2 Associations	16
6.3.3 Aggregation	17
6.3.4 References	18
6.3.5 DataTypes	20
6.3.6 Packages	20
6.3.7 Constraints and Consistency	23
6.3.8 Miscellaneous Metamodeling Constructs	24
6.4 Metamodels and Mappings	25
6.4.1 Abstract and Concrete Mappings	25
6.4.2 The MOF Metamodel to IDL Mapping	26
6.4.3 The MOF Metamodel to XML Mapping	26
6.4.4 Mappings of the MOF Model	27
7 MOF Model and Interfaces	29
7.1 Overview	29
7.2 How the MOF Model is Described	29
7.2.1 Classes	30

7.2.2 Associations	34
7.2.3 DataTypes	35
7.2.4 Exceptions	35
7.2.5 Constants	36
7.2.6 Constraints	36
7.2.7 UML Diagrams	36
7.3 The Structure of the MOF Model	36
7.3.1 The MOF Model Package	36
7.3.2 The MOF Model Service IDL	38
7.3.3 The MOF Model Structure	38
7.3.4 The MOF Model Containment Hierarchy	40
7.4 MOF Model Classes	41
7.4.1 ModelElement	(abstract) 41
7.4.2 Namespace	(abstract) 45
7.4.3 GeneralizableElement	(abstract) 48
7.4.4 TypedElement	(abstract) 52
7.4.5 Classifier	(abstract) 53
7.4.6 Class	54
7.4.7 DataType	(abstract) 55
7.4.8 PrimitiveType	56
7.4.9 CollectionType	57
7.4.10 EnumerationType	58
7.4.11 AliasType	59
7.4.12 StructureType	59
7.4.13 StructureField	60
7.4.14 Feature	(abstract) 60
7.4.15 StructuralFeature	(abstract) 62
7.4.16 Attribute	(idl_substitute_name "MofAttribute") 63
7.4.17 Reference	64
7.4.18 BehavioralFeature	(abstract) 66
7.4.19 Operation	67
7.4.20 Exception	(idl_substitute_name "MofException") 68
7.4.21 Association	69
7.4.22 AssociationEnd	71
7.4.23 Package	74
7.4.24 Import	76
7.4.25 Parameter	78
7.4.26 Constraint	79
7.4.27 Constant	82
7.4.28 Tag	83
7.5 MOF Model Associations	85
7.5.1 Contains	85
7.5.2 Generalizes	86
7.5.3 RefersTo	87
7.5.4 Exposes	(derived) 88
7.5.5 IsOfType	90
7.5.6 CanRaise	90
7.5.7 Aliases	91
7.5.8 Constrains	92
7.5.9 DependsOn	(derived) 93

7.5.10 AttachesTo	95
7.6 MOF Model Data Types	96
7.6.1 PrimitiveTypes used in the MOF Model	96
7.6.2 MultiplicityType.....	96
7.6.3 VisibilityKind.....	97
7.6.4 DirectionKind.....	98
7.6.5 ScopeKind	98
7.6.6 AggregationKind	98
7.6.7 EvaluationKind	98
7.7 MOF Model Exceptions	99
7.7.1 NameNotFound.....	99
7.7.2 NameNotResolved	99
7.8 MOF Model Constants	99
7.8.1 Unbounded	100
7.8.2 The Standard DependencyKinds	100
7.9 MOF Model Constraints	101
7.9.1 MOF Model Constraints and other M2 Level Semantics	101
7.9.2 Notational Conventions	101
7.9.3 OCL Usage in the MOF Model specification	103
7.9.4 The MOF Model Constraints	105
7.9.5 Semantic specifications for some Operations, derived Attributes and Derived Associations	125
7.9.6 OCL Helper functions	131
7.10 The PrimitiveTypes Package	134
7.10.1 Boolean.....	135
7.10.2 Integer	135
7.10.3 Long	135
7.10.4 Float.....	135
7.10.5 Double	135
7.10.6 String	135
7.10.7 IDL for the PrimitiveTypes Package	136
7.11 Standard Technology Neutral Tags	136
8 The MOF Abstract Mapping.....	139
8.1 Overview	139
8.2 MOF Values	139
8.3 Semantics of Data Types	139
8.4 Semantics of Equality for MOF Values	140
8.5 Semantics of Class Instances	141
8.6 Semantics of Attributes	141
8.6.1 Attribute name and type	142
8.6.2 Multiplicity	142
8.6.3 Scope	143
8.6.4 Is_derived	144
8.6.5 Aggregation.....	144
8.6.6 Visibility and is_changeable.....	144
8.7 Package Composition	144
8.7.1 Package Nesting	144
8.7.2 Package Generalization	145
8.7.3 Package Importation	145

8.7.4 Package Clustering	145
8.8 Extents	145
8.8.1 The Purpose of Extents	146
8.8.2 Class Extents	147
8.8.3 Association Extents	147
8.8.4 Package Extents	147
8.9 Semantics of Associations	149
8.9.1 MOF Associations in UML notation	149
8.9.2 Core Association Semantics	150
8.9.3 AssociationEnd Changeability	152
8.9.4 Association Aggregation	152
8.9.5 Derived Associations	152
8.10 Aggregation Semantics	152
8.10.1 Aggregation “none”	152
8.10.2 Aggregation “composite”	153
8.10.3 Aggregation “shared”	153
8.11 Closure Rules	153
8.11.1 The Reference Closure Rule.....	153
8.11.2 The Composition Closure Rule	155
8.12 Recommended Copy Semantics	156
8.13 Computational Semantics	157
8.13.1 A Style Guide for Metadata Computational Semantics	157
8.13.2 Access operations should not change metadata	158
8.13.3 Update operations should only change the nominated metadata	158
8.13.4 Derived Elements should behave like non-derived Elements	158
8.13.5 Constraint evaluation should not have side-effects	158
8.13.6 Access operations should avoid raising Constraint exceptions	159
9 MOF to IDL Mapping	161
9.1 Overview	161
9.2 Meta Objects and Interfaces	161
9.2.1 Meta Object Type Overview	161
9.2.2 The Meta Object Interface Hierarchy	163
9.3 Computational Semantics for the IDL Mapping	165
9.3.1 The CORBAIdl Types Package.....	165
9.3.2 Mapping of MOF Data Types to CORBA IDL Types.....	169
9.3.3 Value Types and Equality in the IDL Mapping	170
9.3.4 Lifecycle Semantics for the IDL Mapping	170
9.3.5 Association Access and Update Semantics for the IDL Mapping	173
9.3.6 Link Addition Operations	173
9.3.7 Attribute Access and Update Semantics for the IDL Mapping	176
9.3.8 Reference Semantics for the IDL Mapping	181
9.3.9 Cluster Semantics for the IDL Mapping	182
9.3.10 Atomicity Semantics for the IDL Mapping	182
9.3.11 The Supertype Closure Rule	182
9.3.12 Copy Semantics for the IDL Mapping	183
9.4 Exception Framework	183
9.4.1 Error_kind string values	185
9.4.2 Structural Errors	185
9.4.3 Constraint Errors	188

9.4.4 Semantic Errors	188
9.4.5 Usage Errors	189
9.4.6 Reflective Errors	190
9.5 Preconditions for IDL Generation	192
9.6 Standard Tags for the IDL Mapping	194
9.6.1 Tags for Specifying IDL #pragma directives	194
9.6.2 Tags for Providing Substitute Identifiers	195
9.6.3 Tags for Specifying IDL Inheritance	196
9.7 Generated IDL Issues	198
9.7.1 Generated IDL Identifiers	198
9.7.2 Generation Rules for Synthesized Collection Types	200
9.7.3 IDL Identifier Qualification	202
9.7.4 File Organization and #include statements	202
9.8 IDL Mapping Templates	202
9.8.1 Template Notation	203
9.8.2 Package Module Template	203
9.8.3 Package Factory Template	205
9.8.4 Package Template	206
9.8.5 Class Forward Declaration Template	209
9.8.6 Class Template	209
9.8.7 Class Proxy Template	210
9.8.8 Instance Template	212
9.8.9 Class Create Template	213
9.8.10 Association Template	214
9.8.11 Attribute Template	222
9.8.12 Reference Template	231
9.8.13 Operation Template	240
9.8.14 Exception Template	242
9.8.15 DataType Template	243
9.8.16 Constraint Template	245
9.8.17 Annotation Template	245
10 The Reflective Module	247
10.1 Introduction	247
10.2 The Reflective Interfaces	248
10.2.1 Reflective Argument Encoding Patterns	248
10.2.2 Reflective::RefBaseObject	(abstract) 250
10.2.3 Reflective::RefObject	(abstract) 254
10.2.4 Reflective::RefAssociation	(abstract) 265
10.2.5 Reflective::RefPackage	(abstract) 269
10.3 The CORBA IDL for the Reflective Interfaces	270
10.3.1 Introduction	270
10.3.2 Data Types	271
Annex A (normative) Conformance Issues	273
Annex B (normative) Legal Information	275
INDEX	279

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 19502 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 19502 is related to

- ISO/IEC 19501, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*
- ISO/IEC 19503, *Information technology — XML Metadata Interchange (XMI)*
- ISO/IEC 14769, *Information technology — Open Distributed Processing — Type Repository Function*

Introduction

This International Standard defines a metamodel (defined using MOF), a set of interfaces (defined using ODP IDL (ITU-T Recommendation X.920 (1997) | ISO/IEC 14750:1999), that can be used to define and manipulate a set of interoperable metamodels and their corresponding models. It also defines the mapping from MOF to ODP IDL (ITU rec X920|ISO 14750). These interoperable metamodels include the Unified Modeling Language (UML) metamodel (ISO/IEC 19501:2005), the MOF meta-metamodel, as well as future standard technologies that will be specified using metamodels. The MOF provides the infrastructure for implementing design and reuse repositories, application development tool frameworks, etc. The MOF specifies precise mapping rules that enable the CORBA interfaces for metamodels to be generated automatically, thus encouraging consistency in manipulating metadata in all phases of the distributed application development cycle. Mappings from MOF to W3C XML and XSD are specified in the XMI (ISO/IEC 19503) specification. Mappings from MOF to Java™ are in the JMI (Java Metadata Interchange) specification defined by the Java Community Process.

In order to achieve architectural alignment considerable effort has been expended so that the UML and MOF share the same core semantics. This alignment allows the MOF to reuse the UML notation for visualizing metamodels. In those areas where semantic differences are required, well-defined mapping rules are provided between the metamodels. The UML has been the subject of a separate PAS submission.

The OMG adopted the MOF (version 1.0) in November 1997. It was developed as a response to a request for proposal, issued by the OMG Analysis and Design Task Force, for Metadata repository facility (<http://www.omg.org/cgi-bin/doc?cf/96-05-02>). The purpose of the facility was to support the creation, manipulation, and interchange of meta models. The most recent revision of MOF, 1.4 was adopted in April 2002, and includes corrections and clarifications to the original 1.3 version, and minor modeling feature additions.

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 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. 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 closely related to work on the standardization of Open Distributed Processing (ODP). In particular, the ODP Type Repository Function (ISO/IEC 14769 | Rec. X.960) references the OMG Meta Object Facility, version 1.3. This function specifies how to use the OMG MOF as a repository for ODP types.

Information technology — Meta Object Facility (MOF)

1 Scope

This International Standard specifies the following:

- a. An abstract language for specifying, constructing, and managing technology neutral metamodels: A metamodel is in effect an abstract language for some kind of metadata.
- b. A framework for implementing repositories & integration frameworks (e.g., tool integration frameworks) that hold metadata (e.g., models) described by the metamodels and which uses standard technology mappings to transform MOF metamodels into metadata APIs.

This International Standard also provides the following:

- a. A formal definition of the MOF meta-metamodel; that is, the abstract language for specifying MOF metamodels.
- b. A mapping from arbitrary MOF metamodels to CORBA IDL that produces IDL interfaces for managing any kind of metadata.
- c. A set of “reflective” CORBA IDL interfaces for managing metadata independent of the metamodel.
- d. A set of CORBA IDL interfaces for representing and managing MOF metamodels.
- e. An XMI format for MOF metamodel interchange (OMG XMI Specification).

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

2.2 International Standards

- ISO/IEC 14769:2001, *Information technology — Open Distributed Processing — Type Repository Function*
- ISO/IEC 19501:2005, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*
- ISO/IEC 19503:2005, *Information technology — XML Metadata Interchange (XMI)*