TECHNICAL SPECIFICATION

ISO/TS 10303-18

First edition
2021-05

Industrial automation systems and integration — Product data representation and exchange —

Part 18: Description methods: SysML XMI to Web services transformation

Reference number
ISO/TS 10303-18:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 4, *Industrial data*.

A list of all parts in the ISO 10303 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product and independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving.

This document is a member of the description methods series. This document specifies the web services defined for ISO 10303. It includes the rationale for the selection of the web services and their scope. It does not include any specifications for management or maintenance of information and data on the server. This document supports the STEP extended architecture[12][13].

The need for standardized services

The traditional means for exchanging data using ISO 10303 is to share large packets of information using files. However, with increasingly interconnected organizations and tool sets, the need has arisen for secure flexible sharing of smaller packets of data. This need is valid for every industrial domain from analysis to design, and lifecycle support. It is also true for the different models of the STEP Extended architecture with services needed for the core, domain and data planning models.

The rapid evolution of web technologies means that the services definitions need to be independent of technology, with technology specific implementation schemas generated from the definitions.

The purpose of standardized services is to:

— define services against a model so that two or more vendors provide consistent implementations;

— provide a minimum specification to ensure consistency so that a third party can use the services from either of the vendors (such as “plug and play”).

The document development method:

Much of the content for Clause 5 was harvested from public deliverables of the Aerospace Technology Institute[21] APROCONE project[22] where intent was declared in these deliverables for the content to be used in this document. This was approved by all APROCONE project members.

The main components of this document are:

— types of services: a description of the categories of services from domain and technology independent to specific, and the rationale for the scope for this document;

— technology-independent services definition: the definition of the technology-independent services along with their inputs and outputs;

— technology-dependent methods: SysML XMI to OpenAPI 3.0.0 JSON schema;

— Annex B: SysML to OpenAPI - Canonical XMI and equivalent in OpenAPI JSON schema;

— Annex C: SysML to OpenAPI – Example files;

— Annex D: SysML to OpenAPI - A listing of, and mapping between, SDAI and HTTP response status codes.

Industrial automation systems and integration — Product data representation and exchange —

Part 18: Description methods: SysML XMI to Web services transformation

1 Scope

This document specifies the definition for services at the point of interaction between a client and server.

The following are within the scope of this document:

— the specification of the structure, components and conventions for domain- and technology-independent services implementation methods for STEP (ISO 10303-1);

— transformation of the SysML metamodel constructs to OpenAPI constructs for RESTful web services (see OpenAPI:3.0.0[25] and IETF RFC7231).

The following are outside the scope of this document:

— domain specific services definitions;

— the transformation of SysML metamodel constructs into OpenAPI constructs that are not used in the STEP extended architecture[12][13];

— the transformation of SysML metamodel constructs into OpenAPI constructs for other purposes than representing SysML constructs as STEP concepts;

— codes and scripts to transform SysML XMI to OpenAPI schema;

— the transformation of SysML constraints into OpenAPI schema;

— implementation of technology-specific services definitions other than RESTful OpenAPI;

— definition of management and maintenance of information and data on a server.

2 Normative references

There are no normative references in this document.

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1.1 Terms and definitions for generic concepts

3.1.1.1
data
representation of information in a formal manner suitable for communication, interpretation, or processing by human beings or computers.

[SOURCE: ISO 10303-1:2021, 3.1.29]

3.1.1.2
implementation method
part of ISO 10303 that specifies a technique used by computer systems to exchange product data

[SOURCE: ISO 10303-1:2021, 3.1.39, modified — In the definition, the text after "data" has been removed.]

3.1.1.3
information
facts, concepts or instructions.

[SOURCE: ISO 10303-1:2021, 3.1.41]

3.1.1.4
information model
conceptual model of product data

Note 1 to entry: In ISO 10303, an information model is based on the Object-relationship modeling technique that organizes the product data as represented in different system aspects.

Note 2 to entry: In ISO 10303 information models are may be developed using EXPRESS modeling language.

EXAMPLE Application resource model for ISO 10303-242 managed model-based 3D engineering.

[SOURCE: ISO 10303-1:2021, 3.1.42 modified — The example has been changed.]

3.1.2 Terms and definitions for STEP concepts

3.1.2.1
entity
class of information defined by common properties

[SOURCE: ISO 10303-11:2004, 3.3.6, modified — In the definition, the article "a" has been removed.]

3.1.2.2
value
unit of data

[SOURCE: ISO 10303-11:2004, 3.3.22, modified — In the definition, the article "a" has been removed.]

3.1.3 Terms and definitions for SysML constructs

3.1.3.1
association
association classifies a set of tuples representing links between typed model elements

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.2]

3.1.3.2
auxiliary
a stereotype applied to an abstract *block* (3.1.3.3) that has no properties

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.3]

3.1.3.3
block
modular construct used for defining an *entity* (3.1.2.1)

Note 1 to entry: Used for defining objects in *information models* (3.1.1.4) such as Application activity model concepts, Application Data Planning objects, Application Domain Model Business Objects, Core model objects and ARM in SysML Entities. They may include reference, part, and value properties; constraints. They can be specializations of other Blocks.

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.4]

3.1.3.4
composite aggregation
responsibility for the existence of composed object.

Note 1 to entry: If a composite object is deleted, all of its part instances that are objects are deleted with it

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.5]

3.1.3.5
directed association
association between a collection of source model elements and a collection of target model elements that is said to be directed from the source elements to the target elements

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.6]

3.1.3.6
enumeration
Value Type whose values are enumerated

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.7]

3.1.3.7
enumeration literal
named value for an *enumeration*

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.8]

3.1.3.8
data type
type whose instances are identified only by their value

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.9]

3.1.3.9
generalization
directed relationship between a more general supertype and a more specific subtype

Note 1 to entry: Each Generalization relates a specific Classifier to a more general Classifier. Given a Classifier, the transitive closure of its general Classifiers is often called its generalizations, and the transitive closure of its specific Classifiers is called its specializations. The immediate generalizations are also called the Classifier's subtype, and where the Classifier is a Class, its supertype.

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.10]

3.1.3.10
part property
property that specifies a part with strong ownership and coincidental lifetime of its containing *block* (3.1.3.3).

Note 1 to entry: It describes a local usage or a role of the typing Block in the context of the containing Block. Every Part Property has Composite Aggregation and is typed by a Block.

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.12]

3.1.3.11
primitive type
definition of a predefined DataType, without any substructure

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.11]

3.1.3.12
reference property
property that specifies a reference of its containing *block* (3.1.3.3) to another block

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.13]

3.1.3.13
stereotype
limited kind of metaclass that cannot be used by itself but must always be used in conjunction with one of the metaclasses it extends

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.14]

3.1.3.14
value property
property of a *block* (3.1.3.3) that is typed with a ValueType

[SOURCE: ISO/TS 10303-15:2021, 3.1.3.15]

3.1.4 Terms and definitions used in OpenAPI specification

3.1.4.1
openapi
<OpenAPI> semantic version number field of the OpenAPI specification version

Note 1 to entry: A required fixed field in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.2
info
<OpenAPI> metadata about the schema field

Note 1 to entry: A required fixed field in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.3
servers
<OpenAPI> connectivity information to one or more target servers field

Note 1 to entry: A fixed field in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.4
tags
<OpenAPI> metadata representing logical grouping field.

Note 1 to entry: The full definition is provided in OpenAPI specification.

3.1.4.5
paths
<OpenAPI> available relative paths to the individual endpoints field

Note 1 to entry: A required fixed field in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.6
components
<OpenAPI> container for holding various schemas

Note 1 to entry: The full definition is provided in OpenAPI specification.

3.1.4.7
response object
<OpenAPI> description of a single response from an operation

Note 1 to entry: A response object may include design-time, static links to operations based on the response.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.8
responses
<OpenAPI> reusable *response objects* (3.1.4.7) field

Note 1 to entry: A fixed field in the field *components* in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.4.9
schema object
<OpenAPI> input and output data types definition

Note 1 to entry: The full definition is provided in *OpenAPI specification.*

3.1.4.10
schemas
<OpenAPI> reusable *schema objects* (3.1.4.9)

Note 1 to entry: A fixed field in the field *components* (3.1.4.6) in an OpenAPI schema.

Note 2 to entry: The full definition is provided in OpenAPI specification.

3.1.5 Terms and definitions used in hypertext transfer protocol

3.1.5.1
resource
<HTTP> HTTP request target

Note 1 to entry: HTTP does not limit the nature of a resource; it merely defines an interface that might be used to interact with resources. Each resource is identified by a uniform resource identifier (URI).

Note 2 to entry: The full definition is provided in RFC 7231:2014, 2.

3.1.5.2
representation
<HTTP> information reflecting a past, current, or desired state of a given *resource* (3.1.5.1).

Note 1 to entry: the representation is in a format that can be readily communicated via the protocol, and that consists of a set of representation metadata and a potentially unbounded stream of representation data

Note 2 to entry: The full definition is provided in RFC 7231:2014, 3.

3.1.5.3
response status code
<HTTP> result of the attempt to realise the request represented as a three-digit integer code

Note 1 to entry: only the response status codes starting with 2 (Successful), 4 (Client error) and 5 (Server error) are used.

Note 2 to entry: The full definition is provided in RFC 7231:2014, 6.

3.1.5.4
POST
method that requests that the target *resource* (3.1.5.1) process the *representation* (3.1.5.2) contained in the request

Note 1 to entry: The full definition is provided in RFC 7231:2014, 4.3.3.

3.1.5.5
GET
method that requests a *representation* (3.1.5.2) for the target *resource* (3.1.5.1) is transferred

Note 1 to entry: The full definition is provided in RFC 7231:2014, 4.3.1.

3.1.5.6
PATCH
method that requests that a set of changes described in the request entity be applied to the *resource* (3.1.5.1)

Note 1 to entry: The full definition is provided in RFC 6902:2013, 1.

3.1.5.7
PUT
method that requests that the target *resource* (3.1.5.1) state be created or replaced with the state defined by the *representation* (3.1.5.2) contained in the request

Note 1 to entry: The full definition is provided in RFC 7231:2014, 4.3.4.

3.2 Abbreviated terms

AP	application protocol
API	application programming interface
APROCONE	advanced product concept analysis environment
CRUD	create read update delete
HTTP	hypertext transfer protocol
ID	identifier
JSON	Java script object notation
OCL	object constraint language
OMG	object management group
REST	representational state transfer
SDAI	standard data access interface
SDL	schema definition language

SOAP	simple object access protocol
STEP	standard for the exchange of product model data
SysML	system modeling language
UID	unique identifier
UML	unified modeling language
URI	universal resource indicator
UUID	universal unique identifier
WSDL	web services description language
XMI	xml metadata interchange
XML	extensible markup language
XSD	xml schema definition

4 Domain-independent, technology-independent services

4.1 Domain- and technology-independent services overview (CRUD+Query)

Figure 1 below shows the classification of the services according to their domain and implementation technology independence. This clause concerns the domain and implementation technology independent services definitions, shown in the bottom left of Figure 1.

NOTE 1 The domain independent, technology specific services are covered in Clause 5. The domain-specific services are not considered in this document.

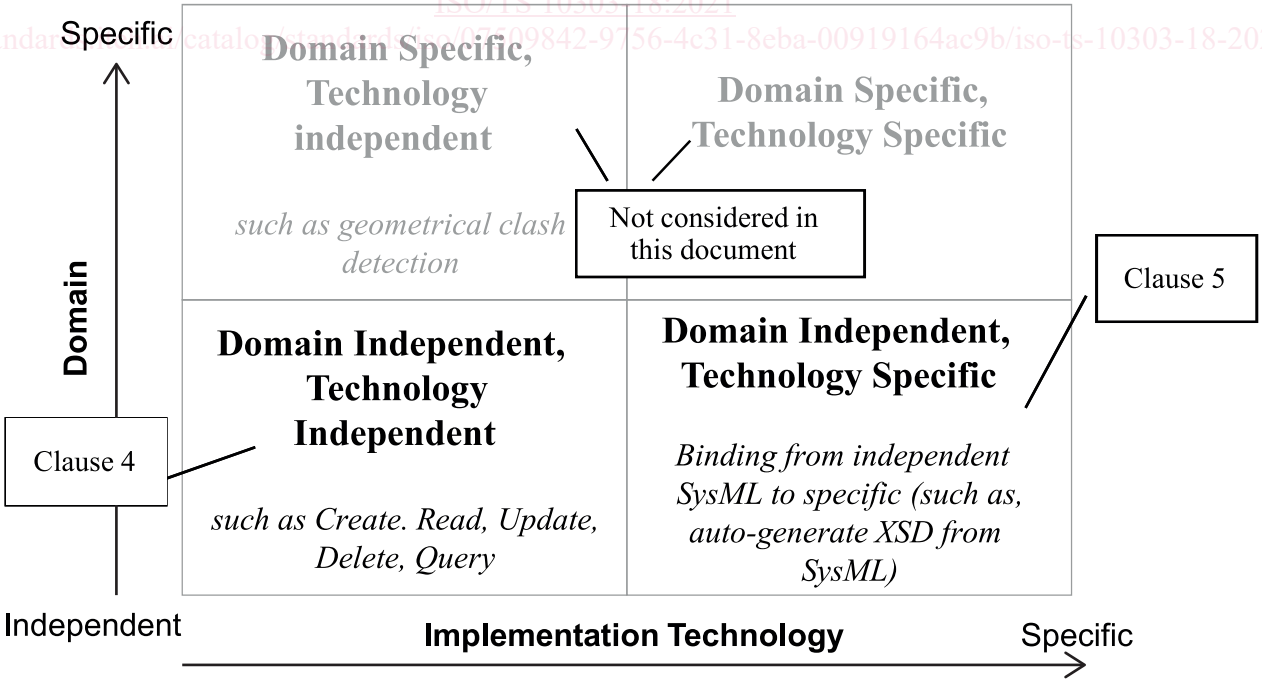

Figure 1 — Classification of services according to domain and technology independence

The domain- independent and technology-independent services are used to mutate (Create, Update, Delete) and interrogate (Read, Query) the data.

The standard to which the services apply, shall be directly reflected in any specializations and in the payload of the service request and response;

EXAMPLE 1 If the standard has an entity called “Circle” that has two part properties called “position” and “radius”, then the service can be CreateCircle(position, radius), or better still Create(type arguments) which for the Circle example equates to Create(type=Circle, arguments={position, radius}).

The services can also be used to define more advanced CRUD+Query services which combine aspects of the model. They are defined by modelling additional entities or properties and using parametric diagrams to detail what shall be created behind the scenes. The approach means that normal “CRUD+Query” can then be used for these new entities. Internal to an implementation, a developer can choose whether to directly use the new property or use the mapping defined in the parametric diagram, depending on which is best for their internal data model.

EXAMPLE 2 A Person “AssignTo” (see Figure 2) service can be defined to create/read the appropriate assignment object (OrganizationOrPersonInOrganizationAssignment or PersonInOrganization) depending on the type to which the person is being assigned.

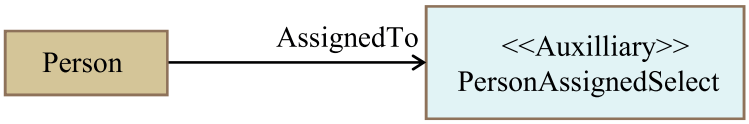

Figure 2 — Person AssignedTo

NOTE 2 The following types of services are not considered to be domain independent services and so are not considered in this document:

— combined services:
  — these services do complex tasks, but can be defined using a series of CRUD+Q services;
  — these types of services are generally specific to a particular use case and, as such, cannot be included in this document. Instead, they can be implemented in the client;
  — if a generic combined service is needed, then the approach of abstracting the service to a property and using parametric diagrams to define the behaviour (as described in this subclause) can be suitable for defining this type of service;
— value-added services:
  — these services do other tasks with the data;
  — the parametric diagrams approach (as described in this subclause) can, in some instances, be used to define this type of service.

EXAMPLE 3 Calculation or method services.

EXAMPLE 4 Create Circle(pt1, pt2, pt3). Assuming a standard "ABC" that defined the circle by using a centre point and radius, this service first needs to compute the centre point and radius from the three points before it can use these values to create a Circle following the standard "ABC".

EXAMPLE 5 Read CircleArea(). This service needs to read the circle retrieving the radius and then compute the area

EXAMPLE 6 Compare two sets of data (for validation – are they the same).