Meta-Modeling and the OMG Meta Object Facility (MOF)

A White Paper by the OCUP 2 Examination Team, March, 2017
A Reference for the OCUP 2 Advanced level examination

1 INTRODUCTION

The MOF 2.5 specification provides the basis for metamodel definition in OMG's family of modeling languages (including the UML) and is based on a simplification of UML 2's class modeling capabilities. In addition to providing the means for metamodel definition, it adds core capabilities for model management in general including Identifiers, a simple generic Tag capability, and Reflective operations that are defined generically and can be applied regardless of metamodel. Core MOF 2 is the foundation of other OMG MOF specifications, including the following (In this list 'MOF based model' means any model that instantiates a metamodel defined using MOF, which includes metamodels themselves):

- XMI - for interchanging MOF-based models in XML
- MOF 2 Facility and Object Lifecycle - for connecting to and managing collections of MOF-based model elements
- MOF 2 Versioning and Development Lifecycle - for managing versions and configurations of MOF-based models
- MOF Queries Views and Transformations - for transforming MOF-based models
- MOF Models to Text - for generating text, such as programs, from MOF-based models
- Object Constraint Language - for specifying constraints on MOF-based models

2 WHAT IS METADATA?

Metadata is simply data about data. One example of metadata is the schema in a relational database. Data like how many columns are in a table and what type of data is stored in each cell is metadata about the information in each row of the table. Another example of metadata is the set of elements defined by UML to be used by modelers to construct their models. In an example of the hierarchy of metamodels and models, the model so constructed becomes a metamodel (and so metadata) for the application based upon it.

The Meta Object Facility provides an open and platform-independent metadata management framework and associated set of metadata services to enable the development and interoperability of model- and metadata-driven systems. Examples of systems that use MOF include modeling and development tools, data warehouse systems, metadata repositories, etc. MOF has contributed significantly to the core principles of the OMG Model Driven Architecture (MDA). Building on the modeling foundation established by UML, MOF introduced the concept of formal metamodels and Platform Independent Models (PIM) of metadata (examples include several standard OMG metamodels including UML, and MOF itself).

3 META-MODEL HIERARCHY

The OMG meta-model hierarchy starts with MOF at the highest level of abstraction, with items at each level being dependent upon the items at the level above.

M3 MOF	Defines a language for specifying a metamodel Example: MOF
M2 UML	Defines a language for specifying models Example: UML
M1 User Models	Defines a language that describe semantic domains Example: model of a problem domain
M0 Instance Models	Contains run-time instances of the model elements defined in a model

Table 1. Model Hierarchy

4 HISTORY OF MOF

The current version of the MOF specification is 2.5.1[1]. This version of the specification is aligned with the UML 2.5 specification. The evolution of the definition of MOF began in 1996 with the original Request for Proposal (RFP):

- 1996 – MOF 1.0 RFP - The OMG Meta Object Facility was intended to be a suite of services designed to support the management of meta-information in a CORBA environment.
- 2006 – MOF 2.0 published – Sought to align MOF with Model Driven Architecture (MDA) concepts and UML 2.0.
- 2016 – MOF 2.5.1 published – Aligned MOF with the UML 2.5 specification simplification.

5 MOF ARCHITECTURE

There are two parts of the OMG MOF specification: Essential MOF (EMOF), and Complete MOF (CMOF). The specification states that "Compliant implementations may support EMOF only, or may support CMOF, which includes EMOF."

Essential MOF is the subset of MOF that closely corresponds to the facilities found in OOPLs and XML. The value of Essential MOF is that it provides a straightforward framework for mapping MOF models to implementations such as JMI (Java Metadata Interface) and XMI (XML Metadata Interchange) for simple metamodels. A primary goal of EMOF is to allow simple metamodels to be defined using simple

[1]The Meta Object Facility (MOF) V2.5.1, June 2015, Object Management Group, - http://www.omg.org/spec/MOF/2.5.1/

concepts while supporting extensions (by the usual class extension mechanism in MOF) for more sophisticated metamodeling using CMOF. Both EMOF and CMOF reuse the UML metamodel. The motivation behind this goal is to lower the barrier to entry for model driven tool development and tool integration.

The CMOF Model is the metamodel used to specify other metamodels such as UML 2. It is built from EMOF and selected elements of the UML metamodel. The Model package does not define any classes of its own. Rather, it merges packages with its extensions that together define basic metamodeling capabilities.

6 ABSTRACT SYNTAX

The MOF is described using both textual and graphic presentations. The MOF specification uses a combination of languages (a subset of UML, an object constraint language, and precise natural language) to precisely describe the abstract syntax and semantics of the MOF. Unlike MOF 1 and UML 1 where slightly different techniques (sometimes subtly different) were used, the MOF 2.5 specification reuses much of the formalisms in the UML 2.5 specification. In particular, EMOF and CMOF are both described using CMOF, which is also used to describe UML 2.5. EMOF is also completely described in EMOF by applying package import and merge semantics from its CMOF description. As a result, EMOF and CMOF are described using themselves, and each is derived from the UML 2.5 metamodel.

7 SEMANTICS

The semantics for the MOF elements are provided in the specification typically as natural language descriptions. For example, the Class *Element* is the top of the MOF hierarchy. The semantics for it are described in the MOF 2.5 specification as:

> Class *Element* is the superclass of all classes defined in MOF, and is an implicit superclass of all metaclasses defined using MOF: this superclass relationship to Element does not need to be explicitly modeled in MOF-compliant metamodels, and if implicit in this way Element is not included in the list of superclasses. By creating Properties with type Element it is possible to reference elements in any MOF-compliant model, similar to the use of xsd:any in XML Schemas. Each element can access its metaClass in order to obtain a Class that provides a reflective description of that element. By having both MOF and instances of MOF be rooted in class Element, MOF supports any number of meta layers.

8 MOF 2 DESIGN GOALS

The primary purpose of MOF is to provide a next-generation platform-independent metadata framework for OMG that builds on the unification accomplished in MOF 1.4, XMI 1.2, XMI production of XML Schemas, and JMI 1.0.

Continuing the tradition of MOF since 1997, MOF2 can be used to define and integrate a family of metamodels using simple class modeling concepts. As in MOF1 only UML class modeling notation is used to describe MOF compliant metamodels. What is significant about MOF2 is that we have unified the

modeling concepts in MOF2 and UML 2 and reused a common metamodel in both the MOF2 and UML 2 specifications. The major benefits of this approach include:

- Simpler rules for modeling metadata (just understand a subset of UML class modeling without any additional notations or modeling constructs).
- Various technology mappings from MOF (such as XMI, JMI, etc.) now also apply to a broader range of UML models including UML profiles.
- Broader tool support for metamodeling (any UML modeling tool can be used to model metadata more easily).

9 MOF RELATIONSHIP TO UML

9.1 THE DERIVATION OF MOF

Since version 2.4 for UML and MOF Core, MOF reuses the UML metamodel, narrowed down to the specific subsets for EMOF and CMOF by applying constraints. Standard class modeling concepts (importing, subclassing, adding new classes, associations, and adding associations between existing classes) are used for MOF 2 extensibility. These concepts are used to define additional packages (such as Reflection, Extents, and Identities) in MOF 2 as well as in any other MOF 2 compliant model.

9.2 UML 2.5 KERNEL REUSE

Figure 1, which is reproduced from Figure 14.2 in the MOF 2.5.1 specification, shows some of the key concrete classes and associations of UML 2.5 that are used to specify CMOF metamodels. There are many other important elements that are used from UML, but the ones shown in the figure provide the structure of class modeling. See the UML specification for full details regarding the classes and the MOF 2.5 specification for information on the other UML elements.

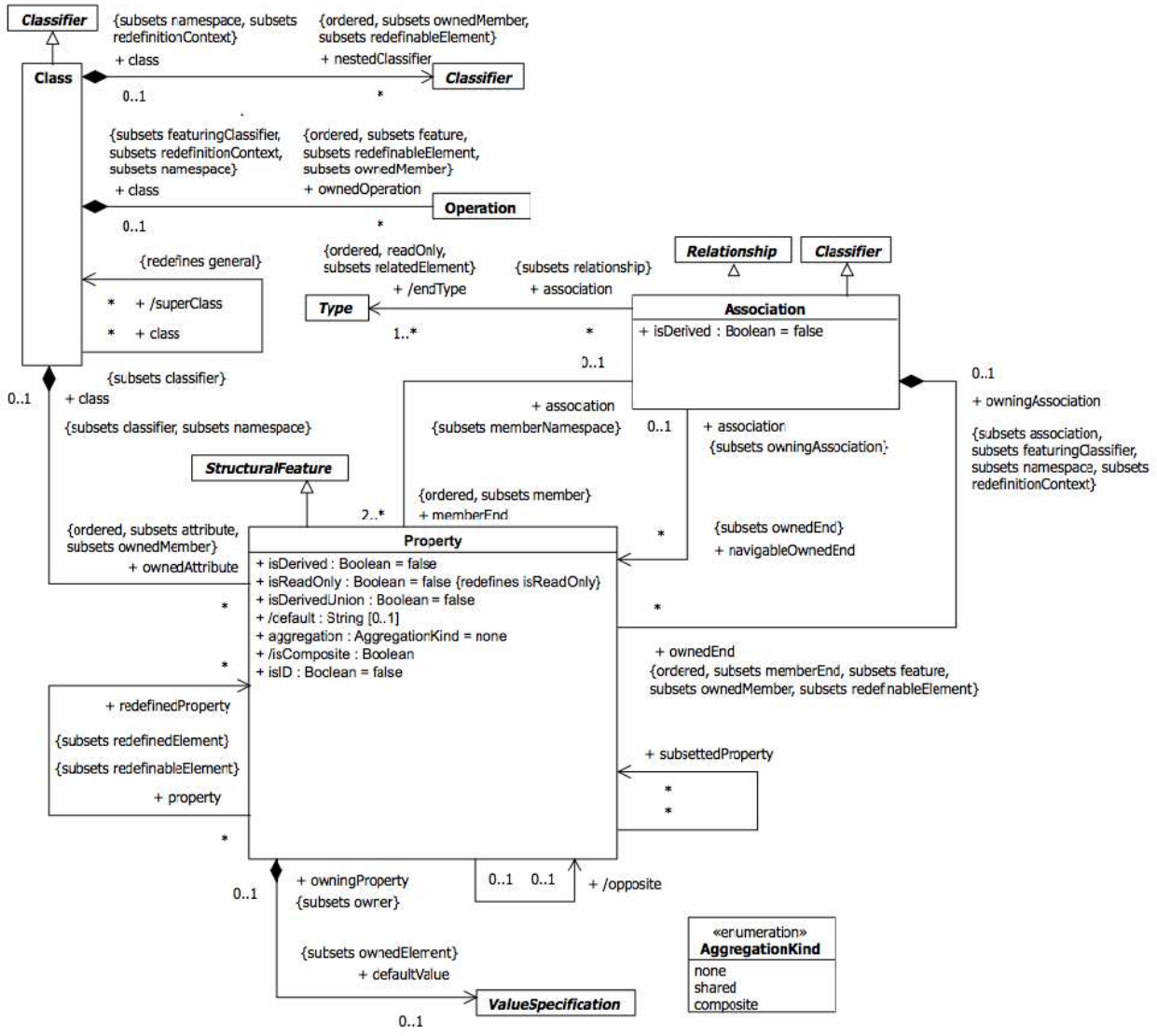

Figure 1. Key Concrete Classes

10 MODEL DRIVEN ARCHITECTURE (MDA)

The MDA provides an approach for deriving value from models and architecture in support of the full lifecycle of physical, organizational and information technology systems. The MDA approach represents and supports everything from requirements to business modeling to technology implementations. By using MDA models, we are able to better deal with the complexity of large systems and the interaction and collaboration between organizations, people, hardware, software. To better understand how MOF supports MDA, readers are directed to http://www.omg.org/mda/ and the OMG MDA Guide[2].

Glossary

CMOF	Complete MOF
CORBA	Common Object Request Broker Architecture
CWM	Common Warehouse Metamodel
EMOF	Essential MOF
MDA	Model Driven Architecture
MOF	Meta Object Facility
OMG	Object Management Group
OOPL	Object-Oriented Programming Language
PIM	Platform Independent Model
PSM	Platform Specific Model
QVT	Query/View/Transformation
UML	Unified Modeling Language
XMI	XML Metadata Interchange
XML	Extensible Markup Language

[2]Model Driven Architecture (MDA) MDA Guide rev. 2.0, Object Management Group, June 2014, OMG Document ormsc/2014-06-01. "http://www.omg.org/mda/

References

Model Driven Architecture (MDA) MDA Guide rev. 2.0, Object Management Group, June 2014, OMG Document ormsc/2014-06-01. "http://www.omg.org/mda/"

The Meta Object Facility (MOF) V2.5.1, June 2015, Object Management Group, - http://www.omg.org/spec/MOF/2.5.1/

Overbeek, J.F. (2006) *Meta Object Facility (MOF): investigation of the state of the art.* http://purl.utwente.nl/essays/57286