TECHNICAL SPECIFICATION

ISO/TS 15926-8

First edition
2011-10-15

Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities —

Part 8:
Implementation methods for the integration of distributed systems: Web Ontology Language (OWL) implementation

Systèmes d'automatisation industrielle et intégration — Intégration de données de cycle de vie pour les industries de «process», y compris les usines de production de pétrole et de gaz —

Partie 8: Méthodes de mise en œuvre pour l'intégration de systèmes distribués: Mise en oeuvre du langage d'ontologie du Web (OWL)

Reference number
ISO/TS 15926-8:2011(E)

Contents

Page

Foreword . . . vi

Introduction . . . vii

1 Scope . . . 1

2 Terms, definitions, and abbreviated terms . . . 1
2.1 Terms and definitions . . . 1
2.2 Abbreviated terms . . . 7

3 Fundamental concepts and assumptions . . . 8
3.1 General . . . 8
3.2 ISO/TS 15926-7 templates and first order logic rules . . . 8
3.3 ISO/TS 15926-4 reference data . . . 8
3.4 User-defined taxonomy . . . 8
3.5 Templates and specialized templates . . . 9
3.6 Use of OWL . . . 9
3.7 Namespaces . . . 9

4 ISO 15926-2 data model . . . 12

5 ISO/TS 15926-4 reference data . . . 12

6 OWL template specifications . . . 13
6.1 General . . . 13
6.2 Contents . . . 13

7 Templates . . . 14
7.1 General . . . 14
7.2 Template model . . . 14
7.3 Metadata . . . 15
7.4 Lifted templates . . . 15
7.5 Lowered templates . . . 16
7.5.1 Instantiating lowered templates . . . 16
7.6 Templates in RDF . . . 16
7.7 Example: A proto-template . . . 16
7.8 Templates as template instances . . . 16
7.8.1 p7tm classes for meta-templates . . . 16

8 Object information models (OIMs) and specialized templates . . . 17
8.1 General . . . 17
8.2 OIM versus specialized template construct . . . 18
8.3 Specialization . . . 18
8.4 Cardinalities . . . 19
8.5 OIM for assemblies . . . 19
8.6 Naming convention on template roles . . . 19
8.7 Uniqueness context of template roles . . . 20

Annex A (normative) Information object registration . . . 21
A.1 Document identification . . . 21
A.2 Schema identification . . . 21
A.2.1 OWL ontology for data model . . . 21

A.2.2 Template specifications . . . 21
A.2.3 OWL ontology for templates . . . 21
A.2.4 OWL ontology for reference data . . . 21
A.2.5 OWL ontology for metadata . . . 22

Annex B (normative) Templates as RDF/OWL n-ary relations . . . 23
B.1 General . . . 23
B.2 ISO/TS 15926-8 constructs for template signatures . . . 24

Annex C (normative) Rules for usage of OWL . . . 27
C.1 OWL native representation . . . 27
C.2 Template translation from EXPRESS-native to OWL-native instance model . . . 28
C.3 Library object translation from EXPRESS-native to OWL-native instance model . . . 28
C.4 owl:Class . . . 28
C.5 Individual . . . 29
C.6 Property . . . 29
C.6.1 Object property . . . 29
C.6.2 Datatype property . . . 29
C.6.3 Annotation property . . . 29
C.7 Identifiers . . . 30
C.7.1 rdf:ID . . . 30
C.7.2 rdf:about . . . 30
C.7.3 URI reference for handed-over objects . . . 30
C.8 Inheritance . . . 31
C.8.1 Declaration . . . 31

Annex D (normative) Computer interpretable listings . . . 32
D.1 Ontology for data model . . . 32
D.2 Ontology for template model . . . 32
D.3 Ontology for proto templates and templates initial set . . . 32
D.4 Ontology for metadata . . . 32

Annex E (informative) Pattern for an ontology for reference data . . . 33

Annex F (normative) Metadata . . . 34
F.1 General . . . 34
F.2 Metadata declarations for provenance . . . 34
F.3 Metadata declarations for security and rules . . . 35

Annex G (informative) Assertion versus terminological components . . . 36

Annex H (informative) Example use . . . 37
H.1 General . . . 37
H.2 Example: A proto-template . . . 37
H.2.1 The template defined as a class . . . 37
H.2.2 A statement using the template . . . 37
H.3 Example: OIM . . . 39
H.3.1 Description . . . 39
H.3.2 Manchester syntax . . . 39
H.4 Example: temporal parts . . . 40
H.4.1 General . . . 40
H.4.2 Generic representation . . . 41
H.4.3 Specialized template IdentificationOfPhysicalObject . . . 41
H.4.4 Specialized template IdentificationOfPhysicalObject-processplant . . . 42

H.5 A specialized template and a template instance 43

Bibliography 47

Index 49

Figures

Figure 1 — Dependencies between ontologies 12
Figure B.1 — W3C's working group note: defining n-ary relations 23
Figure C.1 — Reified relationship 27
Figure C.2 — OWL-native relationship 27
Figure H.1 — Generic temporal parts, identification OIM 41
Figure H.2 — ProcessPlant temporal part, name OIM 43

Tables

Table 1 — URIs under control of the World Wide Web Consortium 9
Table 2 — Namespaces that are used for the examples in this part of ISO 15926. 10
Table F.1 — Examples of properties that are rdf:subPropertyOf OWL standard annotation properties 35
Table F.2 — Meta data elements from ISO/TS 15926-8 35

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.

In other circumstances, particularly when there is an urgent market requirement for such documents, a technical committee may decide to publish other types of normative document:

— an ISO Publicly Available Specification (ISO/PAS) represents an agreement between technical experts in an ISO working group and is accepted for publication if it is approved by more than 50% of the members of the parent committee casting a vote;

— an ISO Technical Specification (ISO/TS) represents an agreement between the members of a technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/PAS or ISO/TS is reviewed every three years with a view to deciding whether it can be transformed into an International Standard.

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/TS 15926-8 was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 4, *Industrial data.*

ISO 15926 is organized as a series of parts, each published separately. The structure of ISO 15926 is described in ISO 15926-1.

Each part of ISO 15926 is a member of the following series: data model, reference data, implementation methods, conformance testing methodology and framework, characterization methods, abstract test suites. This part of ISO 15926 is a member of the implementation methods series.

A complete list of parts of ISO 15926 is available from the following URL:

```
http://www.tc184-sc4.org/titles/OIL_GAS_Titles.htm
```

Introduction

ISO 15926 is an International Standard for the representation of process plant life-cycle information. This representation is specified by a generic, conceptual data model that is suitable as the basis for implementation in a shared database or data warehouse. The data model is designed to be used in conjunction with reference data: standard instances that represent information common to a number of users, process plants, or both. The support for a specific life-cycle activity depends on the use of appropriate reference data in conjunction with the data model.

ISO 15926 is organized as a number of parts, each published separately. This part of ISO 15926 specifies the Web Ontology Language (OWL) implementation, using World Wide Web Consortium (W3C) Semantic Web technologies.

This part of ISO 15926 deals with the translation of ISO/TS 15926-4 classes, reference data and the ISO/TS 15926-7 template methodology to Resource Description Framework (RDF) and Web Ontology Language (OWL), which can be used in data modelling, integration and interoperability methods. This part of ISO 15926 is independent of infrastructure and test methods.

This part of ISO 15926 serves as the basis for data integration and interoperability infrastructure and test methods.

This part of ISO 15926 addresses:

— the method of translating ISO/TS 15926-4 classes to RDF/OWL;

— the method of translating ISO/TS 15926-7 templates to RDF/OWL;

— the constructs of specialized templates;

— the use of object information models;

— the constructs of metadata.

Readers of this part of ISO 15926 require an understanding of conceptual data models and of ISO/TS 15926-7.

The target audiences for this part of ISO 15926 are as follows:

— technical managers wishing to determine whether ISO 15926 is appropriate for their business needs;

— implementers wishing to make interface software between legacy systems and ISO 15926 compliant systems;

— implementers wishing to make software internally ISO 15926-compliant for the purpose of data integration.

In this part of ISO 15926, the same English language word might be used to refer to a real world thing, to an EXPRESS representation of the real world thing, or to an RDF/XML representation of the real-world thing. These uses are distinguished by the following typographic conventions:

— if a word or phrase occurs in normal typeface, it refers to the real-world thing;

EXAMPLE 1 cooling water pump

— if the word or phrase occurs in **bold** typeface with underscores, it refers to the EXPRESS representation from the ISO 15926-2 data model;

EXAMPLE 2 **class_of_inanimate_physical_object**

— if the word or phrase occurs in **bold** typeface and in CamelCase, it refers to a subtype axiom as defined in ISO/TS 15926-7;

EXAMPLE 3 **ClassOfInanimatePhysicalObject**

— if the word occurs in *italic* typeface, it refers to an RDF/RDFS/OWL native entity type.

EXAMPLE 4 *rdfs:subClassOf*

References to identifiers in examples are fictitious.

Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities —

Part 8: Implementation methods for the integration of distributed systems: Web Ontology Language (OWL) implementation

1 Scope

This part of ISO 15926 specifies implementation methods for integration, sharing, exchange, and hand-over of life-cycle information about process plants, based on the data model of ISO 15926-2 and the template methodology of ISO/TS 15926-7.

The following are within the scope of this part of ISO 15926:

— defining rules for applying RDF and OWL in the context of this part of ISO 15926;

— mapping of the data model of ISO 15926-2 from its EXPRESS format to OWL-2;

— defining a methodology for creating an OWL ontology for the ISO/TS 15926-4 reference data;

— defining an OWL ontology based on the base templates and the initial set of core templates;

— defining a methodology for creating an OWL ontology for "specialized templates" (see 2.1.32) that defines the types of information for any given instance of **possible_individual** during its lifetime.

The following are outside the scope of this part of ISO 15926:

— the specific type of rule language used to implement the first order logic;

— the decision as to whether data storage and exchange is done using lifted data or by use of lowered template instances and objects only.

NOTE This is a business decision.

2 Terms, definitions, and abbreviated terms

2.1 Terms and definitions

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

2.1.1
base template
template with only entity types in the expansion of its template axiom

[ISO/TS 15926-7:2011, definition 2.1.1]

2.1.2
class
category or division of things based on one or more critera for inclusion and exclusion

NOTE 1 A class need not have any known members (things that satisfy its criteria for membership).

NOTE 2 Because of the spatio-temporal paradigm used to define individuals in ISO 15926, all classes are non-well-founded sets.

NOTE 3 Adapted from ISO 15926-1:2004, definition 3.1.1.

2.1.3
core class
class that is a commonly used subdivision corresponding to terms used in common language

NOTE The conditions for membership are often not formally defined; understanding of the class may be conveyed by example.

EXAMPLE Pipe, floor, pump, and light bulb are all core classes.

[ISO 15926-1:2004, definition 3.1.4]

2.1.4
core template
RDL template for which all reference data items in the expansion of its template axiom are core classes

[ISO 15926-7:2011, definition 2.1.6]

2.1.5
data store
computer system that allows data to be stored for future reference

[ISO 15926-1:2004, definition 3.1.6]

2.1.6
data type
domain of values

[ISO 10303-11:2004, definition 3.3.5]

2.1.7
data warehouse
data store in which related data are merged to provide an integrated set of data containing no duplication or redundancy of information, and which supports many different application viewpoints

[ISO 15926-1:2004, definition 3.1.7]

2.1.8
entity
class of information defined by common properties

[ISO 10303-11:2004, definition 3.3.6]

2.1.9
entity data type
representation of an entity

NOTE 1 An entity data type establishes a domain of values defined by common attributes and constraints.

NOTE 2 In this part of ISO 15926 the entity data types are as defined in the data model of ISO 15926-2.

NOTE 3 Adapted from ISO 10303-11:2004, definition 3.3.7.

2.1.10
entity instance
named unit of data which represents a unit of information within the class defined by an entity

NOTE 1 It is a member of the domain established by an entity data type.

NOTE 2 Adapted from ISO 10303-11:2004, definition 3.3.8.

2.1.11
first-order logic
symbolized reasoning in which each sentence, or statement, is broken down into a subject and a predicate

NOTE 1 The predicate modifies or defines the properties of the subject. In first-order logic, a predicate can only refer to a single subject.

NOTE 2 First-order logic is also known as first-order predicate calculus or first-order functional calculus.

[ISO 15926-7:2011, definition 2.1.13]

2.1.12
individual
possible_individual
thing that exists in space and time

NOTE 1 In this context, existence could be within the world we live in, or some "possible" world that can be imagined. This therefore includes actual, hypothetical, planned, expected, or required individuals.

EXAMPLE A pump with serial number ABC123, Battersea Power Station, Sir Joseph Whitworth, and the Starship Enterprise are examples of individuals.

NOTE 2 Adapted from ISO 15926-2:2003, definition 3.1.6.

2.1.13
individual template
template for making statements about individuals

[ISO/TS 15926-7:2011, definition 2.1.14]

2.1.14
instance
named value

[ISO 10303-11:2004, definition 3.3.10]

2.1.15
interoperability
ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful manner

2.1.16
life-cycle information
information about a **possible_individual** (2.1.12), collected at any point in time during the life-cycle of that individual

NOTE Adapted from ISO/TS 15926-7:2011, definition 2.1.17.

2.1.17
Manchester syntax
user-friendly compact syntax for OWL 2 ontologies which is frame-based, as opposed to the axiom-based other syntaxes for OWL 2

NOTE See Reference [21].

2.1.18
metadata
data that describes and defines other data

[ISO/IEC 11179-1:2004, definition 3.2.16]

2.1.19
N-triple
line-based, plain text format for encoding an RDF graph

2.1.20
object information model
OIM
ontology of classes and relations for which a particular class is singled out for characterization

NOTE In Description Logic terminology, it is a TBox (see Annex G).

2.1.21
ontology
<computer and information science> formal representation of a set of concepts within a domain and the relationships between those concepts

NOTE 1 Ontologies are usually used to reason about the properties of that domain, and can be used to define the domain.

NOTE 2 Ontologies are usually expressed in a logic-based language, but this is not a requirement, neither is the need for reasoning capability. In addition to relationships, classes, properties, instances and axioms can be used.

2.1.22
OWL native
modelling style in which a relationship is expressed as an RDF predicate

2.1.23
punning
declaring a class and an individual, having the exact same identifier, in order to use them in different model constructions, which would be illegal to use if applied to one object

NOTE Punning can also be applied to two property declarations of different property types.

2.1.24
RDF graph
graph structure formed by a set of RDF triples

2.1.25
RDF schema
language for describing vocabularies in RDF which is a semantic extension of RDF, providing mechanisms for describing groups of related resources and the relationships between these resources

2.1.26
RDF/XML
format with an XML syntax for RDF, as defined in the W3C recommendation "RDF/XML Syntax Specification (Revised)"

2.1.27
reference data
process plant life-cycle data that represents information about classes or individuals which are common to many process plants or of interest to many users

[ISO 15926-1:2004, definition 3.1.18]

2.1.28
reference data library
RDL
managed collection of reference data

[ISO 15926-1:2004, definition 3.1.19]

NOTE In this part of ISO 15926, "RDL" and "ontology" are used interchangeably.

2.1.29
reification
modelling style in which a relationship is expressed as an object class

EXAMPLE The relation Employed-by is reified by the object Employment which is connected to the objects Employee and Organization. The meaning of the relation with cardinalities at both ends is "an organization has zero or more employees". The reified Employment object can be subject in other relations, defining it.

NOTE The relational entity data types of ISO 15926 are all the entity data types which have exactly two attributes, except `class_of_relationship`.

[ISO/TS 15926-7:2011, definition 2.1.21]

2.1.30
signature
named, ordered and typed list of template roles

2.1.31
SPARQL endpoint
conformant to SPARQL protocol service as defined in W3C's SPARQL Protocol for RDF (SPROT)

NOTE In Reference [23], "endpoint" is defined as "An association between a fully specified InterfaceBinding and a network address, specified by a URI [IETF RFC 2396], that may be used to communicate with an instance of a Web Service. An endpoint indicates a specific location for accessing a Web Service using a specific protocol and data format."

2.1.32
specialized template
set of statements about individuals or classes, that is subclass of a core template or another specialized template, and which has one or more restrictions on its roles

2.1.33
taxonomy
collection of controlled vocabulary terms organized in a hierarchical structure, where each term is in one or more parent/child (broader/narrower) relationship to other terms in the taxonomy

2.1.34
template
set comprising of a first-order logic predicate for which a definition is stated as an axiom, a template signature and a template axiom expansion

[ISO/TS 15926-7:2011, definition 2.1.22]

2.1.35
template
n-ary predicate, represented in OWL reified form as a class with one functional property (*role*) per variable

2.1.36
template axiom
axiom in the template language defining the interpretation of template statements

[ISO/TS 15926-7:2011, definition 2.1.25]

2.1.37
template instance
ordered list of entity instances of which a template is true

NOTE 1 In OWL, the template instance is an individual with *role* relationships to the individuals that instantiate the template. The atomic/ground statement is made by replacing the variables of the template with OWL individuals.

NOTE 2 Adapted from ISO/TS 15926-7:2011, definition 2.1.26.

2.1.38
template language
axioms in first-order logic extending the ISO 15926-2 data model

[ISO/TS 15926-7:2011, definition 2.1.27]

2.1.39
template role
named and numbered argument in a template with required type given as entity data type, data type, or reference data class

NOTE Adapted from ISO/TS 15926-7:2011, definition 2.1.28.

2.1.40
template statement
statement made by instantiating the roles of a template with entity instances

[ISO/TS 15926-7:2011, definition 2.1.30]

2.1.41
core RDL
set of RDLs that only hold core classes and reference individuals

NOTE Part of the content is normalized.

2.1.42
triple
RDF triple
representation of a relation between the objects or data that it links

NOTE A triple comprises at least:

— an object called “subject”;

— a predicate (also called property) that denotes a relationship between a subject and an object;

— an object or data called “object”.

2.1.43
triple store
data store capable of storing **triples** (2.1.42)

2.1.44
value
unit of data

[ISO 10303-11:2004, definition 3.3.22]

2.2 Abbreviated terms

DL	description logic
FOL	first order logic
OIM	object information model
OWL	Web Ontology Language
RDF	Resource Definition Framework
RDFS	RDF Schema

RDL	reference data library
SPARQL	SPARQL Protocol and RDF Query Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

3 Fundamental concepts and assumptions

3.1 General

The ISO 15926-2 data model is generic and highly normalized. Whilst this enables considerable flexibility in what can be said, it can also give rise to complexity in how it is said. ISO/TS 15926-7 specifies templates that are expressions of predefined units of semantics allowing the use of the model in a convenient way.

This part of ISO 15926 translates these templates to OWL constructs, which can be used to perform information storage, data integration, information exchange and interoperability.

3.2 ISO/TS 15926-7 templates and first order logic rules

The ISO 15926-2 data model defines the foundation concepts. These are represented by a generic, conceptual data model that is suitable as the basis for implementation in a shared database or data warehouse. The data model is designed to be used in conjunction with reference data: standard instances that represent information common to a number of users, process plants, or both. The support for a specific life-cycle activity depends on the use of appropriate reference data in conjunction with the data model. In ISO/TS 15926-7, these are translated into first order logic, with EXPRESS entity data types represented as unary predicates and EXPRESS attributes represented as binary predicates.

For this part of ISO 15926, template signatures are the basic structuring elements. In this part of ISO 15926, the template signatures are translated to OWL, with reference to rules fulfilling the first order logic.

3.3 ISO/TS 15926-4 reference data

ISO/TS 15926-4 provides reference data that defines a taxonomy of core classes representing the applicable entity data types defined in ISO 15926-2.

For this part of ISO 15926, reference data is treated as constant terms for use in rules.

NOTE With this approach of having the attributes pointing at reference data instantiating the entity data types, the rule equations are simplified.

3.4 User-defined taxonomy

A user often needs to define discipline-specific concepts in the form of specializations of ISO/TS 15926-4 taxonomy core classes.

EXAMPLE CP-834833 “Model 4HM pump”, a specialized class in the supplier catalog of the XYZ Corporation. The rdfs:subclassOf relation points at the class “Centrifugal Pump” in the Core RDL.

3.5 Templates and specialized templates

Templates (see Clause 7) and specialized templates (see Clause 8) defined in this part of ISO 15926 and in user-defined specializations, form the “derived concepts” as described in ISO/TS 15926-7.

3.6 Use of OWL

This part of ISO 15926 is based on the use of RDF, RDFS, and OWL.

Annex C specifies rules for the use of these languages in the context of this part of ISO 15926.

NOTE RDF, RDF Schema, and OWL allow for various ways to represent the same semantics. In order to make the code more transparent and understandable, certain rules are necessary. These do not impede any linking to RDF/OWL vocabularies that are using OWL in a different manner.

3.7 Namespaces

An ISO 15926 reference data system using OWL is divided into a set of ontologies, corresponding to namespaces.

Table 1 shows the namespace prefixes with namespace URIs that are used for the examples in this part of ISO 15926, which are under the control of the World Wide Web Consortium.

namespace prefix	namespace URI	description
owl	http://www.w3.org/2006/12/owl2-xml#	OWL version 2
rdfs	http://www.w3.org/2000/01/rdf-schema#	RDF Schema
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	Resource Description Framework
xsd	http://www.w3.org/2001/XMLSchema#	XML Schema

Table 1 — URIs under control of the World Wide Web Consortium

Table 2 shows the namespace prefixes with namespace URIs that are used for the examples in this part of ISO 15926.

namespace prefix	namespace URI	description
dm	http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/data-model#	Normative OWL classes, according to the ISO 15926-2 data model and with some extensions.
p7tm	http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm#	Normative OWL declarations for templates and template roles (p7 means ISO/TS 15926-7).
p7tpl	http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#	Templates defined in ISO/TS 15926-7, proto-templates and "initial set" core templates. (p7 means ISO/TS 15926-7).
meta	http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata#	Metadata.

Table 2 — Namespaces that are used for the examples in this part of ISO 15926.

NOTE 1 The dm, p7tm, p7tpl and meta are files and not SPARQL endpoints. This is because given that these declarations are immutable, it can be assumed these files will be locally cached by implementing systems. These files are present on the CD-ROM of this part of ISO 15926.

NOTE 2 The following namespace (prefixes) with example.org addresses are used for the examples in this part of ISO 15926.

— rdl – http://rdl.example.org# – an endpoint having normative core classes and reference individuals, normalization-candidate classes, and specialized templates, known as the core reference data library "Core RDL";

— company1 – http://rdl.example.com# – an endpoint having examples of instances, specialized templates and classes, so to serve as example of a company's RDF store.

NOTE 3 The example namespaces are given the example.org and example.com addresses to show clearly that they are examples and therefore non-dereferencable. Also, the *rdf:ID* and *rdf:about* properties are made-up examples expressed in human-readable wording instead of the letter-numeric IDs that they will be in practice.

EXAMPLE 1 In Manchester syntax:

```
Prefix:  xsd:       <http://www.w3.org/2001/XMLSchema#>
Prefix:  owl:       <http://www.w3.org/2006/12/owl2-xml#>
Prefix:  rdf:       <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Prefix:  rdfs:      <http://www.w3.org/2000/01/rdf-schema#>

Prefix:  dm:        <http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/data-model#>
Prefix:  p7tm:      <http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm#>
Prefix:  p7tpl:     <http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#>
Prefix:  meta:      <http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata#>

Prefix:  rdl:       <http://rdl.example.org#>

Prefix:  steplib:   <http://www.steplib.com/data#>
Prefix:  pca:       <http://www.posccaesar.org/data#>

Prefix:  company1:  <http://rdl.example.com#>
```

EXAMPLE 2 When code examples are used, the RDF heading is omitted. The following is an example of a heading:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
    <!ENTITY owl "http://www.w3.org/2006/12/owl2-xml#">
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
    <!ENTITY dm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/data-model#">
    <!ENTITY p7tm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm#">
    <!ENTITY p7tpl "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#">
    <!ENTITY meta "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata#">
    <!ENTITY rdl "http://rdl.example.org#">
    <!ENTITY steplib "http://www.steplib.com/data#">
    <!ENTITY pca "http://www.posccaesar.org/data#">
    <!ENTITY company1 "http://rdl.example.com#">
]>
<rdf:RDF xmlns="http://rdl.example.com#"
     xml:base="http://rdl.example.com"
     xmlns:owl="&owl;"
     xmlns:xsd="&xsd;"
     xmlns:rdf="&rdf;"
     xmlns:rdfs="$rdfs;"
     xmlns:dm="&dm;"
     xmlns:p7tm="&p7tm;"
     xmlns:p7tpl="&p7tpl;"
     xmlns:meta="&meta;"
     xmlns:rdl="$rdl;"
     xmlns:steplib="&steplib;"
     xmlns:pca="&pca;"
     xmlns:company1="&company1;"
>
<owl:Ontology rdf:about="">
    <rdfs:comment>This ontology provides instances for
       various examples of specialized templates.
    </rdfs:comment>
    <owl:imports rdf:resource="http://rdl.example.org"/>
    <owl:imports rdf:resource="http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl.owl"/>
    <owl:imports rdf:resource="http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata.owl"/>
</owl:Ontology>

<!-- RDF XML here -->

</rdf:RDF>
```

These ontologies are interdependent, in the sense that one ontology will make use of classes defined in another. Figure 1 shows the dependencies, an arrow between prefixes representing an `owl:imports` relationship.

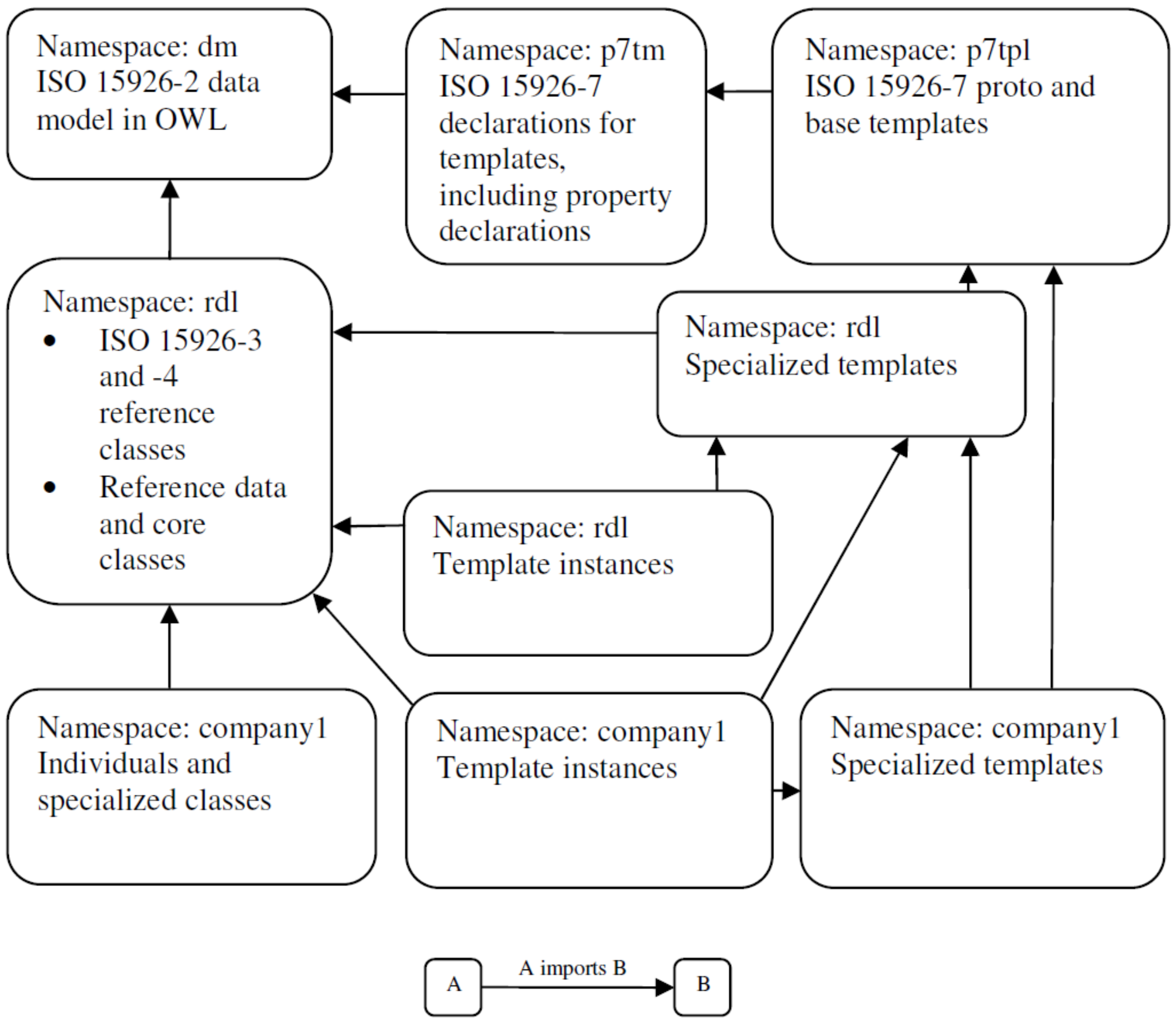
Namespace: dm
ISO 15926-2 data model in OWL
Namespace: p7tm
ISO 15926-7 declarations for templates, including property declarations
Namespace: p7tpl
ISO 15926-7 proto and base templates
Namespace: rdl
ISO 15926-3 and -4 reference classes
Reference data and core classes
Namespace: rdl
Specialized templates
Namespace: rdl
Template instances
Namespace: company1
Individuals and specialized classes
Namespace: company1
Template instances
Namespace: company1
Specialized templates
A
A imports B
B

Figure 1 — Dependencies between ontologies

NOTE 4 In this part of ISO 15926, there are no namespace declarations for OIMs. An OIM should be viewed as informal guides to usage rather than as an ontology in the OWL sense. This does not rule out representing OIMs as reference data modules and making them referable in a SPARQL endpoint. However, for clarifying the OWL/RDF representation of templates, normative representations of OIMs are not needed.

NOTE 5 Any of the namespaces shown in Figure 1 may be importing the meta namespace, as any **thing** can have metadata.

4 ISO 15926-2 data model

Based on the rules in Annex C, the data model of ISO 15926-2 has been mapped to OWL classes. It is defined in the OWL ontology for data model.

In Annex D, a link to the normative RDF/XML listing for the data model is given.

5 ISO/TS 15926-4 reference data

Reference data shall be represented as specified in Annex E.

NOTE ISO/TS 15926-4 contains reference data relevant to this part of ISO 15926.

6 OWL template specifications

6.1 General

A template specification (for templates see Clause 7) contains all the information that is necessary:

— for a human being to understand the content of instances of the template;

— to create the OWL and rules listing for the template, that is added to the ontology for templates (see ISO/TS 15926-7).

When a template is defined, it shall be in the format of template specifications as defined in 6.2.

For a link to an initial set of template specifications, see Annex F.

6.2 Contents

The contents of a template specification shall be:

— template name: a unique English CamelCase identifier of the lifted and lowered template (see 7.1 for lifted and lowered);

— name: the label of the template;

— intent: a brief description of the circumstances in which the template can be used;

— description: a full description of the semantics of the template.

— lifted and lowered graph: a graph that shows how the lifted template collects the nodes of the scope graph, and how the lowered template refers to the variant nodes of the lifted template;

— lifted template elements: an alphabetically ordered list of the URIs of all nodes as collected by the lifted template, with the applicable “elements” numbers;

— OWL code for lowered template: link to OWL ontology for templates;

— specification in First Order Logic - a first-order logic-based specification for the formalization of the lifted and the lowered template;

— sample lowered template instance: a sample instance of the lowered template in RDF/XML format;

— definition of properties for lowered template: a list of the properties of the lowered version of the template with a description of their *rdf:objects*.

NOTE For the lifted and lowered graphs a representation, as customary in the domain of RDF [24], or Manchester syntax [21] can be used.

7 Templates

7.1 General

For the purpose of this part of ISO 15926, information is represented as a complex relationship between independent objects. That complex relationship can be binary or n-ary.

The binary relationship is formed by an RDF triple's predicate.

EXAMPLE The relationship **specialization** is formed by *rdfs:subClassOf*. The relationship **classification** is formed by *rdf:type*.

The n-ary relationship is formed by a template and the independent objects are referred to as "external references".

Templates come in pairs:

— a lifted template (see 7.4), and

— a lowered template (see 7.5).

Where the lowered template provides the structure for data to be exchanged using the template, and the lifted template provides a definitional mapping of the lowered template to the data model and reference data of ISO 15926.

NOTE When just the word template is used, it is always a lowered template.

7.2 Template model

A template instance is member of a specialized template, by the *rdf:type* relationship.

A specialized template is a subclass of a specialized template or a core template, by the *rdfs:subclassOf* relationship.

A core template is a subclass of a template class, by the *rdfs:subclassOf* relationship.

A template class is a subclass of the Multidimensional Object class, by the *rdfs:subclassOf* relationship.

A template class is a subclass of the Information Representation class, by the *rdfs:subclassOf* relationship.

The Multidimensional Object class is of entity data type **class_of_multidimensional_object**, which has the following attributes:

— cardinalities : OPTIONAL LIST [1:?] OF **cardinality**;

— optional_element : LIST [1:?] OF BOOLEAN;

— parameters : OPTIONAL LIST [1:?] OF **thing**;

— parameter_position : OPTIONAL LIST [1:?] OF INTEGER;

— roles : LIST [1:?] OF **role_and_domain**.

NOTE Template class is more restricted than Multidimensional Object because ISO 15926-2 defines optional lists parameters and parameter_position, which are not used for templates. For the Information Representation class, there are no optional arguments in templates and the parameters and parameter_position lists are not used.

How it all fits together is shown in a series of examples (see Annex H).

7.3 Metadata

Classes, individuals or templates can have metadata.

NOTE Some metadata elements will be normalized in ISO/TS 15926-6, which is under preparation.

The following metadata elements for template instances are added in this part of ISO 15926:

— annRule;

— annAccessCode.

See Table F.2 for the definition of this metadata. The full set of metadata elements and their translation to OWL is given in Annex F.

7.4 Lifted templates

Lifted templates serve as a reference for the exact semantics of the lowered templates.

The instantiated lowered templates are lifted by using the First Order Logic axiom listed in the template specification.

NOTE 1 Lifted templates can be instantiated for reasons including, but not limited to the following:

— if the exchange format is done in full lifted format;

— for validation of entity data types of data connected to the template;

— if the full semantics it provides is required as subject for Semantic Web reasoning;

— if a mapping to a format, defined in ISO 10303-21 (so-called “Part 21 files”) would be required;

— if data needs to be mapped between templates where there would not be a handle available in lowered templates.

Lifted data can exist without the use of the lifted template information and metadata. Lowering such data will require pattern matching. However if a lowered template is lifted, the template information and metadata is also lifted, so that lowering that lifted template will result into the same lowered template without loss of data.

NOTE 2 The reason for having lifted data is to make the information computer interpretable. ISO 15926 stores data in an explicit way, and when that data is lifted, a computer can infer relations and conclusions about the data.

NOTE 3 Since the decision as to whether data storage and exchange is done using “lifted data” or by use of lowered template instances and objects only, is a business decision, and out of the scope of this part of ISO 15926. Both methodologies are supported.

NOTE 4 Instantiation of lifted templates leads to very verbose representations, but because it is possible to lift data on the fly based on instances of lowered templates, lifted data does not need to be persisted, unless for performance purposes.

A lifted template uses a set of classes defined in the OWL ontology for data model, or template-specific subclasses thereof. These subclasses are, for a given template, valid only in the context of that template.

NOTE 5 For an example of a lifted template see ISO/TS 15926-7.

7.5 Lowered templates

The lowered templates are instantiated and used for the representation of information about (OWL) individuals (instances of OWL classes).

NOTE An initial set of core templates is listed in ISO 15927-7.

7.5.1 Instantiating lowered templates

The definition of template classes is done with OWL constructs. Here the inheritance of, and the constraints for, the Properties are defined. Any instance of a template shall comply with these constraints.

NOTE OWL provides the means to define a schema for the instances, thus enforcing a certain rigour.

Any instance of a lowered template is related to its template class by means of the Property rdf:type.

In Annex H, lowered templates are instantiated. There the relationship between the *owl:Class*es and their instances is shown.

The *rdf:resource* of a property of the template instance shall be a member of the applicable *owl:Class* in the template class definition, or a subclass thereof.

EXAMPLE MPO-4983302 is a member of the *owl:Class* MaterializedPhysicalObject, so also of the *owl:Class* PhysicalObject, according the data model of ISO 15926-2.

The records of the instances can be located in different locations on the internet, but shall not be duplicated.

7.6 Templates in RDF

For the representation of template (signatures) and template (signature) instances in RDF/OWL a methodology is defined in Annex B.

7.7 Example: A proto-template

For an example of a proto-template see H.2.

7.8 Templates as template instances

7.8.1 p7tm classes for meta-templates

For representing templates (i.e., template signatures as data, the meta-templates TemplateDescription and TemplateRoleDescription are used. Here are the class definitions, according to the signature tables given in Clause B.2:

```
Class: p7tm:TemplateDescription
  SubClassOf:
      p7tm:MetaTemplateStatement,
      p7tm:hasTemplate some p7tm:Template,
      p7tm:valNumberOfRoles some integer

Class: p7tm:TemplateRoleDescription
```

```
SubClassOf:
   p7tm:MetaTemplateStatement,
   p7tm:hasRole some p7tm:TemplateRole,
   p7tm:hasTemplate some p7tm:Template,
   p7tm:hasRoleFillerType some Class,
   p7tm:valRoleIndex some integer
```

EXAMPLE For an example of a statement using the template, see H.2.2.

8 Object information models (OIMs) and specialized templates

8.1 General

The application of object information models are informative in this part of ISO 15926. They are not required for ISO/TS 15926-8 compliancy. OIMs can serve as a guidance for specialized template modelling.

In the context of this part of ISO 15926, the OIM methodology is considered to be a part of ISO 15926, but the actual definition of the individual OIMs adding ontology to taxonomy, if centralized and normalized, is considered part of the Core RDL. If not centralized, user-defined ontologies can exist elsewhere, and could become candidates for centralization.

An OIM is an ontology of classes and relations (in Description Logic terminology, a TBox see Annex G) for which a particular class is singled out for characterization. An OIM is a model in the full sense of the word. An OIM specifies what information shall and may be provided to describe a member of the designated class. Typically, it builds on a fragment of a larger ontology (a reference data library), and adds constraints to support a particular application.

Specialized templates are a set of OWL statements about individuals or classes (an ABox, see Annex G) made according to an OIM in the role of a guidance. Specialized Templates are meant to instantiate an OIM, but are themselves OWL schemas.

There is no technical coupling between specialized templates and OIMs.

A specialized template characterizes an individual of the OIMs designated class, using only the vocabulary of classes and relations that is used in the OIM. Specialized templates populates the OIM by providing explicit instances for every existential quantifier, and by not violating any universal constraint.

NOTE 1 Specialized templates may well provide information which isn't obligatory according to the OIM. Where the OIM contains 0..$*$ cardinalities, this will typically be the case.

The business use of OIMs and specialized templates is best explained in an example:

EXAMPLE Example of not using OIMs: Company A exchanges data with Company B. The engineering database system of A contains an instance of a centrifugal pump, and a property impeller diameter. The ISO 15926 interface maps this data into the following statements representing template instances:

Object P12345 is a Centrifugal Pump
Object P12345 has a Impeller Diameter of 51 mm

Company B also mapped this kind of data for import into their database system. Their mapping expects something like the following instances:

Object P12345 is a Centrifugal Pump
Object P12345 has a assembly P56789
Object P56789 is a Impeller
Object P56789 has a Impeller Diameter of 51 mm

These two models are both ISO 15926 compliant. However they cannot be easily interfaced. In order to import the data, Company B needs to write a mapping rule. They may discover that they have to write this mapping rule at the time that the data from Company A is released when they get an error message. They may have to scramble to make the interface work. The risk involved is high.

This example shows that if there exist OIMs, companies could use those conceptual models to determine the architecture of their ISO 15926 interface mapping. Systems would connect to each other without the need for prior human involvement to make mapping rules.

Secondly, at the moment companies have to spend work to create their interface mapping, consulting the object information models will save them work, because they can see how to make their mapping models instead of inventing them.

NOTE 2 If the specialized templates are stored in a central store, like the Core RDL or a store central to a project, it serves the purpose of an OIM just as well. In that case OIMs are optional since the risk is mitigated. All mapping will be following the specialized template structures.

NOTE 3 OIMs are the result of consultations and consensus between domain experts. Since such consultations will be an ongoing activity, the OWL ontology for object information models will grow over time.

NOTE 4 Mapping data in any user system to the format defined in this part of ISO 15926 can be done as a deterministic activity, in case the data are already defined in the applicable OIM.

NOTE 5 In principle an OIM is all-inclusive, but in practice its contents will be determined by business needs.

In Annex H the RDF/OWL listings of example OIMs ontology is given.

8.2 OIM versus specialized template construct

The OIM is not constrained to a specific language, while the specialized template shall be expressed in RDF/OWL using core templates as normalized by this part of ISO 15926.

The OIM serves as requirements how to map and model the data in specialized templates.

NOTE The OIM could be expressed in UML or native OWL. The examples in Annex H are in native OWL.

8.3 Specialization

Any *owl:Class* inherits all *owl:ObjectProperties* from its superclass. In the context of OIMs this inheritance means that the OIM of the superclass is inherited. The inherited relationships can, where necessary, be further constrained.

The relationships that apply only to the subclass are added to the relationships inherited from the superclass.

EXAMPLE The owl:Class RDS327239 “pump” is involved in a relationship that tells that a pump has between 2 and 6 bearings. The subclass RDS416834 “centrifugal pump” inherits that relationship, but it is constrained by reducing the cardinality to 2 to 3 bearings.

8.4 Cardinalities

There are two types of cardinalities:

— the number of allowed template instances connected to the instance of an *owl:Class* or its children;

— the number of objects instanced as target of a role in an instanced template.

Both these types of cardinality can be handled by cardinality statements in an OIM using the modelling language of that OIM.

Cardinality statements are only required if the cardinality is not:

— 0..1 for allowed template instances;

— 1..1 for allowed instanced role targets.

8.5 OIM for assemblies

Information about an object that is a part of another object does not belong to the OIM of which it is part.

NOTE The "whole" object does not participate directly in the relationships representing information of the "part" object.

EXAMPLE The diameter of the impeller of a centrifugal pump does not belong to the OIM of that pump, but to that of the impeller. The template with the information that the impeller is or can be a part of a centrifugal pump is both part of the OIM of the impeller and of the OIM of the pump.

8.6 Naming convention on template roles

For specialized templates, the roles shall follow a naming convention.

The name of a role shall be prefixed with:

— "has" for object properties;

— "val" for data properties;

— "ann" for annotation properties.

EXAMPLE The ISO 15926-7 role "Property value" is declared as:

```
<owl:DatatypeProperty rdf:about="#valPropertyValue">
  <rdfs:label>Property value</rdfs:label>
  <rdfs:comment></rdfs:comment>
  <rdfs:subPropertyOf rdf:resource="#valInitialSetTemplateDataRoleFiller"/>
</owl:DatatypeProperty>
```

There are cases of the same role name being used in, once translated from ISO 15926-2 EXPRESS into OWL, both object and data property senses. This applies to the proto templates, where the role names given in ISO 15926-2 shall be followed. In general, to prevent *owl:Class* names and *properties* having the same name, *properties* shall be prefixed with "has", "val" and "ann". Also, OWL's strict separation between object properties and data properties is satisfied.

8.7 Uniqueness context of template roles

In this part of ISO 15926, in the text and in the attached files, the declarations of roles are unique in the context of the namespace. The range of roles per template are defined in the template declarations.

However, it is valid to have sub properties of roles having the same rdfs:label such that they are unique in the context of a template declaration.

EXAMPLE The role “Property value” is declared as:

```
<owl:DatatypeProperty rdf:about="#role_925636494">
  <rdfs:label>Property value</rdfs:label>
  <rdfs:comment></rdfs:comment>
  <rdfs:subPropertyOf rdf:resource="#valPropertyValue"/>
</owl:DatatypeProperty>
```

Annex A
(normative)
Information object registration

A.1 Document identification

To provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 15926 part(8) version(1) }

is assigned to this part of ISO 15926. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

A.2 Schema identification

A.2.1 OWL ontology for data model

To provide for unambiguous identification of the OWL ontology for data model in an open information system, the object identifier

{ iso standard 15926 part(8) version(1) object(2) owl-ontology-for-data-model(1) }

is assigned to the OWL ontology for data model (see Clause 4). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

A.2.2 Template specifications

To provide for unambiguous identification of the template specifications in an open information system, the object identifier

{ iso standard 15926 part(8) version(1) object(2) template-specifications(2) }

is assigned to the template specifications (see Clause 6). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

A.2.3 OWL ontology for templates

To provide for unambiguous identification of the OWL ontology for templates in an open information system, the object identifier

{ iso standard 15926 part(8) version(1) object(2) owl-ontology-for-templates(3) }

is assigned to the OWL ontology for templates (see Clause 7). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

A.2.4 OWL ontology for reference data

To provide for unambiguous identification of the OWL ontology for object information models in an open information system, the object identifier

{ iso standard 15926 part(8) version(1) object(2) owl-ontology-for-reference-data(4) } is assigned to the OWL ontology for reference data (see Clause 5). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

A.2.5 OWL ontology for metadata

To provide for unambiguous identification of the OWL ontology for metadata in an open information system, the object identifier

{ iso standard 15926 part(8) version(1) object(2) owl-ontology-for-metadata(5) }

is assigned to the OWL ontology for metadata (see Clause 7.3). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex B
(normative)
Templates as RDF/OWL n-ary relations

B.1 General

For the representation of template (signatures) and template (signature) instances in RDF/OWL, a representation described in *Pattern 1, Use Case 3* of Reference [22] is adopted.

An example given in that note is that of a *Purchase*, which involves a buyer, a seller, an object, a purpose, and an amount, all modelled as shown in this illustration[1]:

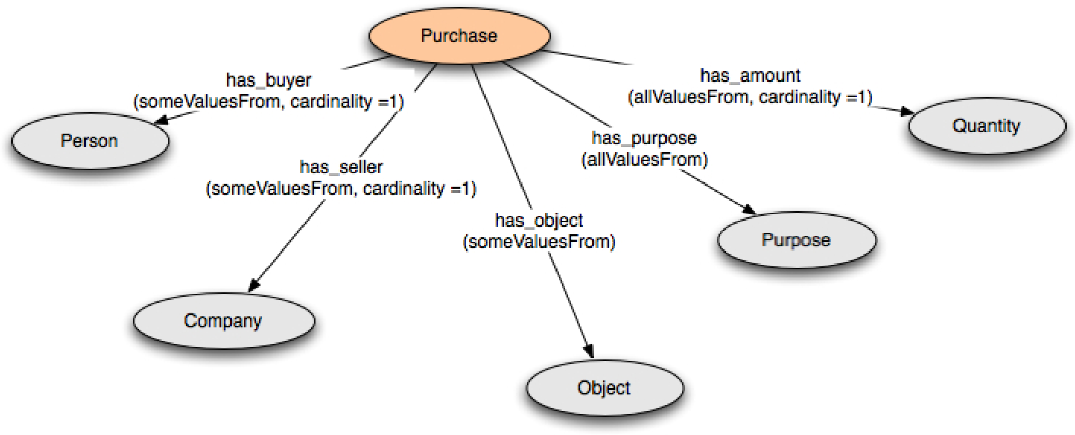

Figure B.1 — W3C's working group note: defining n-ary relations

There are differences of this representation of the *Purchase* relation to ISO 15926 templates. In the example:

a) the whole construction for *Purchase* is analogous to an ISO 15926 template *signature*.

NOTE 1 In the W3C note, as for a signature in ISO 15926, no consideration is given to modelling how the participants in the reified relation are themselves related. In ISO/TS 15926-7 terms, no axiom is given to give an explicit representation of *Purchase*;

b) some roles are optional. This is not allowed for ISO 15926 templates.

NOTE 2 The main reason for disallowing optional roles is that they severely increase the requirements on processing;

c) there is no expression that says exactly which roles are involved in a *Purchase* n-ary relation;

d) the *Purchase* class, as any ISO 15926 template instance, represents a statement;

NOTE 3 From the Note, Use Case 3: "we create an individual to represent the relation instance with links to all participants". It is not the purchase, but the relation, which is represented.

Point c) ties in with the issue of *numbering* for roles in ISO 15926 templates. In the Note example, no attempt is made to be explicit about precisely which "attributes" are involved in the definition of

[1]The original image is found at [22].

Purchase. The definition is therefore open-ended. This, in contrast to ISO 15926 templates, for which the list of roles is always given as a numbered, finite list.

Point d) is quite crucial for the structure of n-ary relationships, in that the members of the *Purchase* class (and in the same way for instances of an ISO 15926 template), are not meant to represent purchases, but rather statements that describe purchases. For instance *it would be inappropriate to use the identifier for a* Purchase *instance to refer to a particular purchase*. If an identifier for purchases is wanted, it shall be added to the n-ary relation as a role. This observation is made in the W3C Note [22], but could be given more emphasis:

> In most cases, these individuals do not stand on their own but merely function as auxiliaries to group together other objects. Hence a distinguishing name serves no purpose.
>
> (*ibid.*, Use case 3, "Considerations")

The distinction comes out particularly clearly if we consider the fact that many different statements (template instances) can be made about a particular purchase. The *Purchase* class of the W3C Note is not well suited for this usage, since it does not include an Identifier role for a purchase. This should be taken as a lesson for modelling templates: Anything that one may want to talk about in template statements needs to have a designated role. (Its only in exceptional cases that we will want to talk about statements themselves.)

In working with this representation, be aware that OWL does not have all the desirable resources for defining identity of instances. In a relational database, clearly there would not be two distinct rows with all the same values. This is however not ruled out by the Note example, and neither can it be for an ISO 15926 representation in OWL (although the numbering representation will approximate it; more below).

For a full expression of identity criteria, first order logic is needed.

In the ISO 15926 context, a reasonable assumption is that identity will be handled at an implementation/application level, rather than by constructs in the semantic modelling itself. This will be in line with common Semantic Web practice.

B.2 ISO/TS 15926-8 constructs for template signatures

This clause defines a small ontology for the representation of templates. For the example files accompanying this document, the ISO/TS 15926-7 template model ontology is given the namespace:

```
http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm
```

with shorthand prefix "p7tm".

The classes and relations of p7tpl that cover template signature representations are the following:

`p7tm:TemplateStatement` This `owl:Class` is the ultimate superclass of all templates. The suffix "Statement" is significant, since it emphasizes that instances of templates represent statements. The proper dm superclass of `p7tm:TemplateStatement` is `dm:MultidimensionalObject`. Subclasses of `p7tm:TemplateStatement` are:

p7tm:BaseTemplateStatement – used as superclass of all base templates, which in turn are used as superclass for core templates.

p7tm:MetaTemplateStatement – model structure for all templates. with subclasses:

p7tm:TemplateDescription

p7tm:TemplateRoleDescription

p7tm:TemplateSpecialization

`p7tm:hasObjectRoleFiller` This `owl:ObjectProperty` is the ultimate superrelation of all roles that take individuals as role-fillers.

NOTE 1 There is no natural dm superrelation for this relation, because in ISO 15926-2, with `dm:MultidimensionalObject`, the corresponding entitites are represented as lists. This relation is a "technical" device that supports the representation of n-ary predicates, but which has no natural interpretation in terms of ISO 15926-2.

`p7tm:hasDataRoleFiller` This `owl:DatatypeProperty` is the ultimate superrelation of all roles that take datatype values.

NOTE 2 As with `p7tm:hasObjectRoleFiller`, there is no natural dm superrelation for this relation.

A template signature will be a subclass of `p7tm:TemplateStatement`.

There is also a need for representing information about templates. In particular, the number of roles in a template, and the position of each role shall be represented. For this, templates need to be represented as individuals, for which the *punning* mechanism of OWL 2 is suitable. p7tm defines two templates for describing templates and their roles.

`p7tm:TemplateDescription` With roles *template* and *numberOfRoles*, this template is suitable for describing a template.

NOTE 3 Name, definition and other characteristics of the template can be found elsewhere, maybe using generic identification templates.

Order	Role name	Role type
1	p7tm:hasTemplate	p7tm:Template
2	p7tm:valNumberOfRoles	xsd:integer

`p7tm:TemplateRoleDescription` With roles *template*, *positionInTemplate*, *role*, and *type*, this template is suitable for describing a role within a template.

Order	Role name	Role type
1	p7tm:hasRole	p7tm:TemplateRole
2	p7tm:hasTemplate	p7tm:Template
3	p7tm:valvalRoleIndex	xsd:integer
4	p7tm:hasRoleFillerType	dm:Class

These "meta-templates" provide a way to represent templates themselves as data.

NOTE 4 One should be aware that they represent a potential maintenance problem: If the definition of a template class changes, all meta-template instances that describe that template need to be updated accordingly.

The *template specialization* is represented as:

`p7tm:TemplateSpecialization` For expressing that one template is a specialization of another.

NOTE 5 This implies OWL subclassing, but OWL subclassing does not imply template specialization as understood in ISO/TS 15926-8.

Order	Role name	Role type
1	p7tm:hasSuperTemplate	p7tm:Template
2	p7tm:hasSubTemplate	p7tm:Template

Annex C
(normative)
Rules for usage of OWL

C.1 OWL native representation

The representation of ISO 15926 content is assumed to be *OWL native*, representing entity data types as OWL classes and properties.

NOTE 1 This is in contrast to a *reified* format of ISO 15926-2 in which both classes and properties are represented as individuals. The reified format represents ISO-15926-2 data model as given in the EXPRESS language, but that is not suitable for the use cases intended for template formats, as shown in this ISO 15926 part.

The ISO 15926-2 data model/upper ontology translated to OWL can be found in Annex D.

The following figures illustrate some aspects of an OWL native representation of ISO 15926. Relations are *not reified*, but replaced with simple binary relations (OWL object and data properties). Instead of the construct in Figure C.1,

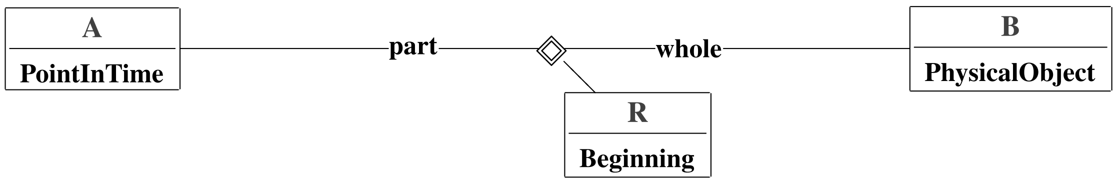

Figure C.1 — Reified relationship

where every Beginning relationship is represented (reified) by an individual member of the Beginning class, the following pattern in Figure C.2 is introduced:

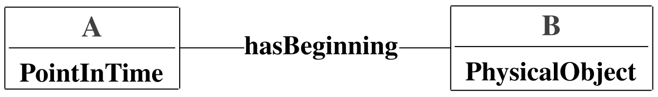

Figure C.2 — OWL-native relationship

NOTE 2 The native OWL representation puts greater demands on the modeller to be explicit about semantics than does a reified approach. With reification, an entity can appear in both individual, class, and relation roles; with the native representation, just one has to be chosen. Because certain ISO 15926-2 constructs rely on the flexibility of reification, a translation into OWL native form is necessarily incomplete. Arguably, this is no great disadvantage, since OWL-native constructs are more widely accepted and can count on a wide range of tools using it as a basis.

The following are the main features of the OWL native representation.

entity data types as classes: The ISO 15926-2 entity data types that are naturally represented by unary predicates are given an "OWL native" representation as classes (`owl:Class`);

entity data types as relations: The ISO 15926-2 entity data types **ClassOfClassOfRelationship**, **ClassOfRelationship**, and **Relationship** are naturally represented by binary predicates (binary relations). These are represented as OWL relations, of which there are two kinds:

individual–individual relations: The entity data types that relate pairs of individuals are typed as `owl:ObjectProperty`;

individual–data value relations: The entity data types (**ClassOfRepresentationOfThing** and its subtypes) that relate individuals to data values are typed as `owl:DatatypeProperty`.

C.2 Template translation from EXPRESS-native to OWL-native instance model

When implementing lowered templates using the axioms of ISO/TS 15926-7, the template roles are used as Predicate properties. The external data as the range of these Object or Data Properties and the result is native OWL.

In the case of lifted templates, the template expansion as normalized in ISO/TS 15926-7 results in an OWL-native representation of the instance model.

NOTE Both methods do not require this part of ISO 15926 to dictate how to translate from an EXPRESS instance model to one that is OWL-native.

C.3 Library object translation from EXPRESS-native to OWL-native instance model

For the library objects like classes and reference data, in the Core RDL as well as in other data stores, the following clauses show how to translate them.

The mapping from the ISO 15926-2 data model to OWL is not a subject here, because this translation has already been done and is available as endpoint (see Annex D).

When mapping the ISO 15926-2 data model to OWL ontology some incompatibilities between the two languages had to be dealt with.

In this annex, the mapping from EXPRESS constructs used in ISO 15926-2 to OWL constructs, including some workarounds for above incompatibilities is defined.

RDF and OWL offer, in some cases, more than one way to represent certain information. In this annex, alternative representations are prohibited. Specific representation is specified.

NOTE Links to OWL vocabularies with different, but valid, representations remain possible.

C.4 owl:Class

For each entity data type in the ISO 15926-2 data model an instance of *owl:Class* has been created.

Subtyping in EXPRESS shall be mapped to the *rdfs:subClassOf* construct.

EXAMPLE 1 The fact that, in the ISO 15926-2 data model's EXPRESS schema, **point_in_time** is a subtype of **event** is represented in OWL as:

```
<owl:Class rdf:ID="PointInTime">
      <rdfs:subClassOf rdf:resource="#Event"/>
</owl:Class>
```

owl:ObjectProperties and *owl:DataTypeProperties* are inherited from the superclass see C.8. Multiple inheritance is supported by OWL.

NOTE 1 owl:AnnotationProperties are not inherited from the superclass.

An instance of an *owl:Class* can be part of an RDF/XML file. This also applies to any member of an instance of an *owl:Class*.

NOTE 2 In OWL this member is called "individual", not to be confused with possible_individual.

C.5 Individual

An *individual* in OWL shall be an instance of an *owl:Class*.

NOTE There may be confusion about the difference between an "instance" and a "member" of an *owl:Class*. In this part of ISO 15926 an "instance" indicates the applicable ISO 15926-2 entity data type, and a "member" indicates to which set of things, that have certain characteristics in common, the thing belongs by complying with the criteria for membership of that set of things.

C.6 Property

A *Property* in RDF is a name for the predicate in an RDF *triple*. It gives information about the *subject* of that *triple* by referring to the *object* of that *triple*.

NOTE The concept of **property** (the ISO 15926-2 entity data type) is different from *property* (the OWL relation).

In OWL, three kinds of *Property* have been defined:

— *owl:ObjectProperty*;

— *owl:DatatypeProperty*;

— *owl:AnnotationProperty*.

C.6.1 Object property

An *owl:ObjectProperty* characterizes a *subject individual* by making reference to an *object individual*.

NOTE It is important to notice that only instances of OWL *classes* can be the *subject* or *object* of an *owl:ObjectProperty*.

C.6.2 Datatype property

An *owl:DatatypeProperty* characterizes a subject individual by making reference to an object XML Schema datatype value.

For strings, numbers, date-times, or other data types an instance of the applicable XML Schema data type (see [18]), as made part of the Reference Data Library of ISO/TS 15926-4, shall be used.

owl:DatatypeProperty shall be used for:

— in lifted data, for the ISO 15926-2 holder of literals such as *ClassOfInformationRepresentation* with attribute "content";

— roles of lowered templates referring literals, like strings and numbers.

C.6.3 Annotation property

An *owl:AnnotationProperty* represents metadata of a *subject Class* or *individual* by making reference to an *object URI Reference*. See Annex F for its use.

C.7 Identifiers

C.7.1 rdf:ID

Every instance of an entity data type defined in the ISO 15926-2 data model has a unique **id** attribute, inherited from **thing**.

In an RDF/XML document claiming conformance to this part of ISO 15926, the representation of an **id** attribute by means of an *rdf:ID* attribute shall be of the RDF URI References format, as defined in [15].

In this part of ISO 15926, the *rdf:ID* attributes are preferred to be of the URI#fragment-identifier format.

NOTE 1 This part of ISO 15926 cannot dictate that all IDs must be URI#fragment-identifiers. IDs from other parts and other sources can also be URIs (e.g. no # in it). However if a choice can be made, the URI#fragment-identifier is preferred. See the article at [19].

NOTE 2 The constraint invoked by the use of *rdf:ID*, is that the values of all the attributes and elements that have an ID datatype in a document shall be unique. If the identifier used is not unique because it is a reference to another *rdf:ID* either inside or outside of the data store, **rdf:about** shall be used. An extract from [14] says:

> The *rdf:ID* attribute on a node element (not property element, that has another meaning) can be used instead of *rdf:about* and gives a relative RDF URI reference equivalent to # concatenated with the *rdf:ID* attribute value. So for example if rdf:ID="name", that would be equivalent to rdf:about="#name". *rdf:ID* provides an additional check since the same name can only appear once in the scope of an xml:base value (or document, if none is given), so is useful for defining a set of distinct, related terms relative to the same RDF URI reference.

NOTE 3 The datatype of *rdf:ID* is NCName. An extract from [18] Clause 2 says:

```
NCName    ::=    (Letter | '_') (NCNameChar)*
NCNameChar    ::=    Letter | Digit | '.' | '-' | '_' | CombiningChar | Extender
```

In short, *rdf:ID* shall begin with a letter, then is allowed to contain letters and digits, dot, dash and underbar, and is not allowed to contain any colons. For the full definition refer “Extensible Markup Language (XML) 1.0 (Fifth Edition)”.

C.7.2 rdf:about

The RDF Production syntaxTerm *rdf:about* shall be for use of reoccurrence of an object at a position where it is not defined or inside a dataset where the object data is not self-owned.

NOTE Where an *rdf:ID* is used to declare a class or individual, *rdf:about* is used to add some information about that declared class or individual.

C.7.3 URI reference for handed-over objects

In case an object is handed over to another data store the URI Reference shall not change. It shall remain to have the same URI and unique ID within that URI.

NOTE 1 For data created by companies in the supply chain of a project or by the project’s EPC contractors and subcontractors the URI identifiers should be the URI of the final RDF data store after “as built” data handover, i.e. the URI of the plant system or unit at the client.

NOTE 2 The data created in the workflow preceding the data handover will be located in local RDF stores which forms a problem because linked data URIs need to be dereferencable (i.e. the URIs must respond to give data). Refer to W3C best practices to overcome this problem, for example refer [19].

C.8 Inheritance

An *owl:Class* that is related to another *owl:Class* by means of a *rdfs:subClassOf Property* inherits the definition, by inheriting the relationships to the specialized templates of the latter.

Excepted from inheritance are any instances of *owl:AnnotationProperty* (see Annex F, which is used for meta information about the *owl:Class*.

A member of an *owl:Class* that is related to that *owl:Class* by means of an *rdf:type Property* must comply with the constraints of the definition of that *owl:Class*.

C.8.1 Declaration

Where in OWL resources are normally not declared, this is mandatory in this part of ISO 15926.

The only exception is that any temporal part, being the *rdf:object* of a template property, shall not be explicitly declared.

NOTE A "temporal part" is defined in the ISO 15926-2 data model in the context of the relationship called **temporal_whole_part**. A temporal part inherits its essential classification from its temporal whole. Explicit declaration could generate redundancy, and potentially data corruption.

Annex D
(normative)
Computer interpretable listings

D.1 Ontology for data model

The listing referred to below is the result of mapping the data model defined in ISO 15926-2 to OWL.

In case mapping rules in this part of ISO 15926 contradict the mapping result as defined in the OWL file referred below, the OWL file definition has priority.

The complete data model is present on the CD-ROM of this part of ISO 15926, as an RDF file called "data-model.owl".

D.2 Ontology for template model

The listing referred to below comprises the declarations needed by template constructs.

In case mapping rules in this part of ISO 15926 contradict the mapping result of below OWL file, the OWL file has priority.

The complete template model is present on the CD-ROM of this part of ISO 15926, as an RDF file called "p7tm.owl".

D.3 Ontology for proto templates and templates initial set

The listing referred to below comprises the template constructs of proto templates and the template initial set as defined in ISO/TS 15926-7.

In case mapping rules in this part of ISO 15926 contradict the mapping result of below OWL file, the OWL file has priority.

The listing is present on the CD-ROM of this part of ISO 15926, as an RDF file called "p7tpl.owl".

D.4 Ontology for metadata

The listing referred to below contains some example metadata as well as metadata defined in this part of ISO 15926.

In case mapping rules in this part of ISO 15926 contradict the mapping result of below OWL file, the OWL file has priority.

The listing is present on the CD-ROM of this part of ISO 15926, as an RDF file called "metadata.owl".

Annex E
(informative)
Pattern for an ontology for reference data

RDL classes are defined by their metadata (see Annex F) and template instances. In this annex an example of an RDL core class is given with some metadata.

EXAMPLE The instance of a class called "centrifugal pump":

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
<!ENTITY owl "http://www.w3.org/2006/12/owl2-xml#">
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
<!ENTITY dm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/data-model#">
<!ENTITY p7tm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm#">
<!ENTITY p7tpl "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#">
<!ENTITY meta "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata#">
<!ENTITY rdl "http://standards.iso.org/iso/15926/tech/reference-data#">
<!ENTITY steplib "http://www.steplib.com/data#">
<!ENTITY pca "http://www.posccaesar.org/data#">
<!ENTITY company1 "http://rdl.example.com#">
]>
<rdf:RDF xmlns="http://standards.iso.org/iso/15926/tech/reference-data#"
    xml:base="http://standards.iso.org/iso/15926/tech/reference-data"
    xmlns:owl="&owl;"
    xmlns:xsd="&xsd;"
    xmlns:rdf="&rdf;"
    xmlns:rdfs="$rdfs;"
    xmlns:dm="&dm;"
    xmlns:p7tm="&p7tm;"
    xmlns:p7tpl="&p7tpl;"
    xmlns:meta="&meta;"
    xmlns:rdl="&rdl;"
    xmlns:steplib="&steplib;"
    xmlns:pca="&pca;"
>

<owl:Class rdf:ID="#RDL7436"> <!-- from unique number -->
   <rdfs:subClassOf rdf:resource="#RDL7611"/> <!-- dynamic pump -->
   <rdf:type rdf:resource="&dm;ClassOfInanimatePhysicalObject"/>
   <rdf:type rdf:resource="#RDL1957"/> <!-- rotating equipment class -->
   <meta:annUniqueName>centrifugal pump</meta:annUniqueName>
   <meta:annTextDefinition rdf:datatype="&xsd;string">
      A dynamic pump that contains impellers provided with vanes to generate
      centrifugal force to achieve the required pressure head.
   </meta:annTextDefinition>
   <steplib:identifier rdf:resource="&steplib;SL130058"/>
   <pca:identifier rdf:resource="&pca;RDS416834"/>
   <meta:annCreationDate rdf:datatype="&xsd;dateTime">
      1999-07-01T21:32:52Z
   </meta:annCreationDate>
   <meta:annRegistrationStatus rdf:datatype="&xsd;string">
      IS
   </meta:annRegistrationStatus>
</owl:Class>

</rdf:RDF>
```

Annex F
(normative)
Metadata

F.1 General

In this part of ISO 15926, metadata is used to annotate the provenance of instanced and persisted objects, or to annotate a security group, or a rule. If data related to an object is not intended for provenance, security or rule, it shall be modelled not as annotation but in the usual way, using templates and other constructions.

NOTE 1 Some metadata for provenance will be published in ISO/TS 15926-6, which is under preparation, therefore it will not be referred to normatively.

NOTE 2 Metadata is optional.

NOTE 3 Metadata can be properties of:

— classes;

— instances;

— templates.

NOTE 4 Since the metadata is property of a specific object and not to its children, it is declared such that no inheritance is invoked or intended.

NOTE 5 As defined in OWL, the object of an annotation property must be either a data literal, a URI reference, or an individual.

NOTE 6 Annotation properties are typically ignored by Reasoners.

F.2 Metadata declarations for provenance

If metadata is declared that:

— conforms the intended use of metadata for provenance,

— refers to a literal (like string or number),

— has not an intended use as *rdfs:label*, *rdfs:seeAlso* and *rdfs:comment*,

then the exact names shall be used as IDs in an annotation property declaration, with spaces and underscores removed, prefixed with "ann", in CamelCase. Exception to this is the property intended to define a unique number for an object, which shall be directly mapped to *rdf:ID* and *rdf:about*.

EXAMPLE 1 Name of metadata:

creation date

Declaration as annotation property:

```
<owl:AnnotationProperty rdf:ID="annCreationDate" />
```

If metadata is declared that:

— conforms the intended use of metadata for provenance,

— has an intended use as *rdfs:label*, *rdfs:seeAlso* and *rdfs:comment*,

then the exact names shall be used as IDs in an annotation property declaration, with spaces and underscores removed, prefixed with “ann” if referring to a literal, “has” if referring to an object, in CamelCase, and it shall be declared with a rdf:subPropertyOf relationship.

EXAMPLE 2 Metadata properties having an intended use as *rdfs:label*, *rdfs:seeAlso* and *rdfs:comment* listed as example:

Table F.1 — Examples of properties that are rdf:subPropertyOf OWL standard annotation properties

name	metadata property	rdf:subPropertyOf
unique name	meta:annUniqueName	rdfs:label
source	meta:hasSource	rdfs:seeAlso
text definition	meta:annTextDefinition	rdfs:comment
notes	meta:annNotes	rdfs:comment
administrative note	meta:annAdministrativeNote	rdfs:comment
explanatory comment	meta:annExplanatoryComment	rdfs:comment
change description	meta:annChangeDescription	rdfs:comment

F.3 Metadata declarations for security and rules

Table F.2 shows the metadata property declarations of the metadata that is defined in clause 7.3.

Table F.2 — Metadata elements from ISO/TS 15926-8

name	OWL property declaration	cardinality	description
rule	<owl:AnnotationProperty rdf:ID=”#annRule”> <rdfs:domain rdf:resource=”#TemplateStatement”/> </owl:AnnotationProperty>	0-many	A string that defines any rules, methods, or scripts that are applicable in any application software.
access code	<owl:AnnotationProperty rdf:ID=”#annAccessCode”> <rdfs:domain rdf:resource=”#TemplateStatement”/> </owl:AnnotationProperty>	0-many	A string giving the security code (resource group) of the information represented by an object. It is used as a parameter in the ACL (Access Control List).

NOTE Domain is specified at the annotation property declaration as a TemplateStatement or subclass thereof. That means that these two annotation properties can only be used for templates. Range is not specified at the annotation property declaration. That means that the property can point at a data literal, a URI reference, or an individual.

Annex G
(informative)
Assertion versus terminological components

An ABox is an assertion component — a fact associated with a terminological vocabulary within a knowledge base.

A TBox is a terminological component — a vocabulary associated with a set of facts ABox.

The terms ABox and TBox are used to describe two different types of statements in ontologies. TBox statements describe a system in terms of controlled vocabularies, for example, a set of classes and properties.

ABox are TBox-compliant statements about that vocabulary.

TBox statements tend to be more definitional in nature.

EXAMPLE A dictionary of words is a TBox.

NOTE ABox statements typically have the form:

A is an instance of B
or
P-101 is classified as Pump
or
The maximum design pressure of P-101 is 33 barg.

This should be contrasted with TBox statements or (statements) about terminology such as

All Fire Pumps are Pumps
or
There can be two types of Fire Pumps: Reciprocating or Centrifugal
or
Cooling Water Pumps can have an inlet temperature of 0 to 18 degrees Centigrade

TBox statements tend to be more permanent within a knowledge base and tend to be stored in a data model or a metadata registry. In contrast, ABox statements are much more dynamic in nature and tend to be stored as instance data within transactional systems within databases.

Annex H
(informative)
Example use

H.1 General

This annex consists of examples.

Listed data is fictitious and does not exist online. IDs are made human-readable for convenience.

In order to improve readability, the RDF is in Manchester syntax [21].

H.2 Example: A proto-template

The proto-template p7tpl:Beginning serves as a minimal example of RDF representation of templates.

H.2.1 The template defined as a class

The *class* p7tpl:Beginning is defined as a ProtoTemplateStatement, as follows:

```
Class: Beginning
  SubClassOf:
      ProtoTemplateStatement,
      hasPart some dm:Event,
      hasWhole some dm:PossibleIndividual
```

The roles p7tpl:hasPart and p7tpl:hasWhole are themselves subtypes of p7tm:hasObjectRoleFiller.

```
ObjectProperty: hasObjectRoleFiller
  Range:
      dm:Thing

ObjectProperty: hasPart
  Characteristics:
      Functional
  Range:
      dm:PossibleIndividual
  SubPropertyOf:
      hasObjectRoleFiller
```

H.2.2 A statement using the template

This example shows the template instance.

Let Alfred be an individual, and _1970-01-01 be an **Event** that is the beginning of Alfred.

NOTE The beginning event has a further representation by means of an `xsd:dateTime`.

Alfred and his beginning are defined as follows:

```
Individual: company1:Alfred
  Types:
      WholeLifeIndividual
```

```
Individual: company1:_1970-01-01
  Types:
     Event
  Facts:
     ClassOfRepresentationOfThing "1970-01-01T00:00:00"^^xsd:dateTime
```

An instance of the Beginning template that corresponds to the statement

The Beginning of Alfred is the event _1970-01-01

is then expressed as follows in RDF:

```
Individual: company1:Beginning_of_Alfred_statement
   Types:
      p7tpl:Beginning
   Facts:
      p7tpl:hasWhole company1:Alfred,
      p7tpl:hasPart company1:_1970-01-01
```

The Beginning template described above gets the following representation by instances of the meta-templates.

First, its template-asserted that the Beginning has 2 arguments.

```
Individual: p7tpl:TemplateDescription_of_Beginning
  Types:
     p7tm:TemplateDescription
  Facts:
     p7tm:hasTemplate p7tpl:Beginning,
     p7tm:valNumberOfRoles 2
```

Then, two other template instances state which roles are involved, and their index (the position within the sequence of roles in the template).

```
Individual: p7tpl:TemplateRoleDescription_of_Beginning_1
  Types:
     p7tm:TemplateRoleDescription
  Facts:
     p7tm:hasTemplate p7tpl:Beginning,
     p7tm:hasRole p7tpl:hasPart,
     p7tm:hasRoleFillerType Event,
     p7tm:valRoleIndex 1

Individual: p7tm:TemplateRoleDescription_of_Beginning_2
  Types:
     p7tm:TemplateRoleDescription
  Facts:
     p7tm:hasTemplate p7tpl:Beginning,
     p7tm:hasRole p7tpl:hasWhole,
     p7tm:hasRoleFillerType PossibleIndividual,
     p7tm:valRoleIndex 2
```

H.3 Example: OIM

H.3.1 Description

This example OIM is made in native OWL. That means that there is some accuracy lost in comparison to the use of templates. For example, oim:MaximumDesignPressure can only have a value of xsd:float, and does not have a unit of measurement slot.

The OIM gives a guidance how to build up the specialized template model. It shows that the "identification" template is a relation of the PossibleIndividual class, so that through inheritance all subclasses of PossibleIndividual can have use of an identification. The specialized "identification" template will classify the identification to a certain type (like Tag Name, Line Number, Serial Number, Activity Code or RFID).

The OIM model also gives a guidance how to build up the "assembly" and "contained" relationships between the different OIMs.

Note that a PumpSuction contains a ProcessItemFluidStream. The ProcessItemFluidStream is involved in ProcessCases that can be identified like "Startup Case" and "Design Case". A ProcessCase is involved in ProcessConditions that can be identified like "Minimum", "Normal" and "Maximum". And that these ProcessConditions contain process data like pressure, temperature, viscosity etc.

H.3.2 Manchester syntax

```
Class: rdl:PossibleIndividual
   Types:
      rdl:IndividualClass
   SubClassOf:
      rdl:Thing
      oim:hasIdentifier xsd:string

Class: rdl:CentrifugalPump
   SubClassOf:
      rdl:DynamicPump
   oim:Assembly
      some rdl:PumpImpeller
      some rdl:PumpSuction
      some rdl:PumpDischarge
   oim:DirectConnection some rdl:PumpDriver
   oim:MaximumDesignPressure xsd:float

Class: rdl:PumpImpeller
   SubClassOf:
      rdl:Impeller

Class: rdl:Impeller
   SubClassOf:
      rdl:Artefact
      oim:ImpellerDiameter xsd:float

Class: rdl:PumpDischarge
   SubClassOf:
      rdl:Artefact
```

```
        oim:Contains some rdl:ProcessItemFluidStream

Class: rdl:ProcessItemFluidStream
    SubClassOf:
        rdl:FluidStream
        oim:InvolvedIn
        rdl:ProcessItemFluidStreamCase some rdl:ProcessCase

Class: rdl:ProcessCase
    SubClassOf:
        rdl:Processing
        oim:InvolvedIn
        rdl:ProcessItemFluidCondition some rdl:ProcessCondition

Class: rdl:ProcessCondition
    SubClassOf:
        rdl:Processing
        oim:Pressure xsd:float
        oim:MassFlow xsd:float
        oim:Temperature xsd:float
        oim:Viscosity xsd:float
```

H.4 Example: temporal parts

H.4.1 General

This is an example on the representation of temporal parts as defined in ISO 15926-2. First is shown the modelling patterns used in ISO/TS 15926-8 for this construction, then the concrete representation in RDF/OWL of models and templates.

H.4.2 Generic representation

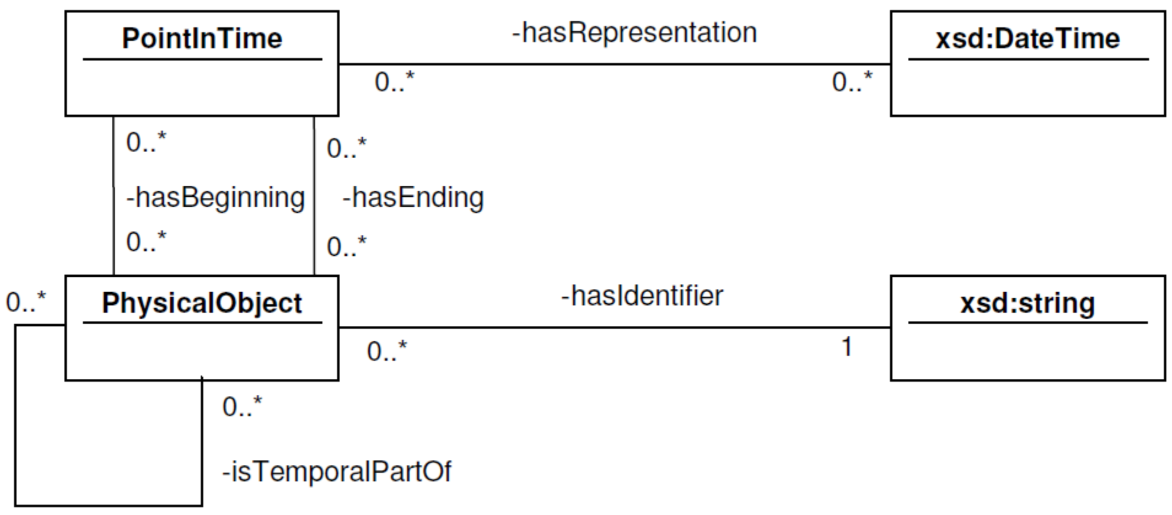

Figure H.1 — Generic temporal parts, identification OIM

Figure H.1 UML diagram illustrates the object information model for generic aspects of temporal parts, the events that mark beginnings and ends of temporal parts, and assignment of names.

— a **PhysicalObject** may be a temporal part of another;

— a **PhysicalObject** may be related to beginning and end of life events, of type **PointInTime**. Such events may further be represented by dates (of type *xsd:dateTime*);

— a **PhysicalObject** may be named by a string. (of type *xsd:string*).

H.4.3 Specialized template IdentificationOfPhysicalObject

The modelling pattern of Figure H.1 is used to describe individuals, expressing information that uses/instantiates the modelling. Templates are suitable for capturing instantiation patterns. For illustration, specialized template IdentificationOfPhysicalObject is introduced, with the following intuitive use:

is a five-place template for asserting that a physical object (x_1) has a temporal part (x_2) that comes into existence at a certain point in time (x_3), and which has a certain type (x_4) of name (x_5).

NOTE 1 The OIM serves to motivate why the template is constructed the way it is. The broader ontology (RDL) serves as a basis from which an OIM is drawn to focus on a particular kind of thing; then a template is introduced to capture a usage pattern for the OIM.

The template signature for rdl:IdentificationOfPhysicalObject looks as follows:

Order	Role name	Role type
1	p7tpl:hasWhole	dm:PhysicalObject
2	p7tpl:hasPart	dm:PhysicalObject
3	p7tpl:valStartTime	xsd:dateTime
4	p7tpl:hasContext	dm:ClassOfClassOfIdentification
5	p7tpl:valIdentifier	xsd:string

NOTE 2 IdentificationOfPhysicalObject is a template of which the axiom contains templates ClassifiedIdentification and BeginningOfIndividual. See ISO/TS 15926-7.

NOTE 3 The hasContext role representing the type of identification, does not have a model element to represent it in Figure H.1. This is mainly because there is no UML symbol to represent subtypes of relations, which means that the diagram fails to capture an aspect of the intended representation The diagram representation is therefore incomplete. To obtain a fully explicit interpretation of the template, a template axiom should be provided according to ISO 19526-7. Stated in terms of the reified format, with the whole set of ISO 15926-2 constructs available, the axiom should give a fully explicit representation.

H.4.4 Specialized template IdentificationOfPhysicalObject-processplant

The following is an example of a specialized OIM, and a template to match. Figure H.2 displays a model that is a restricted version of the one shown in Figure H.1. Let the focus for this OIM be ProcessPlant temporal parts. Here, naming of temporal parts is restricted to ProcessPlantName assignments (distinguishing the naming acts in scope for this OIM from various other kinds of name assignments).

NOTE 1 TheObject in this OIM is represented by the ProcessPlant class in the middle of the diagram. A conventional means is needed, e.g. a colored class box, to point out which class is in focus.

For the restricted OIM, cardinality constraints are added that were not present in the more generic one. According to Figure H.1, the temporal part ProcessPlant is a part of at least one other ProcessPlant. Furthermore, the temporal part is assigned exactly one ProcessPlantName identifier (an *xsd:string*), and exactly one Beginning, the latter which is given exactly one timestamp (an *xsd:dateTime*).

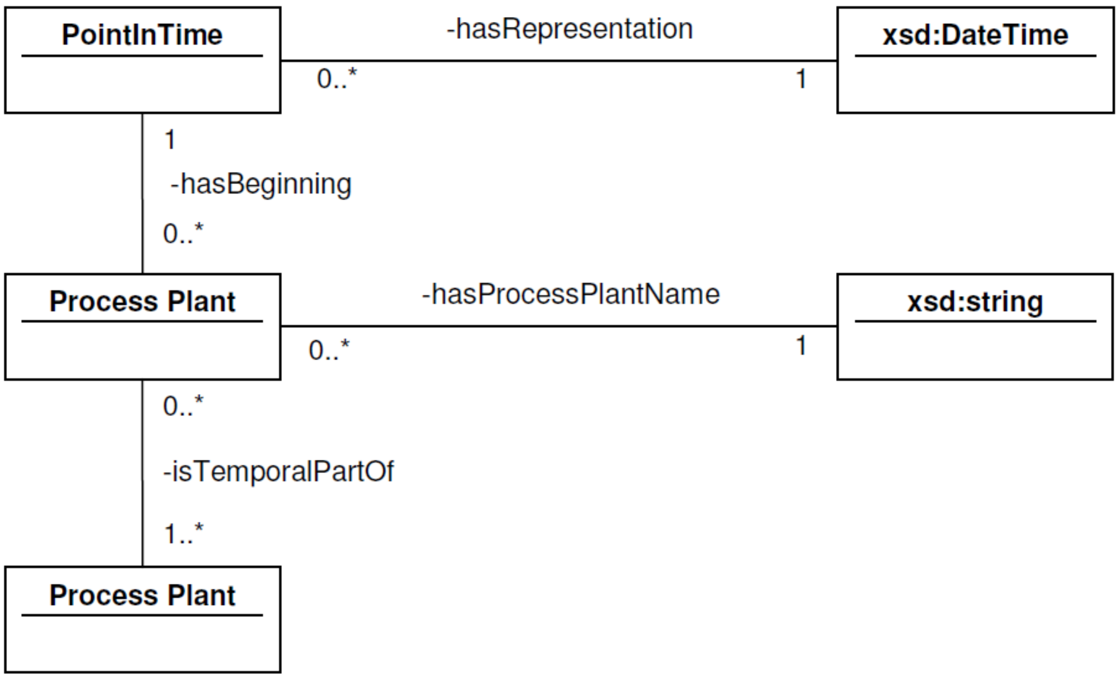

Figure H.2 — ProcessPlant temporal part, name OIM

Just as one OIM is a restricted version of another, the template IdentificationOfPhysicalObject-processplant is introduced as a specialization of template IdentificationOfPhysicalObject. For specialization, the first two roles of IdentificationOfPhysicalObject are restricted to the class ProcessPlant, and the fourth role of IdentificationOfPhysicalObject is fixed to the value ProcessPlantName (rdl:R1234).

NOTE 2 Role 4 is fixed to a constant by asserting that its type is given by the singleton set ProcessPlantName, i.e., the set containing just ProcessPlantName. The requirement for a fixed value is secured because the type of the role has only one member – any valid template instance will have to have ProcessPlantName in role 4.

Order	Role name	Role type
1	p7tpl:hasWhole	rdl:ProcessPlant
2	p7tpl:hasPart	rdl:ProcessPlant
3	p7tpl:valStartTime	xsd:dateTime
4	p7tpl:hasContext	rdl:ProcessPlantName
5	p7tpl:valIdentifier	xsd:string

NOTE 3 References to identifiers in examples are fictitious, and IDs are made human-readable.

H.5 A specialized template and a template instance

Let t100101 be an instance of IdentificationOfPhysicalObject-processplant, as follows:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
    <!ENTITY owl "http://www.w3.org/2006/12/owl2-xml#">
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
    <!ENTITY dm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/data-model#">
    <!ENTITY p7tm "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tm#">
    <!ENTITY p7tpl "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#">
    <!ENTITY meta "http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/metadata#">
    <!ENTITY rdl "http://rdl.example.org#">
    <!ENTITY steplib "http://www.steplib.com/data#">
    <!ENTITY pca "http://www.posccaesar.org/data#">
    <!ENTITY company1 "http://rdl.example.com#">
]>
<rdf:RDF xmlns="http://rdl.example.com#"
     xml:base="http://rdl.example.com"
     xmlns:owl="&owl;"
     xmlns:xsd="&xsd;"
     xmlns:rdf="&rdf;"
     xmlns:rdfs="$rdfs;"
     xmlns:dm="&dm;"
     xmlns:p7tm="&p7tm;"
     xmlns:p7tpl="&p7tpl;"
     xmlns:meta="&meta;"
     xmlns:rdl="$rdl;"
     xmlns:steplib="&steplib;"
     xmlns:pca="&pca;"
     xmlns:company1="&company1;"
>

<rdl:IdentificationOfPhysicalObject-processplant rdf:ID="#t100101">
    <meta:annUniqueName rdf:datatype="&xsd;string">
        pcs125425 identified by process plant B14 since 2009-12-16T14:28:00Z
    </meta:annUniqueName>
    <meta:annRule rdf:datatype="&xsd;string">#com1_26539436548</meta:annRule>
    <meta:annAccessCode rdf:datatype="&xsd;string">#com1_836379</meta:annAccessCode>
    <p7tpl:hasWhole rdf:resource="#pcs125425"/>
    <p7tpl:hasPart rdf:resource="#pcs125789"/>
    <p7tpl:valStartTime rdf:datatype="&xsd;dateTime">
        2009-12-16T14:28:00Z
    </p7tpl:valStartTime>
    <p7tpl:hasContext rdf:resource="&rdl;ProcessPlantName"/>
    <p7tpl:valIdentifier rdf:datatype="&xsd;string">
        B14
    </p7tpl:valIdentifier>
</rdl:IdentificationOfPhysicalObject-processplant>

</rdf:RDF>
```

This example shows the identification string B14 for a specific process plant as template instance.

NOTE 1 The data literals valStartTime and valIdentifier are not modelled as EXPRESS types but as RDF literals. This as per best practice OWL, not following the ISO 15926-2 method for EXPRESS literals.

NOTE 2 The template contains a temporal whole-part construction modelling the identification as temporal part. To model aspects as temporal part is not normative, but rather best practice modelling which is business-case driven. Sometimes it is useful to model aspects as temporal parts, sometimes it is not, and it is also possible to introduce temporal parts when an aspect is revised.

Some of the referred objects examples are:

Base namespace “company1” (example project database):

```
<rdl:ProcessPlant rdf:ID="#pcs125425">
    <meta:annUniqueName rdf:datatype="&xsd;string">
        process plant B14
    </meta:annUniqueName>
    <meta:annAccessCode rdf:datatype="&xsd;string">#com1_836379</meta:annAccessCode>
</rdl:ProcessPlant>

<rdl:ProcessPlant rdf:ID="#pcs125789">
    <meta:annUniqueName rdf:datatype="&xsd;string">
        temporal part - identified by process plant
    </meta:annUniqueName>
    <meta:annAccessCode rdf:datatype="&xsd;string">#com1_836379</meta:annAccessCode>
</rdl:ProcessPlant>
```

Base namespace “rdl” (the core RDL):

```
<dm:ClassOfInanimatePhysicalObject rdf:ID="#ProcessPlant">
    <rdfs:subClassOf rdf:resource="#PhysicalObject"/>
    <meta:annUniqueName rdf:datatype="&xsd;string" xml:lang="en">
        PROCESS PLANT
    </meta:annUniqueName>
    <meta:annUniqueName rdf:datatype="&xsd;string" xml:lang="nl">
        PROCESS FABRIEK
    </meta:annUniqueName>
    <meta:annTextDefinition rdf:datatype="&xsd;string" xml:lang="en">
        A plant employed in carrying out chemical processes,
        including the required supporting processes.
    </meta:annTextDefinition>
    <meta:annTextDefinition rdf:datatype="&xsd;string" xml:lang="nl">
        Een fabriek voor het verrichten van chemische processen,
        inclusief de vereiste ondersteunende processen.
    </meta:annTextDefinition>
</dm:ClassOfInanimatePhysicalObject>
```

The core template OWL used for this instance looks like this:

Base namespace “rdl” (the core RDL):

```
<p7tpl:Template rdf:ID="#IdentificationOfPhysicalObject">
    <meta:annUniqueName xml:lang="en">Identification of Physical Object</meta:annUniqueName>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasWhole"/>
                    <owl:allValuesFrom rdf:resource="&dm;PhysicalObject"/>
                </owl:Restriction>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasWhole"/>
                    <owl:onClass rdf:resource="&dm;PhysicalObject"/>
                    <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">
                    1
                    </owl:qualifiedCardinality>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasPart"/>
                    <owl:allValuesFrom rdf:resource="&dm;PhysicalObject"/>
                </owl:Restriction>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasPart"/>
                    <owl:onClass rdf:resource="&dm;PhysicalObject"/>
                    <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">
                    1
                    </owl:qualifiedCardinality>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="&p7tpl;valStartTime"/>
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasContext"/>
                    <owl:allValuesFrom rdf:resource="&dm;ClassOfClassOfIdentification"/>
                </owl:Restriction>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasContext"/>
                    <owl:onClass rdf:resource="&dm;ClassOfClassOfIdentification"/>
                    <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">
                    1
                    </owl:qualifiedCardinality>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="&p7tpl;valIdentifier"/>
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</p7tpl:Template>
```

NOTE 3 This is a specialized template, typed as "p7tpl:Template" which is an *owl:Class*.

The specialized template OWL used for this instance looks like this:

Base namespace "rdl" (the core RDL):

```
<p7tpl:Template rdf:ID="#IdentificationOfPhysicalObject-processplant">
    <meta:annUniqueName xml:lang="en">Identification of Physical Object</meta:annUniqueName>
    <rdfs:subClassOf rdf:resource="#IdentificationOfPhysicalObject"/>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasWhole"/>
                    <owl:allValuesFrom rdf:resource="&rdl;ProcessPlant"/>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasPart"/>
                    <owl:allValuesFrom rdf:resource="&rdl;ProcessPlant"/>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&p7tpl;hasContext"/>
                    <owl:allValuesFrom rdf:resource="&rdl;ProcessPlantName"/>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
</p7tpl:Template>
```

Bibliography

[1] ISO/IEC 8824-1, *Information technology – Abstract Syntax Notation One (ASN.1) – Part 1: Specification of basic notation.*

[2] ISO 10303-1, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

[3] ISO 10303-11:2004, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual.*

[4] ISO 10303-21, *Industrial automation systems and integration — Product data representation and exchange — Part 21: Implementation methods: Clear text encoding of the exchange structure.*

[5] ISO 15926-1:2004, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 1: Overview and fundamental principles.*

[6] ISO 15926-2:2003, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 2: Data model.*

[7] ISO/TS 15926-3, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 3: Reference data for geometry and topology.*

[8] ISO/TS 15926-4, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 4: Initial reference data.*

[9] ISO/TS 15926-6[2)], *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 6: Scope and representation for additional reference data.*

[10] ISO/TS 15926-7:2011, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 7: Implementation methods for the integration of distributed systems: Template methodology.*

[11] *OWL 2 web ontology language – structural specification and functional-style syntax.* [online]. W3C recommendation 2009-10-27. Available from World Wide Web: <`http://www.w3.org/TR/owl2-syntax`>.

[12] *OWL 2 web ontology language – quick reference guide.* [online]. W3C recommendation 2009-10-27. Available from World Wide Web: <`http://www.w3.org/TR/owl2-quick-reference`>.

[13] *RDF vocabulary description language 1.0: RDF schema.* [online]. W3C recommendation 2004-02-10. Available from World Wide Web: <`http://www.w3.org/TR/2004/REC-rdf-schema-20040210`>.

[14] *RDF/XML syntax specification (revised).* [online]. W3C recommendation 2004-02-10. Available from World Wide Web: <`http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210`>.

2) Under preparation

[15] *Resource description framework (RDF): concepts and abstract syntax.* [online]. W3C recommendation 2004-02-10. Available from World Wide Web: <`http://www.w3.org/TR/2004/REC-rdf-concepts-20040210`>.

[16] *RFC 2396 uniform resource identifiers (URI): generic syntax.* [online]. IETF, 1998-08. Available from World Wide Web: <`http://www.ietf.org/rfc/rfc2396.txt`>.

[17] *SPARQL protocol for RDF (SPROT).* [online]. W3C recommendation 2008-01-15. Available from World Wide Web: <`http://www.w3.org/TR/rdf-sparql-protocol`>.

[18] *XML schema part 2: datatypes second edition.* [online]. W3C recommendation 2004-10-28. Available from World Wide Web: <`http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/`>.

[19] BERNERS-LEE, Tim. *Linked Data.* [online]. Status: personal view only. Editing status: imperfect but published. Available from World Wide Web: <`http://www.w3.org/DesignIssues/LinkedData.html`>.

[20] BRAY, Tim, PAOLI Jean, SPERBERG-MCQUEEN, C.M., MALER, Eve, YERGEAU, Fran-ois, COWAN, John. *Extensible markup language (XML) 1.1 (second edition).* [online]. W3C recommendation 2006-08-16, edited in place 2006-09-26. Available from World Wide Web: <`http://www.w3.org/TR/xml11`>.

[21] HORRIDGE, Matthew and PATER-SCHNEIDER, Peter F. *OWL 2 web ontology language manchester syntax.* [online]. W3C working group note 2009-10-27. Available from World Wide Web: <`http://www.w3.org/TR/owl2-manchester-syntax`>.

[22] NOY, Natasha, RECTOR, Alan, HAYES, Pat, WELTY, Chris. *Defining n-ary relations on the semantic web (refer pattern 1).* [online]. W3C working group note 2006-04-12. Available from World Wide Web: <`http://www.w3.org/TR/swbp-n-aryRelations#pattern1`>.

[23] SCHLIMMER, Jeffrey C. *Web services description requirements.* [online]. W3C working draft 2002-10-28. Available from World Wide Web: <`http://www.w3.org/TR/ws-desc-reqs`>.

[24] *Graph stylesheets (GSS) in IsaViz.* [online]. Available from World Wide Web: <`http://www.w3.org/2001/11/IsaViz/gss/gssmanual.html`>.

[25] ISO/IEC 11179-1:2004, *Information technology ð Metadata registries (MDR) ð Part 1: Framework*

Index

accessCode 15

base template 1
BaseTemplateStatement 24

core class 2
Core RDL 7
core template 2, 14

data store 2
data warehouse 2

entity data type 3
example OIM 39
example temporal Parts 40
EXPRESS entity types 8

first order logic 8
First Order Logic axiom 15

hasDataRoleFiller 25
hasObjectRoleFiller 25

interoperability 4

lifted templates 15
lowered templates 16

Manchester syntax 4, 37
metadata 4, 15
MetaTemplateStatement 25
Multidimensional Object 15
Multidimensional Object class 14

NCName 30

OIM 4
ontology 4
OWL native . . . . viii, 4, 27

punning 5

rdf:about 30
rdf:ID 30
RDL 5
reification 5
rule 15

signature 5
SPARQL endpoint 6
specialized template 6

taxonomy 6
template 6
template statement 7
TemplateDescription 25

TemplateRoleDescription . . . 25
TemplateSpecialization . . . 26
TemplateStatement . . . 24
triple . . . 7
triple store . . . 7
type of rule language . . . 1

ICS 25.040.40; 75.020

Price based on 50 pages