INTERNATIONAL STANDARD

ISO/IEC 19506

First edition
2012-04-15

Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)

Technologies de l'information — Modernisation conduite par l'architecture (ADM) de l'OMG — Métamodèle de découverte de connaissances (KDM)

Reference number
ISO/IEC 19506:2012(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 19500-1 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 Common Object Request Broker Architecture (CORBA) specification Part 1 Version 3.1 CORBA Interfaces.

ISO/IEC 19506 is related to:

- ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1995, Information Technology - Open Distributed Processing - Reference Model: Foundations
- ITU-T Recommendation X.903 (1995) | ISO/IEC 10746-3:1995, Information Technology - Open Distributed Processing - Reference Model: Architecture
- ITU-T Recommendation X.920 (1997) | ISO/IEC 14750:1997, Information Technology - Open Distributed Processing - Interface Definition Language

ISO/IEC 19506 consists of the following, under the general title *Information technology - Open distributed processing - Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM)*.

Apart from this Foreword, the text of this International Standard is identical with that for the OMG specification for Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM), v1.3.

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 which 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 3 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.

This International Standard for Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM) is a standard for the technology specification of an ODP system. It defines a technology to provide the infrastructure required to support functional distribution of an ODP system, specifying functions required to manage physical distribution, communications, processing and storage, and the roles of different technology objects in supporting those functions.

Information technology - Object Management Group Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM)

1 Scope

This International Standard defines a meta-model for representing *existing software assets,* their associations, and operational environments, referred to as the Knowledge Discovery Meta-model (KDM). This is the first in the series of specifications related to Software Assurance (SwA) and Architecture-Driven Modernization (ADM) activities. KDM facilitates projects that involve *existing software systems* by insuring interoperability and exchange of data between tools provided by different vendors.

One common characteristic of various tools that address SwA and ADM challenge is that they analyze *existing software assets* (for example, source code modules, database descriptions, build scripts, etc.) to obtain explicit knowledge. Each tool produces a portion of the knowledge about existing software assets. Such tool-specific knowledge may be implicit ("hard-coded" in the tool), restricted to a particular source language, and/or particular transformation, and/or *operational environment*. All the above may hinder interoperability between different tools. The meta-model for Knowledge Discovery provides a common repository structure that facilitates the exchange of data contained within individual tool models that represent *existing software assets*. The meta-model represents the physical and logical assets at various levels of abstraction. The primary purpose of this meta-model is to provide a common interchange format that will allow interoperability between existing modernization and software assurance tools, services, and their respective intermediate representations.

2 Conformance and Compliance

KDM is a meta-model with a very broad scope that covers a large and diverse set of applications, platforms, and programming languages. Not all of its capabilities are equally applicable to all platforms, applications, or programming languages. The primary goal of KDM is to provide the capability to exchange models between tools and thus facilitate cooperation between tool suppliers by allowing integration information about a complex *enterprise application* from multiple sources, as the complexity of modern *enterprise applications* involves multiple platform technologies and programming languages. In order to achieve interoperability and especially the integration of information about different facets of an *enterprise application* from multiple analysis tools, this International Standard defines several compliance levels thereby increasing the likelihood that two or more compliant tools will support the same or compatible meta-model subsets. This suggests that the meta-model should be structured modularly, following the principle of separation of concerns, with the ability to select only those parts of the meta-model that are of direct interest to a particular tool vendor. Consequently, the definition of compliance for KDM requires a balance to be drawn between modularity and ease of interchange. Separation of concerns in the design of KDM is embodied in the concept of KDM domains.

2.1 KDM Domains

Separate facts of knowledge discovery in enterprise application in KDM are grouped into several KDM domains (refer to Figure 2.1). Each KDM domain consists of a single KDM package that defines meta-model elements to represent particular aspects of the system under study. KDM domains correspond to the well-known concept of *architecture views*. For example, the Structure domain enables users to discover architectural elements of source code from the system under study, while the Business Rules domain provides users with behavioral elements of the same system such as features or process rules.

The following domains of knowledge have been identified as the foundation for defining compliance in KDM: Build, Structure, Data, Business Rules, UI, Event, Platform, and micro KDM.

From the user's perspective, this partitioning of KDM means that they need only to be concerned with those parts of the KDM that they consider necessary for their activities. If those needs change over time, further KDM domains can be added to the user's repertoire as required. Hence, a KDM user does not have to know the full meta-model to use it effectively. In addition, most KDM domains are partitioned into multiple increments, each adding more knowledge capabilities to the previous ones. This fine-grained decomposition of KDM serves to make the KDM easier to learn and use, but the individual segments within this structure do not represent separate compliance points. The latter strategy would lead to an excess of compliance points and result to the interoperability problems described above. Nevertheless, the groupings provided by KDM domains and their increments do serve to simplify the definition of KDM compliance as explained below.

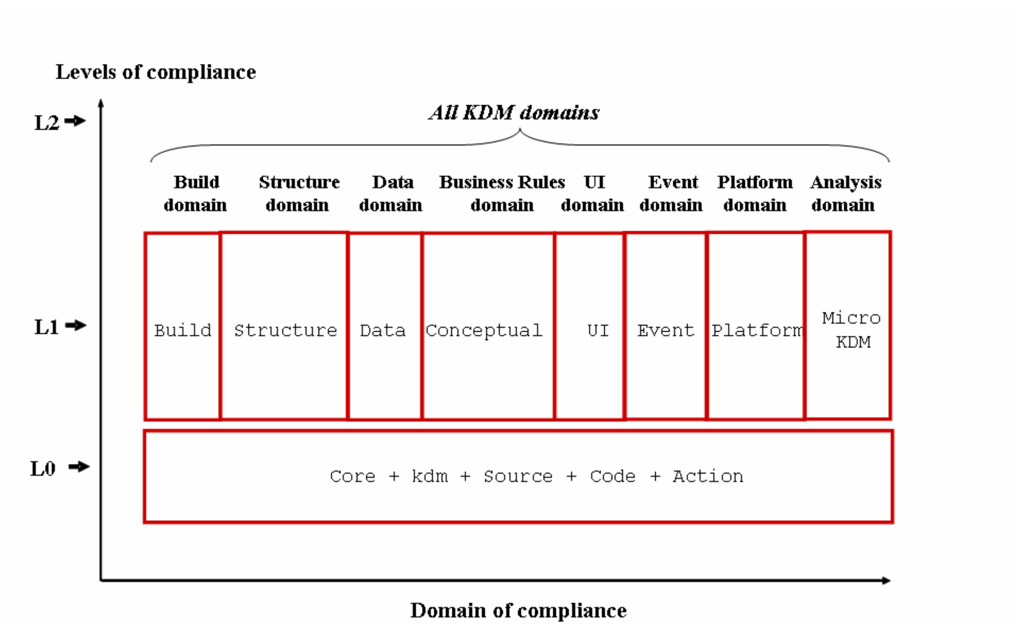

Figure 2.1- Domains and levels of KDM compliance

2.2 Compliance Levels

In addition, the total set of KDM packages is further partitioned into layers of increasing capability called compliance levels. There are two KDM compliance levels:

- Level 0 (L0) - This compliance level contains the following KDM packages: Core, kdm, Source, Code, and Action packages. It provides an entry-level of knowledge discovery capability. More importantly, it represents a common denominator that can serve as a basis for interoperability between different categories of KDM tools.

  To be L0 compliant, a tool must completely support all model elements within all packages for L0 level.

- Level 1 (L1) - This level addresses KDM domains and extends the capabilities provided by Level 0. Specifically, it adds the following packages: Build, Structure, Data, Conceptual, UI, Event, Platform, as well as the set of constraints for the micro KDM domain defined in sub clause 14 "Micro KDM," and Annex A "Semantics of the Micro KDM Action Elements." These packages are grouped to form above-mentioned domains. More importantly, this level represents a layer where tools could be complimentary since their focus would be in different areas of concern. This would be an additional reason why L0 interoperability (which at this level would be viewed as information sharing between tools) is mandated. In this case interoperability at this level would be viewed as correlation between tools to complete knowledge puzzle that end user might need to perform a particular task.

  To be L1 compliant for a given KDM domain, a tool must completely support all model elements defined by the package for that domain and satisfy all semantic constraints specified for that domain.

- Level 2 (L2) - This level is the union of L1 levels for all KDM domains.

2.3 Meaning and Types of Compliance

Compliance to Level 1 (L1) for a certain KDM domain entails full realization of all KDM packages for the corresponding KDM Domain. This also implies full realization of all KDM packages in all the levels below that level (in this case Level 0 (L0)). It is not meaningful to claim compliance to Level 1 without also being compliant with the Level 0. A tool that is compliant at a Level 1 must be able (at least) to import models from tools that are compliant to Level 0 without loss of information. So, "full realization" for a KDM domain means supporting the complete set of concepts defined for that KDM domain at L1 and complete set of concepts defined at L0.

For a given compliance level, a *KDM implementation* can provide:

- The capability to analyze physical artifacts of existing applications and export their representations based on the XMI schema corresponding to the given compliance level.
- The capability to import representations of existing software systems based on the XMI schema corresponding to the given compliance level and perform operations suggested by the corresponding packages.

Table 2.1- Compliance Statements

Compliance Statement				
Compliance Level		**Import-Analysis**	**Import API**	**Export**
L0		Import KDM models based on complete KDM XMI schema into existing tool; support specified mapping between KDM and existing model in the tool; extend operations of existing tool to support meta-model elements of KDM framework; extend operations of existing tool to support meta-model elements of Code and Action packages; extend operations of existing tool to traceability to the physical artifacts of the application from Source package.	Import KDM models based on complete KDM XMI schema; support KDM API defined by the KDM Core package; support KDM framework as defined in the Kdm package; support KDM API defined by the Code and Action packages; support traceability to the physical artifacts of the application as defined in the Source package.	Provide capability to analyze artifacts of an application for specified programming language or multiple languages; Generate XMI documents corresponding to the KDM XMI schema; Support KDM framework as defined by the Kdm package; Support Code and Action packages; Provide traceability back to the physical artifacts as defined by the Source package.
L1	**STRUCTURE**	L0 compliance for analysis;extend operations of existing tool to support meta-model elements of the Structure package.	L0 compliance for import; Support KDM API as defined by the Structure package.	L0 compliance for export;Provide capability to analyze architecture components of existing application and generate KDM Structure model according to Structure package.
	DATA	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the Data package.	L0 compliance for import; Support KDM API as defined by the Data package.	L0 compliance for export; Provide capability to analyze persistent data components of existing application for specified database system and generate KDM Data model according to Data package.
	PLATFORM	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the Platform package.	L0 compliance for import; Support KDM API as defined by the Platform and Runtime packages.	L0 compliance for export; Provide capability to analyze platform artifacts for specified platform and generate KDM Platform model according to Platform package.
	BUILD	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the Build package.	L0 compliance for import; Support KDM API as defined by the Build package.	L0 compliance for export; Provide capability to analyze build artifacts for specified build environment and generate KDM Build model according to Build package.

Table 2.1- Compliance Statements

	UI	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the UI package.	L0 compliance for import; Support KDM API as defined by the UI package.	L0 compliance for export; Provide capability to analyze user interface artifacts for specified user interface system and generate KDM UI model according to UI package.
	EVENT	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the Event package.	L0 compliance for import; Support KDM API as defined by the Event package.	L0 compliance for export; Provide capability to analyze artifacts related to event-driven runtime frameworks and state-trasition behavior and generate KDM Event model according to Event package.
	BUSINESS	L0 compliance for analysis; extend operations of existing tool to support meta-model elements of the Conceptual package.	L0 compliance for import; Support KDM API as defined by the Conceptual package.	L0 compliance for export; Provide capability to analyze conceptual and behavior artifacts (e.g., domain concepts, business rules, scenarios) of existing application and generate KDM Conceptual model according to Conceptual package.
	MICRO KDM	L0 compliance for analysis; extend operations of existing tool to support micro KDM actions as specified in sub clause 14 micro KDM and Annex A.	L0 compliance for import; Support micro KDM actions as specified in sub clause 14 micro KDM and Annex A.	L0 compliance for export; Provide capability to analyze artifacts of existing application to the level of detail specified in sub clause 14 and Annex A provide the mapping of semantics of the existing application as it is determined by the programming languages and the runtime platform into KDM micro actions and generate KDM models that represent the same meaning
L2		L0 import compliance for analysis; L1 import-analysis compliance for all KDM domains.	L0 compliance for import; Support KDM API as defined by all KDM packages.	L0 export compliance; L1 export compliance for all KDM domains.

3 Normative References

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

- OMG UML Infrastructure Specification, v2.3, formal/2010-05-03
- OMG Meta-Object Facility (MOF) Specification, v2.0, formal/2006-01-01
- OMG MOF XML Metadata Interchange (XMI) Specification, v2.1, formal/2005-09-01
- OMG Semantics of Business Vocabularies and Business Rules (SBVR) Specification, v1.0, formal/2008-01-02
- ISO/IEC 11404:2007 Information technology -- General Purpose Datatypes (GPD)