TECHNICAL SPECIFICATION

ISO/TS 15926-11

First edition
2015-05-01

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

Part 11: Methodology for simplified industrial usage of reference data

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 11: Méthodologie pour un usage industriel simplifié des données de référence

Reference number
ISO/TS 15926-11:2015(E)

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

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

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is 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.

ISO 15926 consists of the following parts, under the general title *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;*

— *Part 2: Data model;*

— *Part 3: Reference data for geometry and topology* [Technical Specification]*;*

— *Part 4: Initial reference data* [Technical Specification]*;*

— *Part 6: Methodology for the development and validation of reference data* [Technical Specification]*;*

— *Part 7: Implementation methods for the integration of distributed systems: Template methodology* [Technical Specification]*;*

— *Part 8: Implementation methods for the integration of distributed systems: Web Ontology Language (OWL) implementation* [Technical Specification];

— *Part 11: Methodology for simplified industrial usage of reference data* [Technical Specification]

The following parts are under preparation:

— *Part 9: Implementation methods for the integration of distributed systems: Facade implementation* [Technical Specification]*;*

— *Part 10: Conformance testing* [Technical Specification]

Introduction

ISO 15926 is an International Standard for the representation of process industries facility 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 model is designed to be used in conjunction with reference data, that is, standard instances that represent information common to a number of users, production facilities, or both. The support for a specific life-cycle activity depends on the use of appropriate reference data in conjunction with the model.

This part of ISO 15926 focuses on a simplified implementation of the afore mentioned data model in the context of engineering data in the area of the process industry, including the oil, gas, process and power industry and is intended for developers of configuration management processes and systems in general.

This part of ISO 15926 provides the capability to express a product model with RDF triples, RDF Named Graphs and a standardized set of natural language relationships resulting in a table that can be exchanged and shared easily in industry.

There is an industry need for this part of ISO 15926.

— The triple relationships are easy to understand by an engineer so that an engineer can understand the product model. This has been proven by the NL Ship Building group who developed a Gellish-RDF based implementation for standardized exchange of product data of HVAC equipment on a daily basis.

— The standard data sheets from API, NORSOK, etc. used in industry for pumps, compressors, instruments, etc. can be supported by a Gellish-RDF product model enabling industry to continue to work with their specific data sheets and yet exchanging the data in standardized way according this new standard. This has been proven by the ICAAMC compressor group in a pilot for the API 617 data sheet.

— It is used in some projects, e.g. in the Pearl project for oil and gas.

— This part of ISO 15926 can be used as a front end engineering layer for the template methodology used by ISO/TS 15926-7 and ISO/TS 15926-8, e.g. in the FIATECH project IIP. This will make the content of those projects easier to access by engineers.

— An EPC contractor has used the draft of this part of ISO 15926 in various tunnel projects for information modelling in the area of systems engineering which was required by the Dutch authority regulations. With this part of ISO 15926 enriched by the knowledge from ISO/IEC 15288, this became possible. They also built a performance measuring system for operational data in tunnel installations where the methodology of this part of ISO 15926 is used to justify the performance to the ministry of transportation in the Netherlands.

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

Part 11: Methodology for simplified industrial usage of reference data

1 Scope

This part of ISO 15926 enables a flexible creation of product knowledge models that can be exchanged in the plant engineering supply chain by combining RDF triples within named graphs, reference data dictionaries and a standardized set of relationships.

This part of ISO 15926 is appropriate for use with ISO 15926 reference data libraries, and it is applicable to the oil, gas, process and power industries.

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

— process plants in accordance with ISO 15926-1;

— use of RDF triples representing statements as defined in the ISO 15926 series;

— an initial set of relationships required for process plant life cycle representation;

— rules for the use of RDF Named Graphs for product data representation and exchange;

— examples of possible implementations.

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

— definition of reference data libraries;

— the syntax and format of implementations of product data models and/or instance data using this part of ISO 15926;

— any specific methods and guidelines other than RDF Named Graphs for implementing ISO 15926-2.

2 Normative references

The following referenced documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

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*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

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

3.1.1
class
category or division of things based on one or more criteria for inclusion and exclusion

Note 1 to entry: A class need not have any members (things that satisfy its criteria for membership).

[SOURCE: ISO 15926-1:2004, 3.1.1, modified]

3.1.2
characteristic data
description of an entity by the class to which it belongs and a set of property values

Note 1 to entry: ISO 13584, ISO 15926, ISO 22745, ISO 13399, and ISO/TS 29002 all include characteristic data in their data models.

[SOURCE: ISO 8000-2:2012, 7.2, modified]

3.1.3
data
representation of information in a formal manner suitable for communication, interpretation, or processing by human beings or computers

[SOURCE: ISO 10303-1:1994, 3.2.14]

3.1.4
formal syntax
specification of the valid sentences of a formal language using a formal grammar

Note 1 to entry: A formal language is computer-interpretable.

EXAMPLE 1 An XML document type definition (DTD) is a formal syntax.

EXAMPLE 2 ISO 10303-21, contains a formal syntax in WSN for ISO 10303 physical files.

[SOURCE: ISO 8000-2:2012, 6.1, modified]

3.1.5
information
facts, concepts, or instructions

[SOURCE: ISO 10303-1:1994, 3.2.20]

3.1.6
Named Graph
key concept of Semantic Web architecture in which a set of Resource Description Framework (RDF triples) statements (a graph) are identified using a unique URI

Note 1 to entry: Named Graphs is the idea that having multiple RDF graphs in a single document/repository and naming them with URIs provides useful additional functionality built on top of the RDF Recommendations

[SOURCE: W3C Recommendation 25 February 2014, modified]

3.1.7
N-Quad statement
N-Quad
sequence of RDF terms representing the subject, predicate, object and graph identifier of an RDF Triple and the graph it is part of in a dataset

Note 1 to entry: These may be separated by white space. This sequence is terminated by a '.' and a new line (optional at the end of a document).

EXAMPLE < http://one.example/subject1> < http://one.example/predicate1> < http://one.example/object1> < http://one.example/graph3> . # comments here.

[SOURCE: W3C Recommendation 25 February 2014, modified]

3.1.8
RDF graph
graph structure formed by a set of RDF triples

[SOURCE: W3C Recommendation 25 February 2014]

3.1.9
reference data
facility life-cycle data that represents information about classes or individual things which are common to many facilities or of interest to many users

[SOURCE: ISO 15926-1:2004, 3.1.18, modified]

3.1.10
reference data library
RDL
managed collection of reference data

[SOURCE: ISO 15926-1:2004, 3.1.19]

3.1.11
relationship
abstract object that indicates something that one thing has to do with another

[SOURCE: ISO 15926-2:2003, 4.6.4, modified]

3.1.12
semantic encoding
technique of replacing natural language terms in a message with identifiers that reference data dictionary entries

[SOURCE: ISO 8000-2:2012, 6.2]

3.1.13
statement
fact
information that is regarded as indivisible

Note 1 to entry: A statement can be recorded as an instance of the entity relationship in ISO 15926-2. A set of one or more statements can be recorded in shorthand form as a single item as an instance of a template, as defined in ISO/TS 15926-7.

[SOURCE: ISO/TS 15926-6:2013, 3.1.25]

3.1.14
thing
actual part of the real world, perceived part of the real world, or subject of thought

Note 1 to entry: A thing can be a material or non-material object, idea or action.

Note 2 to entry: This definition is adapted from ISO 15926-2, within which “thing” is an entity, but not a defined term.

[SOURCE: ISO/TS 15926-6:2013, 3.1.26]

3.1.15
triple
RDF-triple, representation of a relation between the objects or data that it links

Note 1 to entry: 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”.

[SOURCE: W3C Recommendation 25 February 2014]

3.1.16
product
thing or substance produced by a natural or artificial process

[SOURCE: ISO 10303-1:1994, 3.2.26]

3.1.17
product data
representation of information about a product in a formal manner suitable for communication, interpretation, or processing by human beings or by computers

[SOURCE: ISO 10303-1:1994, 3.2.27]

3.2 Abbreviated terms

IDM	Information Delivery Manual
RDL	Reference Data Library
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SPARQL	Protocol and RDF Query Language
TriX	Triples in XML
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium

4 Fundamental concepts and assumptions

4.1 Purpose and objectives

This part of ISO 15926 provides a semantic modelling methodology of engineering data that can be relatively “easily” understood by engineers and that is flexible in terms of tailoring the methodology for a specific domain or project. The provided modelling methodology is based on the existing parts ISO 15926-2 and ISO/TS 15926-4. To achieve this, the triple concept of the RDF standard of W3C supplemented with the Named Graph concept of W3C is adopted and augmented with a set of relationships further called the “initial set of relationships”.

Reason for developing this methodology can be found in the fact that product specialists and engineers, especially within Small and Medium Enterprises (SMEs), who in general have limited skills in the area of information modelling and related techniques, and they should be supported in their product and or engineering knowledge modelling activities by a simple to use methodology, close to natural languages. Using this methodology will lead to models that are upgradable to full ISO 15926-2 compliant models. In other words this methodology provides a bridge to the much more complex ISO 15926-2 world and provide a low entry threshold to ISO 15926. Also the fact in development work is that humans like to work with simple table based structures rather than relatively complicated schemes should be respected if possible.

In line with this, this part of ISO 15926 offers a way to industry groups to set up and exchange their product model using a low level modelling methodology based on statements in a table manner. This part of ISO 15926 provides for that purpose a normative set of rules that allows engineers to build a product and plant lifecycle models using statements based on a normative set of relationships and normative reference data. A statement is: that which is the case, independent of natural language. A statement can be used to classify things as 'being the case'. Statements can be expressed in languages as relationships between two roles of things (respectively "thing playing role 1" and "thing playing role 2").

With the help of this part of ISO 15926, one is able to express any kind of product or engineering information and exchanging this information based on a managed set of reference data, including information concerning all relevant (system life cycle) processes to realize and maintain the product. This part of ISO 15926 provides a semantic modelling methodology for creating and exchanging engineering data, originating from Systems Engineering processes.

4.2 Positioning of this part of ISO 15926

Figure 1 shows the layers that can be distinguished looking at data exchange in general. Within this figure the upper layer represents the role that a specific organization or enterprise plays in the exchange of information. The origin and scope of the information is represented by the second layer. For this part of ISO 15926, the origin can be seen as applicable quality aspects of information as defined in ISO 8000-1, while the scope of the information exchange is ISO/IEC 15288, specific in the context of process industry.

The content layer represents the meaning of the objects that are exchanged as defined by the RDL ISO/TS 15926-4.

The semantic layer is the scope of this part of ISO 15926 and is represented by the RDF Named Graph methodology as described within this part of ISO 15926.

The syntax layer represents the technology that is used to physically exchange the data that represents the information to be exchanged. This part of ISO 15926 prescribes not a standard for any exchange syntax, only gives examples of possible syntax formats as given in Annex B.

The storage layer describes the technology that is used to store the information and can be done by means of a triple store, a graph database or a traditional relational database management system.

Within a specific project each layer can be specified by means of a project-specific "Information Delivery Manual".

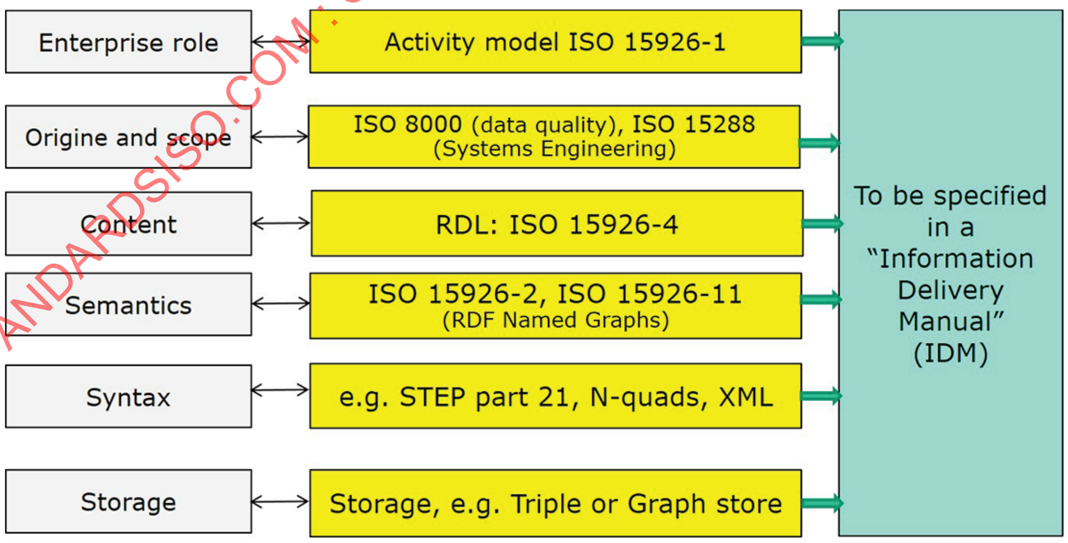
Enterprise role
Activity model ISO 15926-1
Origine and scope
ISO 8000 (data quality), ISO 15288 (Systems Engineering)
Content
RDL: ISO 15926-4
Semantics
ISO 15926-2, ISO 15926-11 (RDF Named Graphs)
Syntax
e.g. STEP part 21, N-quads, XML
Storage
Storage, e.g. Triple or Graph store
To be specified in a "Information Delivery Manual" (IDM)

Figure 1 — Positioning of this part of ISO 15926 in a view on data exchange

4.3 Use of statements within ISO 15926

The aim of this part is to realize a human readable expression of engineering data that can be supported and managed by computers where the aim of ISO 15926-2 is a full understanding of engineering data by computers. Starting point of this part is that engineering data expressed according to this part can be (manually) transformed to the level of ISO 15926-2. For this reason some principles of ISO 15926-2 are respected and other are simplified by means of “short cut” relationships but anyway consistent with part ISO 15926-2

The expression of a statement consists of “object (playing a role 1) –> relationship –> object (playing a role 2)” following the simplest possible grammatical pattern: subject – > verb – > object. This is also called a triple in the context of Resource Description Framework (RDF) developed within the W3C community (see also Figure 4).

RDF is a language for expressing data models using statements in triple formats and sharing them with other people and machines. Since it is a W3C “Recommendation”, large collections of tools and services are available. The principle of using statements describing a “specific world” (an ontology) can replace fixed data models and provides an extensible ontology with reference data with the aim to defining, customizing and harmonizing systems. For this aim, the methodology described in this part of ISO 15926 uses for this goal taxonomy of relationships and a taxonomy of “things” to be able to describe the world in a consistent and explicit way. This is provided by the rule that relevant things in the real world must be classified by a class in a reference data library.

In Figure 2, the principle of the object - relation - object mechanism is shown. The left side object (playing a role 1) “airco xyz” is classified as the right side object (playing a role 2) being an “air-conditioning unit”. In the same way the “capacity airco xyz” is classified as a “nominal cooling capacity”. The given information is for human explicit and readable and correspondents with the information that usually can be found in a datasheet. As example the “capacity airco xyz” is quantified in kW and has a magnitude of 100. Engineers are integrating the facts expressed in sentences with all that they already know such that engineers are able to utilize the meaning that they derived from the statements given to create new knowledge which can be easily communicated with colleagues resulting in actions.

While statements (data model expressed in triple formats) that RDF uses is very simple and easy to understand, but the serialized representation tends to get complicated for engineers to visualize the whole structure of knowledge and communicate it to colleagues. An RDF graphical representation of statements is here to help engineers since RDF also provides a standard way of expressing graphs of statements and sharing them with colleagues and computers on line and/or off line. Further ontologies provide engineers to express the rules for inference from statements. An engineer shares his ontology with his colleague it should have all knowledge he should have to draw same conclusions from statements on hands. And same thing will be true with machines if they commit to share the same ontologies.

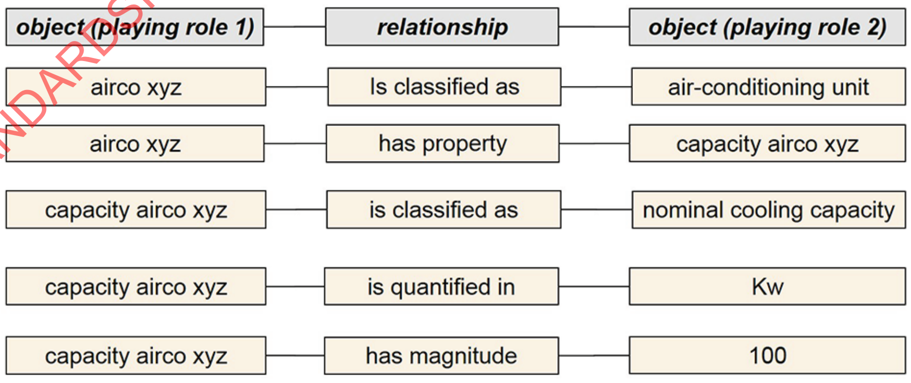

Figure 2 — Representation of a piece of product information in statements

Both objects in a statement have each a specific role, related to the meaning of the relationship. As a result of this a statement always must be readable in two directions, from left to right and from right to left. So each relationship as defined in the initial set has two names: the “prescribed” name, reading from an object playing a role 1 and an object playing a role 2 and the reverse name, reading from an object playing a role 2 to an object playing a role 1 (airco xyz has role 1 and capacity airco xyz has role 2 with respect to the relation). The reverse statements of Figure 2 are shown in Figure 3.

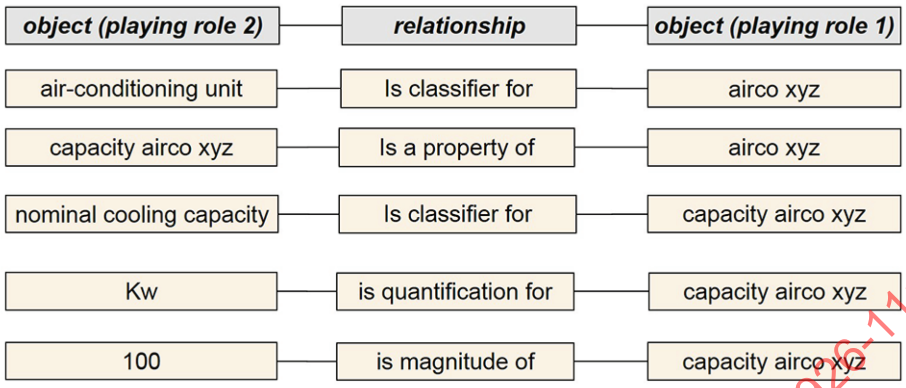

Figure 3 — Statements of Figure 2 in reverse direction

One of the advantages of using statements as shown in this clause is that one is able with one and the same methodology to describe product model data and create instance data in a repository as well but also exchange both kinds of data between parties and or configuration management systems.

4.4 Requirements of statements

In general there is, in the context of engineering data, a need to be able to make statements about statements. This ability is within in standard realized by identifying each statement by a unique identifier. Therefore each statement is provided with a Uniform Resource Identifier (URI). By referring to a specific URI one can make a statement about the statement that is represented by that URI.

EXAMPLE Who has stated a specific statement and when has he made that statement in time are examples of statement about statements.

When in industry parties communicate (exchanging statements) with each other in general there is a need for explicit statements about exchanged statements. These statements about statements support the workflow of processing industrial data along the supply chain. Together with providing provenance, these are fundamental requirements for exchanging as well as referencing and archiving industrial data electronically.

In this part the following statements about statements are recognized to facilitate engineering transactional usage:

— creator of a statement (party or role);

— date and time of creation of the statement;

— modifier of a statement (party or role);

— date and time of modification of a statement;

— certainty of a statement; possible values (instances) for certainty can be “estimated”, “calculated” or “as-build”), supporting of ranking the probability of the correct value of property values;

— modality of a statement; possible modal verbs to express the modus of a statement: possible values (instances) of modality are “Can be”, “Shall be” and “Shall have” supporting requirement management and modelling product knowledge. Default the modus of a statement will be “is the case”;

— intention of a statement, possible values (instances) of intention are: “requested”, “proposed”, “approved”. Supporting the workflow in the exchange process and change management, especially for property values;

— cardinality of an element (specific the “object playing role 2”) within a statement, supporting requirement management and product knowledge modelling;

— versioning of a statement, supporting library versioning and base lining in the context of configuration management;

— relating a statement to a specific system life cycle, making the difference between information that is relevant for the conceptual design, detailed design or the maintenance stage of a system;

— defining the “begin of life” and “end of life” of a statement in order to be able to pinpoint the period in time that the specific statement is valid.

In 5.4 and 5.5 the implementation of afore mentioned requirements is been made explicit by means of examples.

This method can define schemas as well as instances of the scheme defined in neutral format and can be implemented using any syntax layer, e.g. a spread sheet, XML, as shown in Annex B.

4.5 Representing statements in RDF triples

The Resource Description Framework (RDF) is a formal language from the W3C community that makes it possible to describe the semantics of information in similar way as the statement mechanism described in 4.3. RDF is a general-purpose language; originally meant for representing information on the Web. It defines a language for describing relationships among resources in terms of named properties and values. Properties in the context of RDF are instances of the class rdf:Property and describe a relationship between subject resources (left side of the relationship within a statement, the role 1 element) and object resources (right side of the relationship, the role 2 element within a statement). When used as such a property is a predicate in the context of RDF.

RDF provides no mechanisms for describing properties, nor does it provide any mechanisms for describing the relationships between properties and other resources. That is the role of the RDF vocabulary description language, RDF Schema (RDFS). RDFS defines classes and properties that may be used to describe classes, properties and other resources. These resources are used to determine characteristics of other resources, such as the domains and ranges of properties. RDFS vocabulary descriptions are written in RDF.

Most of the abstract model of RDF comes down to four simple rules:

— a statement is expressed as a Subject-Predicate-Object triple: this is similar to a (short) English sentence;

— subjects, predicates, and objects are given as names for entities, also called resources (dating back to the application of RDF to metadata for web resources) or nodes (from graph terminology): entities represent something, a person, website, or something more abstract like states and relations;

— names are URIs, which are global in scope, always referring to the same entity in any RDF document in which they appear;

— objects can also be given as text values, called literal values, which may or may not be typed using XML Schema data types.

The simplicity and flexibility of the triple in combination with the use of URIs for globally unique names for subjects, predicates and objects makes RDF unique, and very powerful. It is a specification that fills a very particular niche for decentralized, distributed knowledge and provides a framework to enable computer applications to answer complex questions.

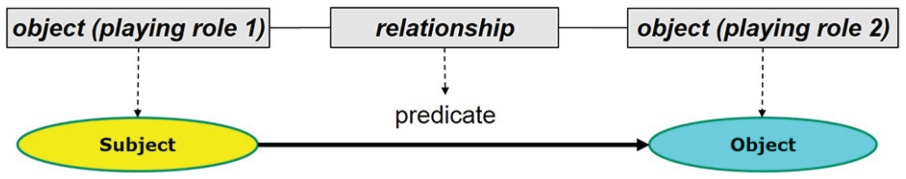

Figure 4 — Adopt RDF triple principle to represent a statement

The design of RDF can be characterized as:

— is based on a simple data model;

— enables a certain level of formal semantics and provable, traceable inference;

— uses an extensible URI-based vocabulary;

— uses an XML-based syntax, supporting XML schema data-types;

— allows anyone to make statements about any resource.

The underlying structure of any expression in RDF is a collection of triples, each consisting of a subject, a predicate and an object. A set of such triples is called an RDF graph. This can be seen in the diagram shown in Figure 4, a triple being described by as a node-arc-node link. An RDF triple is conventionally written in the order subject, predicate, object. The direction of the arc is significant: it always points toward the object by means of an arrow. This way of visualization of statements allows users to gain better grasps of models and instances of models they defined in table manner.

The assertion of an RDF triple says that some relationship, indicated by the predicate, holds between the things denoted by subject and object of the triple, in other words, visualization as an arc is useful to clearly identify the predicate (equal to a relationship) between two things. The assertion of an RDF graph amounts to asserting all the triples in it, so the meaning of an RDF graph is the conjunction (logical AND) of the statements corresponding to all the triples it contains:

— RDF provides a data model for objects and relations between them and provides a simple semantics for the data model;

— RDF Schema provides a vocabulary for describing properties and classes of RDF objects, with semantics for generalization-hierarchies of such properties and classes.

RDF is using the principle of Linked Data. Linked Data describes a method of publishing structured data so that it can be interlinked and become more useful. It enables integrating knowledge from other vocabularies into a product model making a formal reference to elements of those vocabularies in a way that can be read automatically by computer by means of SPARQL, the query language for RDF. Linked Data uses URIs to denote things. Specific HTTP URIs are used so that these things can be referred to and looked up ("dereferenced") by people and user agents.

In this part of ISO 15926, URIs are as follows:

— a URI (Uniform Resource Identifier) is a compact string of characters that is used to identify a subject, predicate, object or Named Graph;

— a hash (#) URI is used whenever one wants to refer to something that doesn't live on the web, with the base URI providing information about that thing;

— a slash (/) URI is used whenever one wants to refer to something that is live and addressable on the web;

— in general, a namespace is a container for a set of identifiers (names). Namespaces usually group names based on their functionality. A namespace can be part of a URI;

— an absolute URI reference consists of three parts: a scheme, a scheme-specific part and a fragment identifier.

EXAMPLE 1 http://standards.iso.org/iso/15926/-4/tech/reference-data#RDL7459 as an example of an hash URI of a class from ISO/TS 15926-4 is broken down as follows, where the first three parts are the name space of the URI:

http	:scheme
http://standards.iso.org/iso	:authority
/15926/-4/tech/reference-data	:path
RDL7459	:fragment

For the fragment part of a Named Graph, preferably a universally unique identifier, is used as defined in ISO/IEC 11578:1996.

EXAMPLE 2 Example of a universally unique identifier (UUID) according to ISO/IEC 11578:1996 is: http://example.com/myproject#40bb99c0-1f74-11e2-81c1-0800200c9a66

NOTE 1 For the naming of objects, this part of ISO 15926 uses the practice of CamelCase for writing compound words. When writing relationships each compound word starts with a lower case letter. Any object not being a relationship starts with a capital, a relationships start with a lower case.

In this part of ISO 15926, the used name spaces are:

— "www.example.com/myproject#" to identify project specific things where the namespace www.example.com/myproject#" is represented by the string "myproject:";

— "www.example.com/part2#" to identify concepts derived from ISO 15926-2 where the namespace "www.example.com/part2#" is represented by the string "part2:";

— "www.example.com/part4#" to identify concepts derived from ISO/TS 15926-4 where the namespace "www.example.com/part4#" is represented by the string "part4:";

— "www.example.com/part11#" to identify relationships and types of Named Graph, where the namespace "www.example.com/part11#" is represented by the string "part11:";

— "www.example.com/library#" for things that comes from a private library where the namespace "www.example.com/library#" is represented by the string "library:".

EXAMPLE 3 Figure 5 shows an example of a set of graphs that represents a set of statements about the specification of a property "nominal load weight" of a typical pedestal crane, including the classification of a pedestal crane. The used namespace is "library:". The statements represented by Figure 5 are:

— "pedestal crane" is a subclass of "crane";

— "crane" has a property "nominal load weight";

— "nominal load weight" is a subclass of property "weight";

— "nominal load weight" has an upper boundary of "440";

— "nominal load weight" is quantified using the UOM "tonnes".

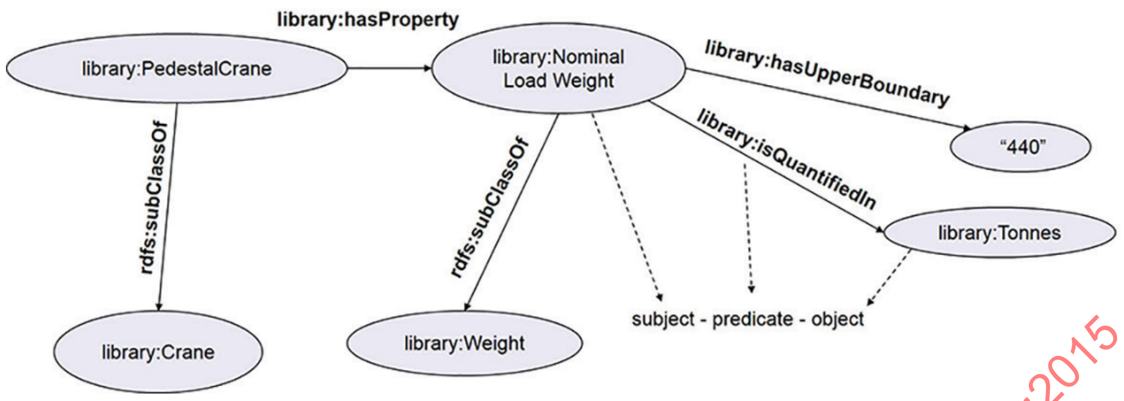

Figure 5 — RDF Example of a specification of a property of a crane

EXAMPLE 4 Figure 6 shows an example of a set of graphs that represents a set of statements about the specification of a property "nominal load weight" of a specific pedestal crane ("deck crane B710" which is an individual) as an instantiation of a typical pedestal crane. The instances are defined in the namespace "myproject:", the classes are defined in the namespace "library:". The statements represented by Figure 6 are:

— "deck crane B710" is an instance of a "pedestal crane";

— "deck crane B710" has a property "nominal load weight deck crane B710";

— "nominal load weight deck crane B710"is an instance of "normal load weight";

— "nominal load weight deck crane B710" is quantified in "tonnes";

— "nominal load weight deck crane B710" has a magnitude of "400".

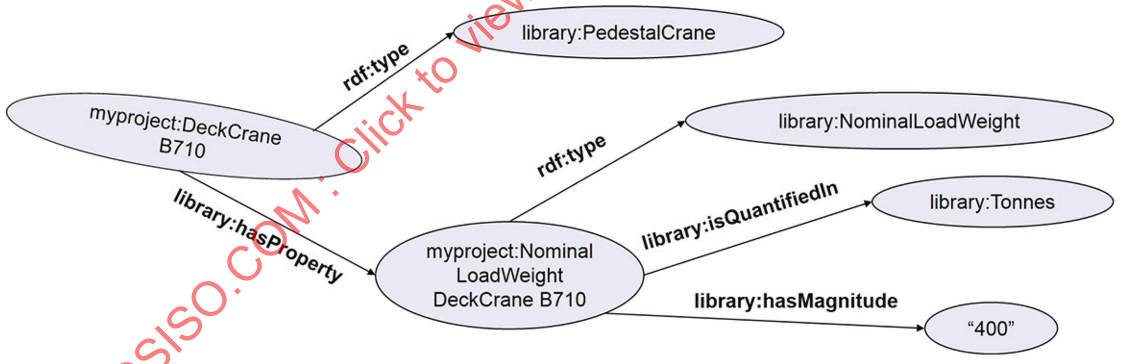

Figure 6 — RDF Example of an instantiation of a "nominal weight" property as defined in Figure 5

NOTE 2 Application software can check if the magnitude of "400" as used in Figure 6 fits in the given upper boundary of "440" defined in the typical of a "Pedestal Crane" as defined in Figure 5.

4.6 RDF Named Graph

Named Graphs are a key concept of Semantic Web architecture in which a set of Resource Description Framework (RDF triples) statements (a graph) are identified using a unique URI. So a statement can be optionally explicit accompanied with additional information. This is necessary to be able to pinpoint the characteristic, context or metadata of a statement.

EXAMPLE 1 An example of information explicit accompanied with a statement is the intention of that statement which states that the statement is required, proposed or approved.

Named Graphs are an easy enhancement of the abstract RDF syntax. Named Graphs enables us to talk about RDF graphs using RDF statements (in other words: allows making statement about statements).

The meta-data of a statement, e.g. source, modifier, modified, context, can be expressed in triples, each belonging to that statement. It is needed to be able to say that these extra triple belongs uniquely to the triple that represents the statement. Therefore the W3C version of RDF named graphs is used in this part. By putting the triple of a "main" statement in a RDF named graph, as many additional triples as needed can point to the first triple, representing metadata.

EXAMPLE 2 A specific statement can define the cardinality of the object that is contained in another specific statement.

EXAMPLE 3 A specific statement can define the certainty of a property value that is defined as the magnitude of a property in another specific statement.

The concept of RDF Named Graphs offers flexibility in modelling products and plants. Also this concept provides the possibility to represent any work or product process in a graphical manner so that it becomes easier to extract required predicates (equal to a relationship). This method has been applied to ISO/IEC 15288 process to define statements in subject, predicate and object so that it becomes possible to bring in systems engineering aspect into the ISO 15926 world.

EXAMPLE 4 Figure 7 shows an example with three Named Graphs that are representing three statements given in Figure 6. The first Named Graph states that DeckCraneB710 is an instance of a "PedestalCrane". The second Named Graph states that DeckCraneB710 has a property "NominalLoadWeightDeckCraneB710" and the third Named Graph states that the magnitude of this property is equal to "400". Each triple is accompanied with metadata such as "isCreated by", "isCreated at" and "isModifiedAt".

NOTE For reason of readability, Named Graphs in the examples used in this part of ISO 15926 have logical names and not a formal UIDD.

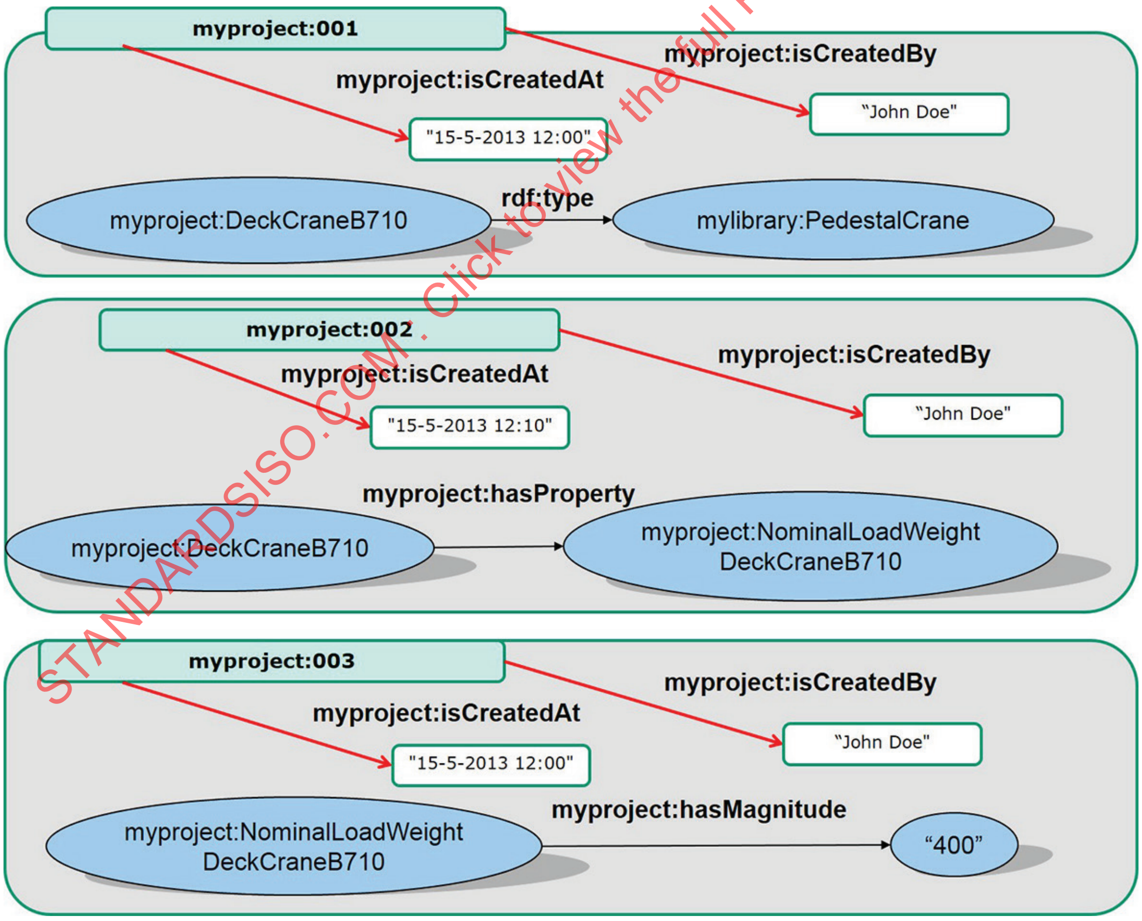

Figure 7 — Named Graph principle applied on three triples taken from Figure 6

In Table 1, the Named Graphs shown in Figure 7 are captured in the N-Quad format of W3C.

Table 1 — W3C N-Quad representation of Named Graphs

Triple			
Subject	**Predicate**	**Object**	**UID Named Graph**
myproject:DeckCraneB710	rdf:type	library:PedestalCrane	myproject:001
myproject:001	library: isCreatedBy	“John Doe”	myproject:001
myproject:001	library: isCreatedAt	“15-5-2013 12:00”	myproject:001
myproject:DeckCraneB710	library: hasProperty	myproject: NominalLoadWeightDeckCraneB710	myproject:002
myproject:002	library: isCreatedBy	“John Doe”	myproject:002
myproject:002	library: isCreatedAt	“15-5-2013 12:11”	myproject:002
myproject: NominalLoadWeightDeckCraneB710	library: hasMagnitude	“400”	myproject.003
myproject:003	library: isCreatedBy	“John Doe”	myproject.003
myproject:003	library: isCreatedAt	“15-5-2013 12:15”	myproject.003

4.7 Connecting RDF with a Reference Data Library (RDL)

An essential part of the methodology is identifying reference data items in the context on the application area of ISO 15926 as well as that of Systems Engineering then to extend a managed set of existing RDLs. In order to build such a set of reference data items, from the RDF namespace this part of ISO 15926 uses:

— rdf:type;

— rdf:Property;

— rdfs:Class;

— rdfs:subClassOf;

— rdfs:subPropertyOf.

The relationship “is a specialization of” is within this part of ISO 15926 identical to “rdfs:subClassOf”, the relationship “is an instance of” is within this part of ISO 15926 identical to “rdf:type”.

To be able to use ISO/TS 15926-4 in combination with this part of ISO 15926 in the context of a RDF environment, there is a class “thing” defined as a rdf:type of rdfs:Class. In a simplified way, ISO 15926-2 “thing” can be seen as the root classes for all object classes of this part of ISO 15926 (see Figure 8). The root classes of this part of ISO 15926 are specialized in Figure 8 to sub classes in order to describe the information models as shown in Clause 6.

Figure 8 shows how the root element “thing” is a generalization of an initial set of basic classes, known by this part of ISO 15926 which are derived from ISO 15926-2 entities which on their turn refer to classes within the RDL ISO/TS 15926-4.

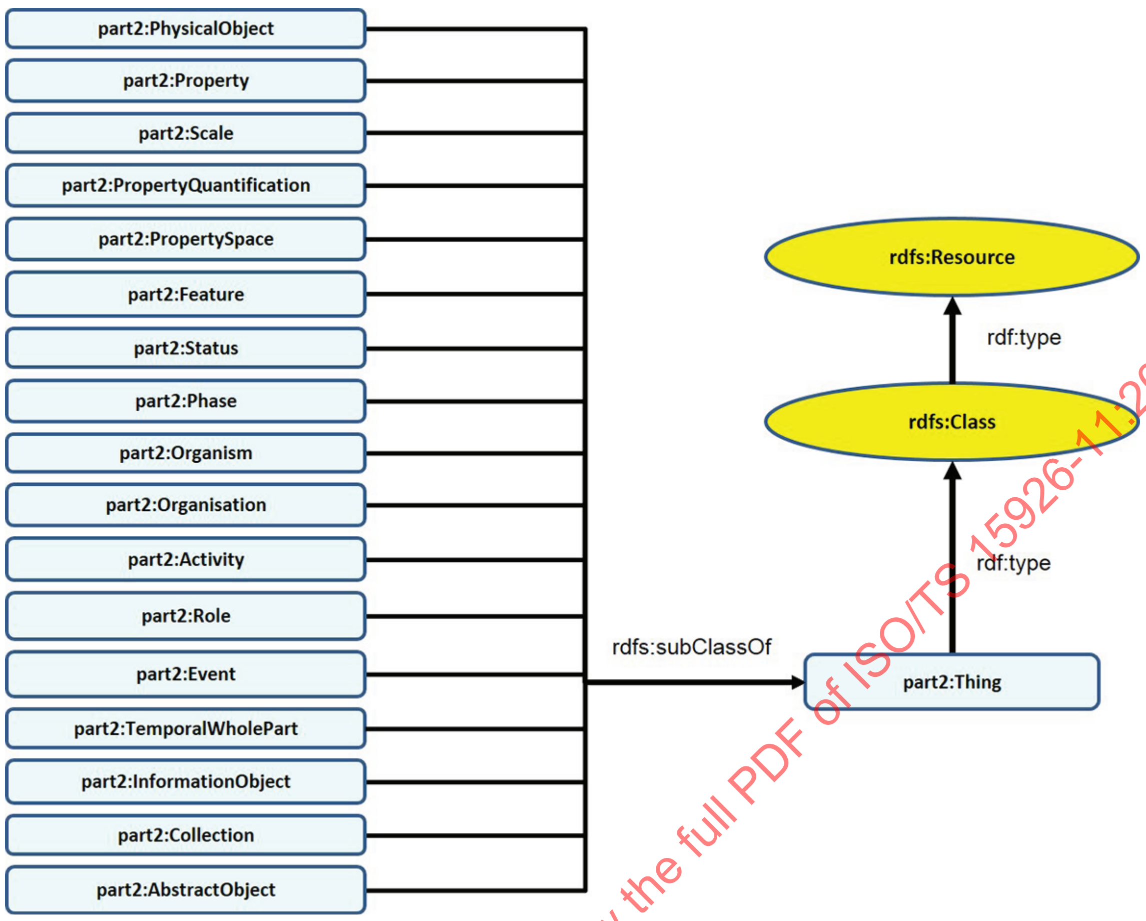

Figure 8 — Expanding RDF with basic classes of an RDL by mapping rdfs:class to part2:thing

The methodology described in this part allows defining private classes as specializations of classes in the Reference Data Library. These specializations then are part of the product, project and or process model and refer to a class in the Reference Data Library by means of the UID of the applicable class within ISO/TS 15926-4.

EXAMPLE Figure 9 shows an example of how an individual (instance of a RDL class) is related to rdfs:Resource.

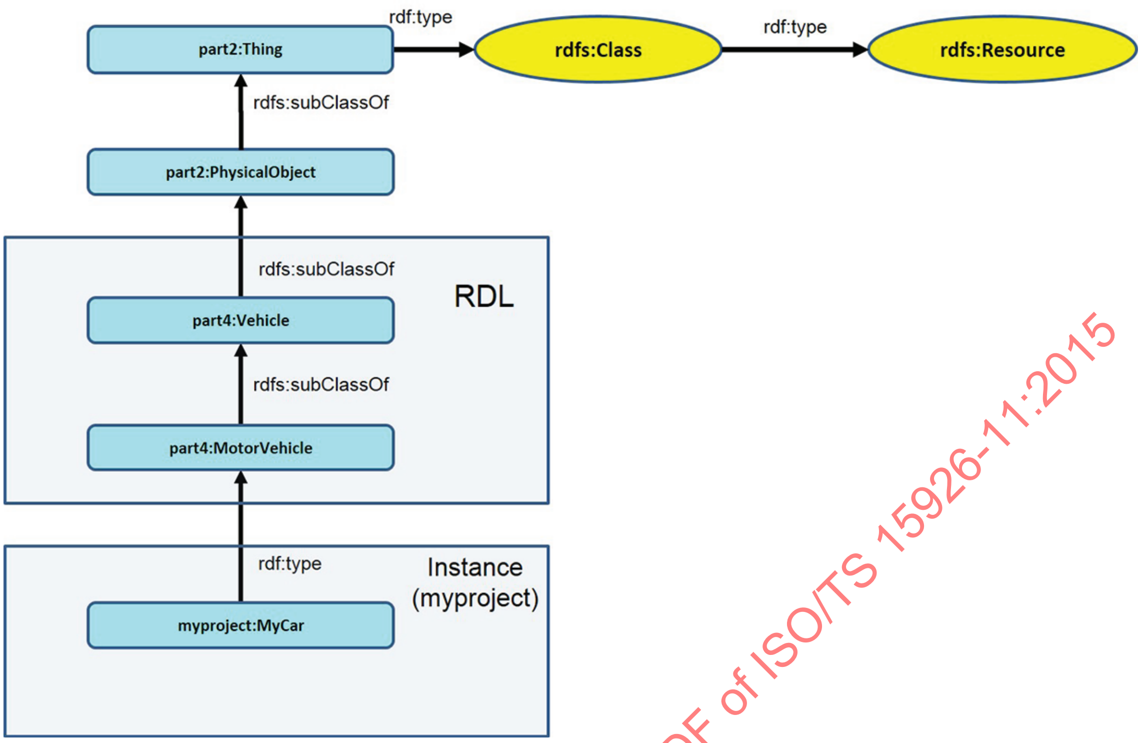

Figure 9 — Example of how an individual "myproject:MyCar" being an instance of a motor vehicle as defined in the library ISO/TS 15926-4 is related to rdfs:Resource

4.8 Use of RDF with a RDL of relationships

To expand RDF within this part of ISO 15926, a reference data library of relationships is incorporated. The root of this reference data library of relationships is "part11:relationship" which is defined as a rdf:type of rdf:Property. All relationships within the initial set of this part of ISO 15926 are a rdfs:subPropertyOf of part11:relationship.

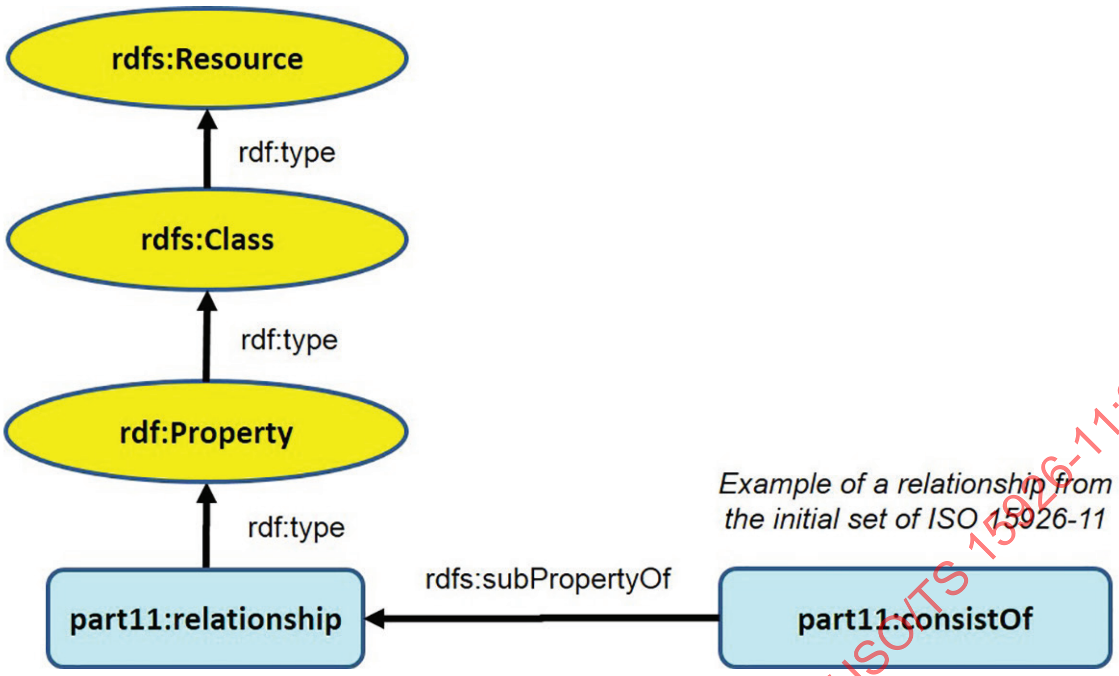

Figure 10 — Expanding RDF with a defined set of relationships with root element part11:relationship

There are two kinds of relationships defined in this part of ISO 15926:

— relationships in the context of making statements about Systems Engineering data;

EXAMPLE 1 "Objective A is threatened by risk B" is a statement in the context of Systems Engineering.

— relationships in order to make statements about statements (metadata).

EXAMPLE 2 "Statement A is created by person B" is a statement about a statement.

Both kinds of relationships are defined in the initial set of relationships as given in Annex A.

Rules:

— when statements are exchanged in the supply chain, only relationships in the prescribed direction shall be used. The inverse name of the relationship shall only be used in end-user application combined by changing the subject and object from position in the statement;

— when implementing this part of ISO 15926, each relationship can be specialized by means of domain and range relationships in order to select the right set of objects (on class level or on instance level) to connect with (see as example Figure 21).

By implementing this functionality within a configuration management environment will improve the semantic quality of statements and thus of information modelling.

NOTE For "part11:relationship" the same definition is valid as the entity "relationship" defined in ISO 15926-2.

5 Methodology for using Named Graphs

5.1 Building constructs of the methodology

In this clause, a semantic modelling methodology is described at logical level, consisting of a set of rules allowing users to combine RDF triples, RDF Named Graphs, set of relationships and RDL classes to build up product and or process models in a consistent manner. There are three types of "building constructs" defined within this methodology, all three represented by a specific Named Graph:

— an "individual graph" whereby any new object class or class instance will be introduced in the product, project and or process model. An "individual graph" is only a placeholder for that specific class or instance, and adds no semantics to that class or instance at this point, only defines a label for the URI of the object class or class instance;

— a "relationship class Graph". This kind of graph introduces, classifies and prescribes a relationship in terms of the name and allowed domain and range;

— a "statement graph" which will be used to make a concrete statement about another statement or any object introduced in an "individual graph".

These three types of Named Graphs allow integration of a class library and instances in one RDF Named Graph environment. Also specializations of classes of a Reference Data Library like ISO/TS 15926-4 can be integrated in the product, project and or process model.

Rules:

— every statement has its own type of RDF Named Graph as defined by this part of ISO 15926 by assigning one of these three types of Named Graphs;

— the relationships that are used are originating from the initial set of relationship given in Clause 7. The initial set is derived from the ISO/IEC 15288 reference models given in 6.2;

— for each relationship within the initial set is defined what the highest upward class in the hierarchy is defined for the subject and the object of a relationship. Each subclass of the defined highest upward class in the hierarchy is thus allowed as subject (domain) respectively the object (range) of a relationship;

— when one wants to restrict the usage of a relationship in terms of domain and range, a specialization of a specific relationship can be created using the relationship class Graph;

— every Named Graph has at least as metadata its creator and date/time of its creation.

There are two scenarios defined in this part how to manage changes in information represented by Named Graphs:

— modifications of named graphs are allowed: a modified named graph then gets as extra metadata: the date/time of modification, additional to the date/time of creation and creator;

— no modifications of any graph is allowed in order to support full traceability of history (nothing will be ever deleted). The specific graph, subject of the modification, will be replaced by a new one, with a "replaces" relationship to the URI of the superseded graph. This "replaces" relationship can be contained in the same named graph as the new Named Graph that supersedes the old Named Graph or can be a "is replaced by" relationship in the old Named Graph or can be contained in a separate Named Graph by means of a "is replaced by" relationship. Depending on the context of the application one can chose to implement one of these three mechanisms.

NOTE 1 For creator, created, modified is made use of the Dublin Core terms dc:creator, dc:created and dc:modified.

NOTE 2 For the "replace relationship" is made use of the Dublin Core term dc:isReplacedBy or dc:replaces.

5.2 Named Graph methodology

Figure 11 shows the three types of Named Graphs, introduced in 5.1. For this purpose, the RDF subclass rdfg:Graph is adopted from the W3C community as the root class for these three types of Named Graphs.

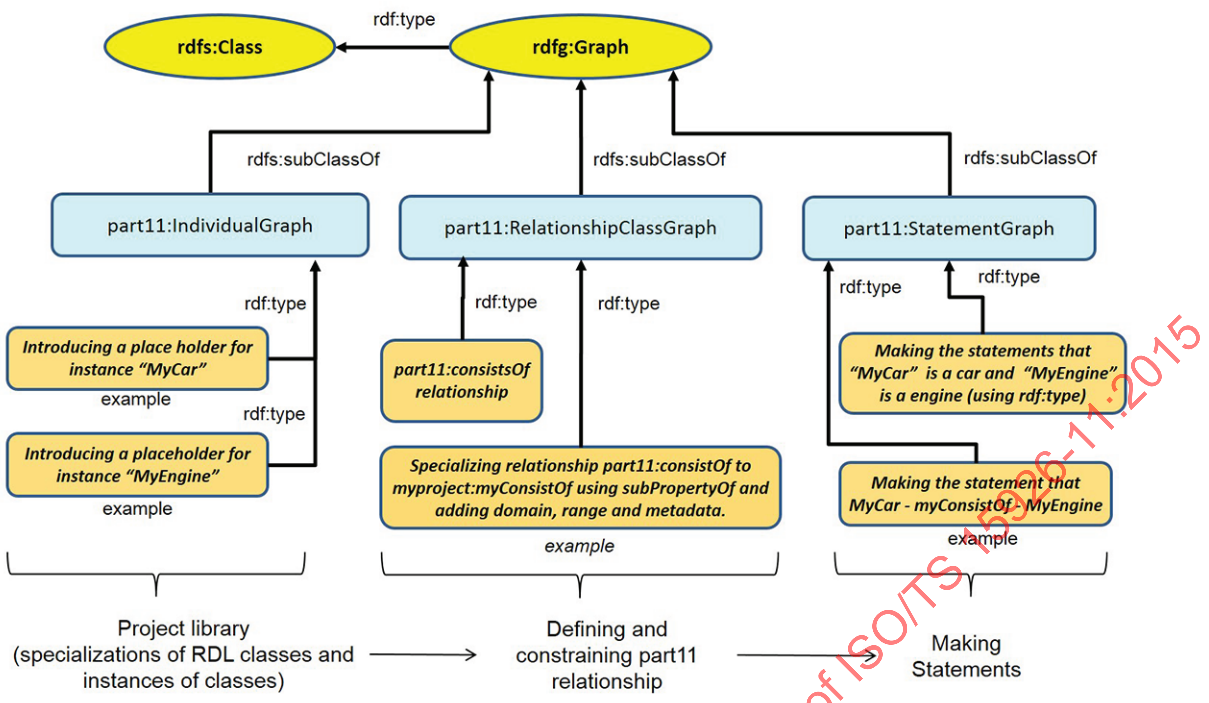
rdfs:Class
rdf:type
rdfg:Graph
rdfs:subClassOf
rdfs:subClassOf
rdfs:subClassOf
part11:IndividualGraph
part11:RelationshipClassGraph
part11:StatementGraph
rdf:type
rdf:type
rdf:type
rdf:type
rdf:type
Introducing a place holder for instance "MyCar"
example
rdf:type
Introducing a placeholder for instance "MyEngine"
example
part11:consistsOf relationship
Specializing relationship part11:consistOf to myproject:myConsistOf using subPropertyOf and adding domain, range and metadata.
example
Making the statements that "MyCar" is a car and "MyEngine" is a engine (using rdf:type)
Making the statement that MyCar - myConsistOf - MyEngine
example
Project library (specializations of RDL classes and instances of classes)
Defining and constraining part11 relationship
Making Statements

Figure 11 — Example of the use of the three types of Named Graphs defined by this part of ISO 15926

Figure 12 shows the way how to specialize relations from the initial set, fit to purpose for use in a project or product specific context.

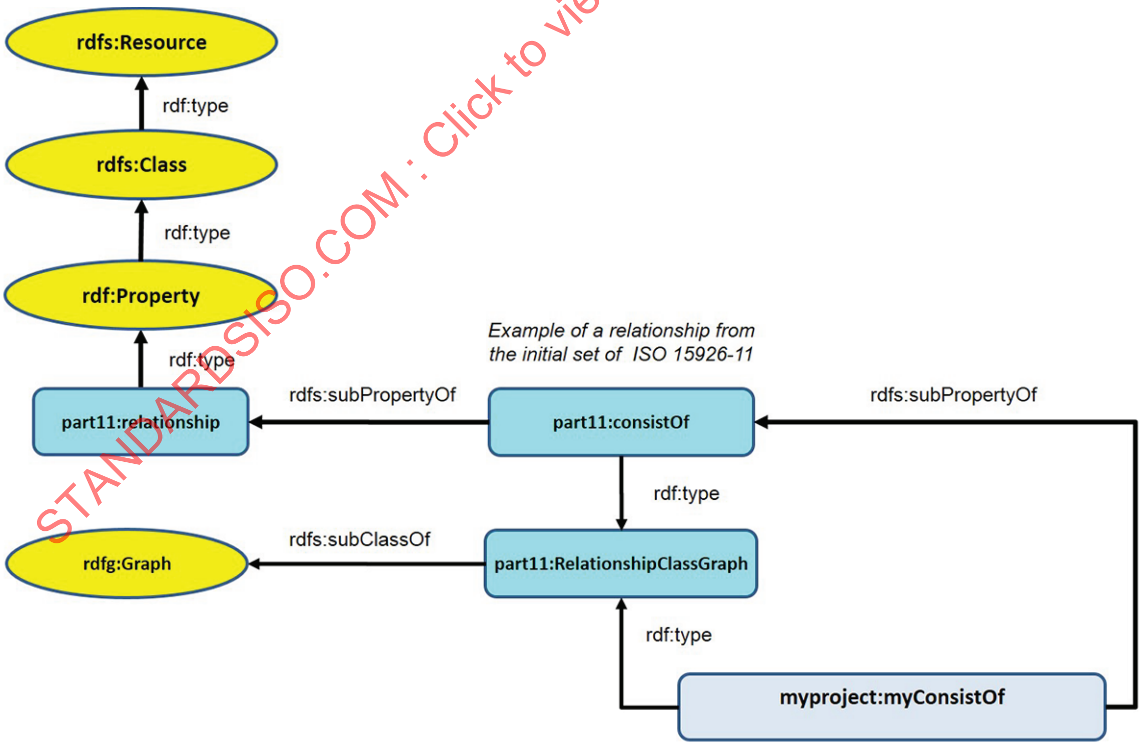

Figure 12 — Example of specializing a relationship from one out of the initial set of relationships

5.3 Metadata of a Named Graph

In order to position a Named Graph in his context, metadata can be added. For this purpose the next terms from the Dublin Core are adopted within this methodology (http://dublincore.org):

— dc:creator;

— dc:created;

— dc:modified;

— dc:source;

— dc:language;

— dc:description;

— dc:provenance;

— dc:replaces, dc:isReplacedBy.

From the RDFS namespace the next predicates are used to describe metadata of Named Graph:

— rdfs:label;

— rdfs:domain;

— rdfs:range.

Additional the next relationships are defined in the initial set of relationships in order to define metadata defining context of a Named Graph which is not covered by the Dublin Core and or RDF(S):

— part11:hasReverseLabel;

— part11:domainIsRestrictedByClass;

— part11:rangeIsRestrictedByClass;

— part11:hasModality;

— part11:hasIntention;

— part11:hasCertainty;

— part11:hasCardinality;

— part11:hasMinCardinality;

— part11:hasMaxCardinality;

— part11:hasAsBeginOfLife;

— part11:hasAsEndOfLife;

— part11:isModifiedBy;

— part11:concernsStage.

5.4 Example Named Graph methodology within projects

5.4.1 Introducing the basic types of Named Graphs within this part of ISO 15926

In this clause, the three building constructs of the Named Graph methodology of this part of ISO 15926 (see 5.1 and Figure 13 are defined by means of a specific Named Graph for each building construct as shown in Figure 13, Figure 14 and Figure 15 respectively. The Figures shown in Figure 13, 14 and 15 are centralized around the URI of the named graph, in Figure 13 “part11:IndividualGraph”.

Within the named Graph the relationships (predicates) are presented between this specific Named Graph and different kinds of objects like creator, date-time of creation and language, representing the metadata and classification of the Named Graph.

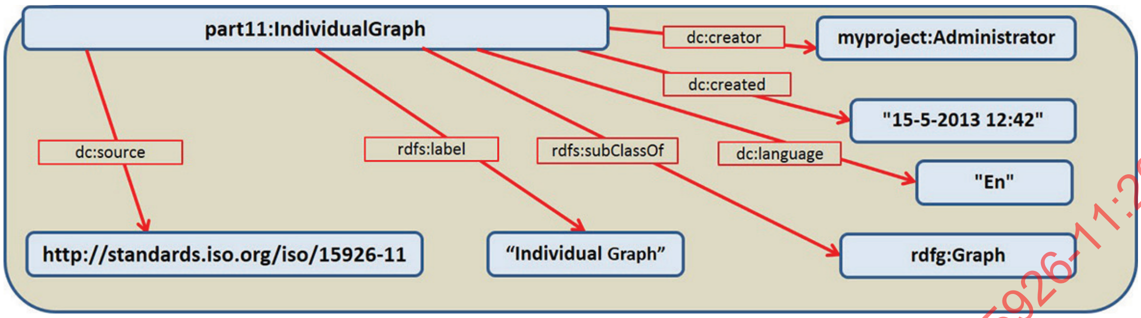

Figure 13 — Named Graph introducing the class “IndividualGraph”

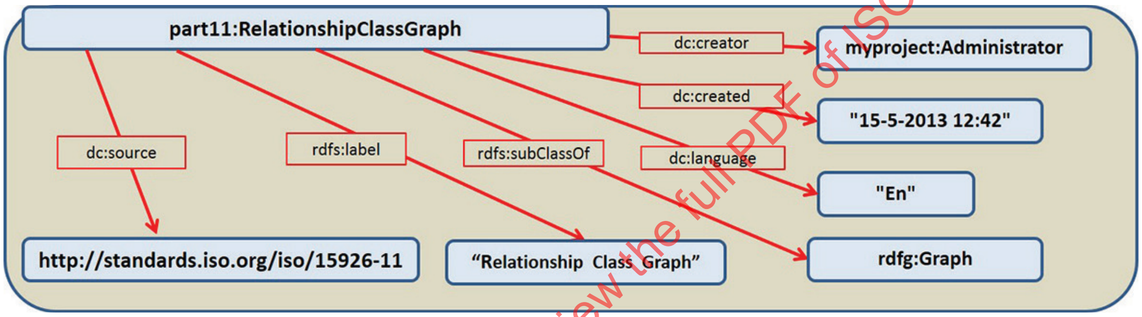

Figure 14 — Named Graph introducing the class “RelationshipClassGraph”

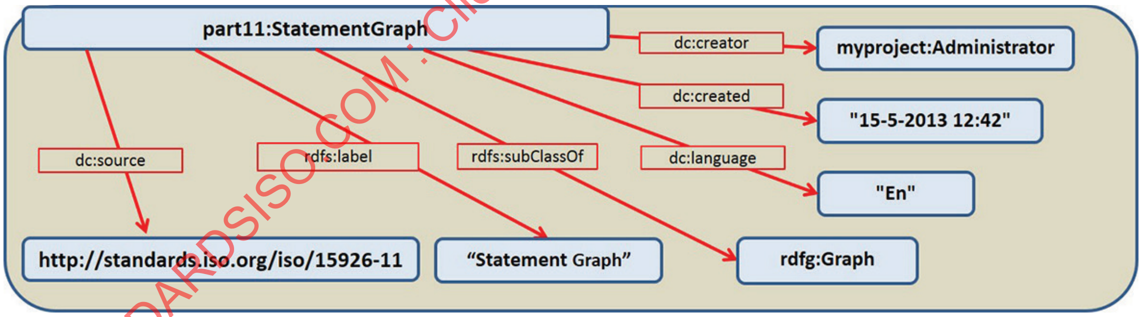

Figure 15 — RDF Named Graph introducing the class “StatementGraph”

The part11:relationship, being the root relationship of this part of ISO 15926 (see Figure 10), is defined by means of an instance of a part11:RelationshipClassGraph. Every relationship defined in the reference data library of relationships as defined in this part of ISO 15926 will be a rdfs:subPropertyOf of part11:relationship.

In Figure 16 this root relationship is introduced in this example project by means of the Named Graph with URI part11:relationship.

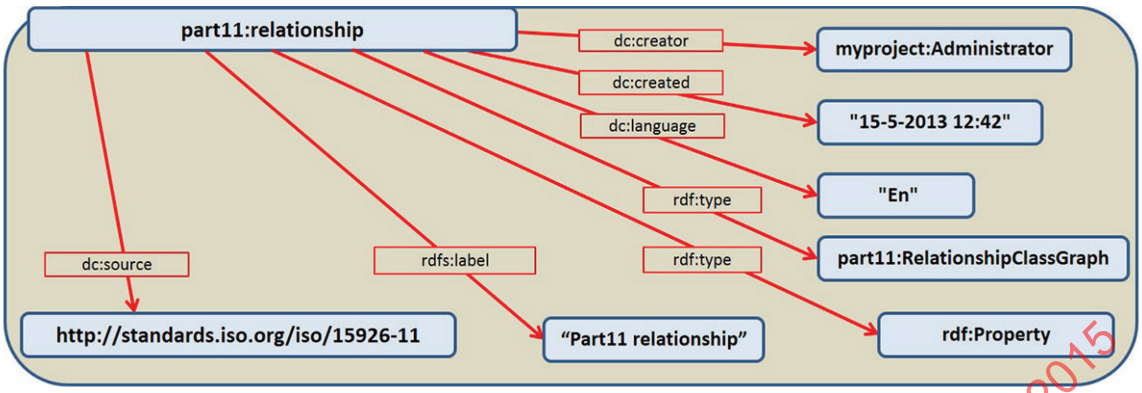

Figure 16 — RDF Named Graph introducing the root class "part11:relationship" as rdf:type of rdf:Property

5.4.2 Introducing project specific individuals

A set of Named Graphs can be used to define project specific individuals. In the following an example is worked out were a car (MyCar) consists of an engine (MyEngine) and the char has a property "Power of the car". This property will be quantified in a UoM, and the given property value will be replaced by a new value.

MyCar, MyEngine and the "Power of the car" are instances of classes that are contained in a RDL.

First, in Figure 17 the place holder for MyCar is introduced by means of an instance of an individual graph with the URI myproject:MyCar. In this Named Graph the label "MyCar"is assigned to this URI. Later on (Figure 28) MyCar will be classified as a car as defined in ISO 15926 part 4.

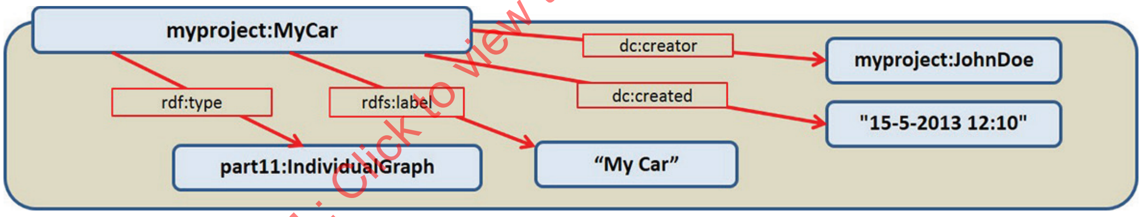

Figure 17 — RDF Named Graph introducing a placeholder for "MyCar"

In Figure 18 the place holder for MyEngine is introduced and will later on (Figure 29) be classified as an "engine" as defined in ISO/TS 15926-4.

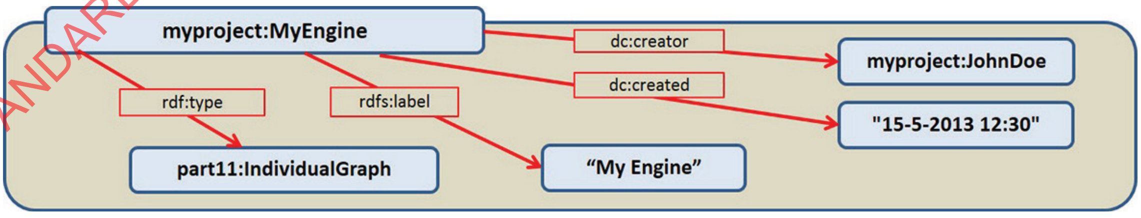

Figure 18 — RDF Named Graph introducing a placeholder for "MyEngine

In Figure 19 the place holder for the property "Power of my car" with URI myproject:PowerOfMyCar is introduced and will later on (Figure 33) be classified as "nominal power" according to its definition in ISO 15926 part 4.

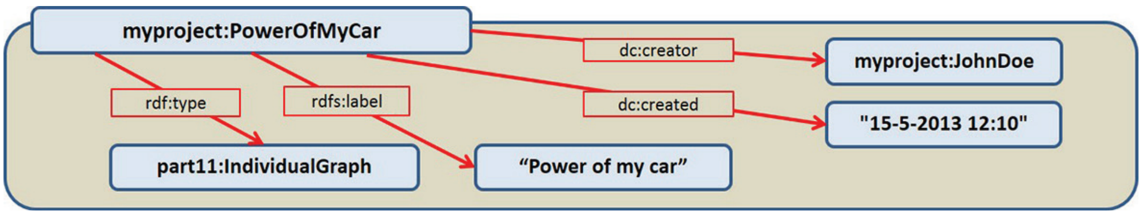

Figure 19 — RDF Named Graph introducing a placeholder for "PowerOfMyCar"

5.4.3 Introducing project specific relationships

First a basic version of the relationship "consists of" is introduced in the example project by means of a part11:RelationshipclassGraph with URI part11:consistsOf (Figure 20) with label "consists of" and is defined as an rdfs:subPropertyOf of part11:relationship as defined in Figure 16.

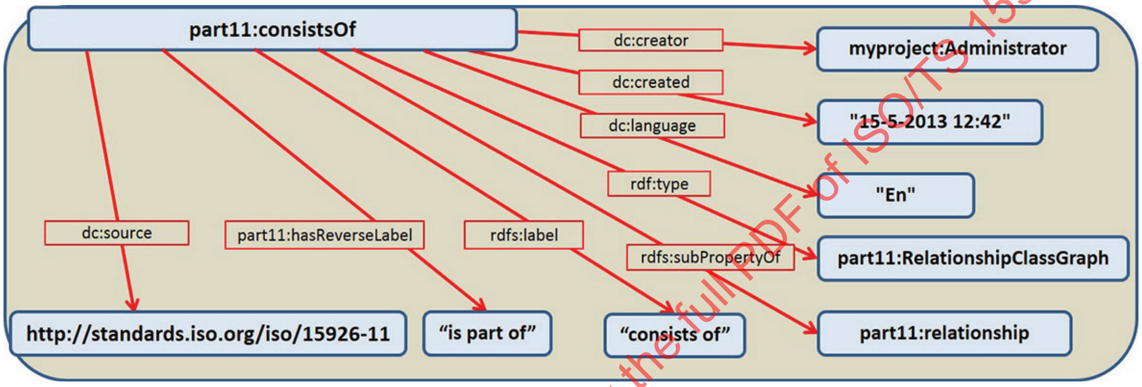

Figure 20 — RDF Named Graph introducing the part11:consistsOf relationship as a rdfs:subPropertyOf of part11:relationship

Second the in Figure 20 introduced basic version of the relationship "consists of" is further specialized for this example to a relationship in the namespace "myproject" with a domain and range both declared as instances of the RDL class "physical object". The URI is myproject:myConsistOf and has label "consist of Physical Object".

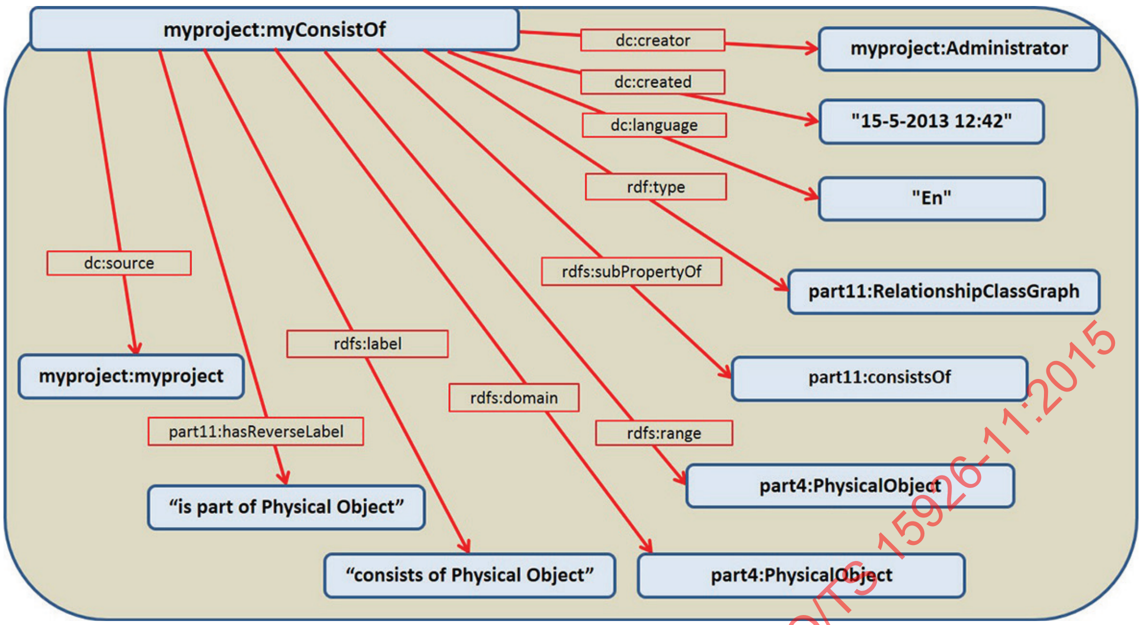

Figure 21 — RDF Named Graph specializing within the namespace "myproject" the in Figure 20 introduced relationship with label "consistsOf into the relationship with label "consistsOfPhysicalObject"

In Figure 22 the relationship with label "has modality" from the initial set is introduced with URI part11:hasModality with label "has modality" as an rdfs:subPropertyOf of part11:relationship (see Figure 16).

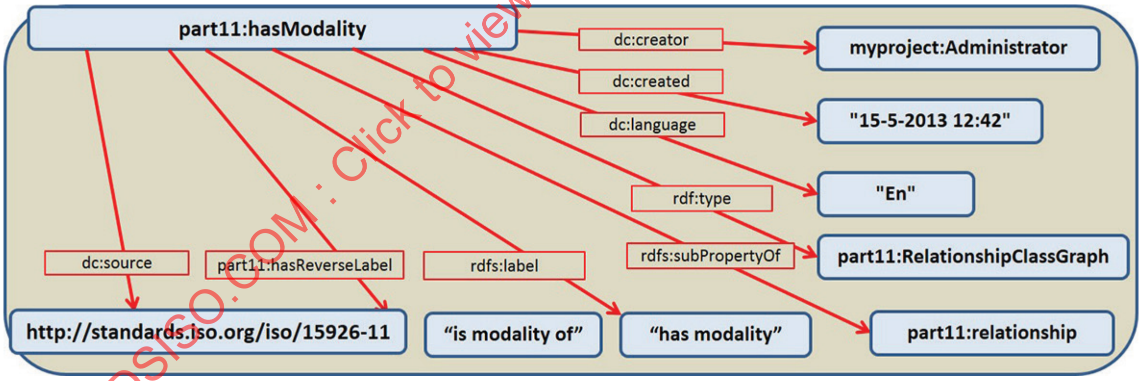

Figure 22 — RDF Named Graph introducing the part 11 relationship with label "has modality"

In Figures 23, 24, 25, 26 and 27 the following relationships are introduced in the part11 namespace in the same way as in Figure 22 "part11:hasModality": part11:isQuantifiedIn, part11:hasMagnitude, part11:hasCertainty, part11:hasIntention, and part11:hasProperty.

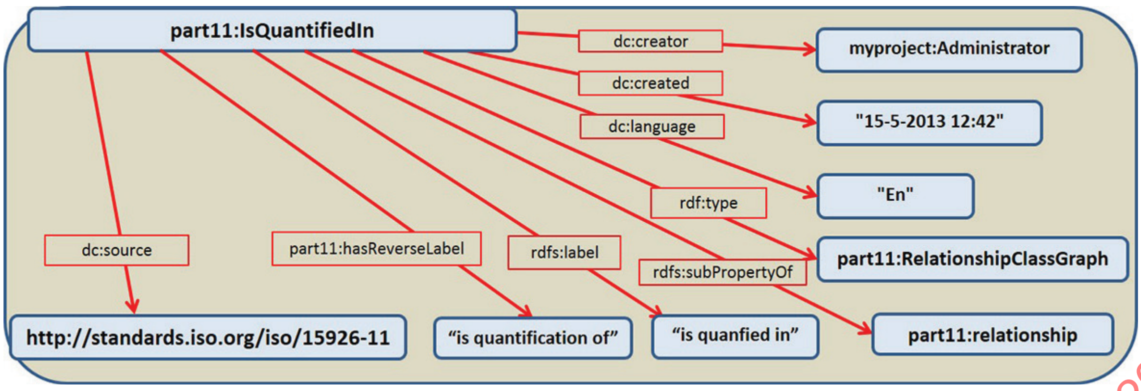

Figure 23 — RDF Named Graph introducing the part 11 relationship with label "is quantified in"

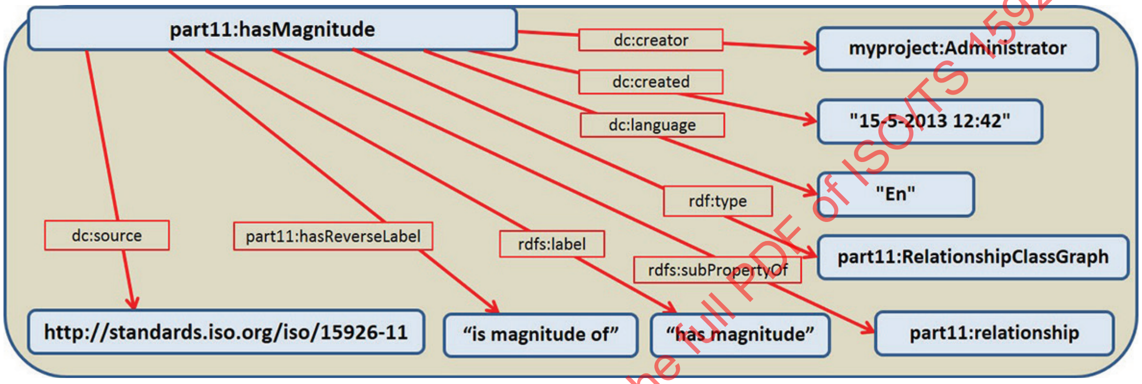

Figure 24 — RDF Named Graph introducing the part 11 relationship with label "has magnitude"

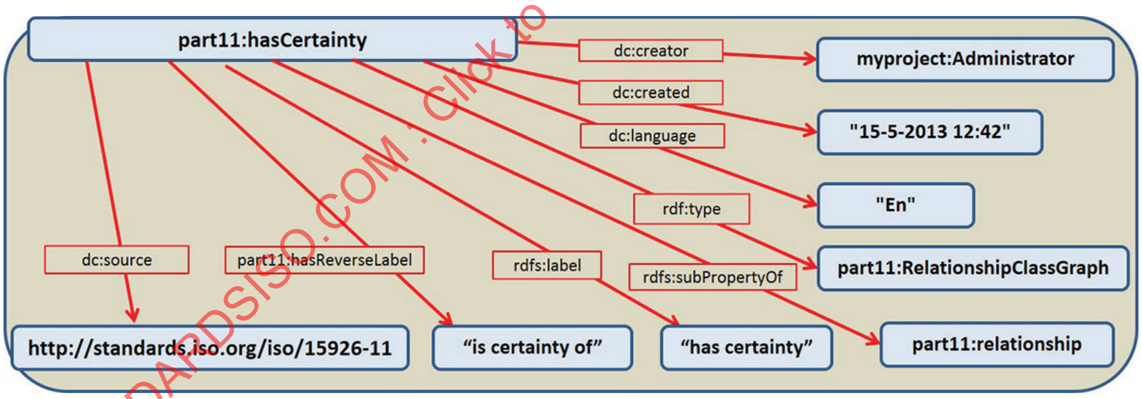

Figure 25 — RDF Named Graph introducing the part 11 relationship with label "has certainty"

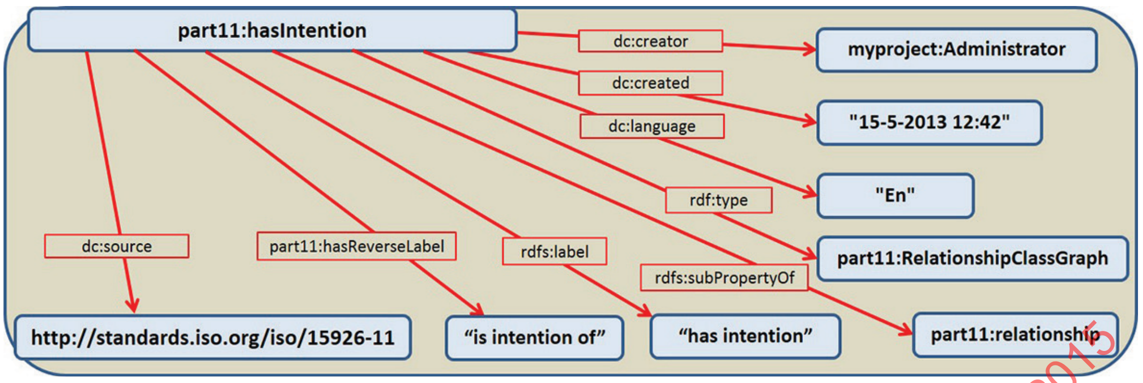

Figure 26 — RDF Named Graph introducing the part 11 relationship with label "has 6intention"

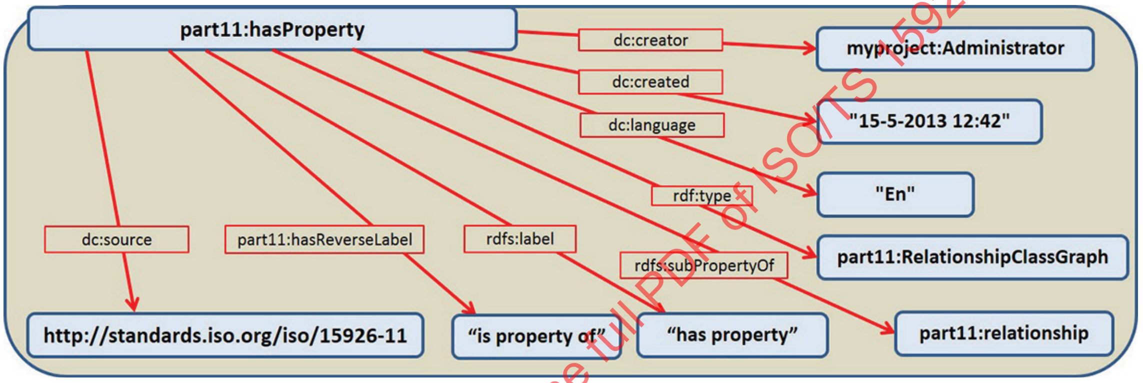

Figure 27 — RDF Named Graph introducing the part 11 relationship with label "has property"

5.4.4 Introducing project specific statements

In the previous clauses the placeholders are introduced and the needed relationships are defined. In this clause, both are used to make statements about the configuration of the product and to make statements about statements. In this example the labels of the placeholders has been put in separate statements, supporting controlled change of these labels. In principle these labels can also be part of the Named Graph that introduces the placeholder.

In Figure 28 URI myproject:MyCar with label "MyCar" is stated to be an instance from the ISO/TS 15926-4 class "part4:MotorVehicle".

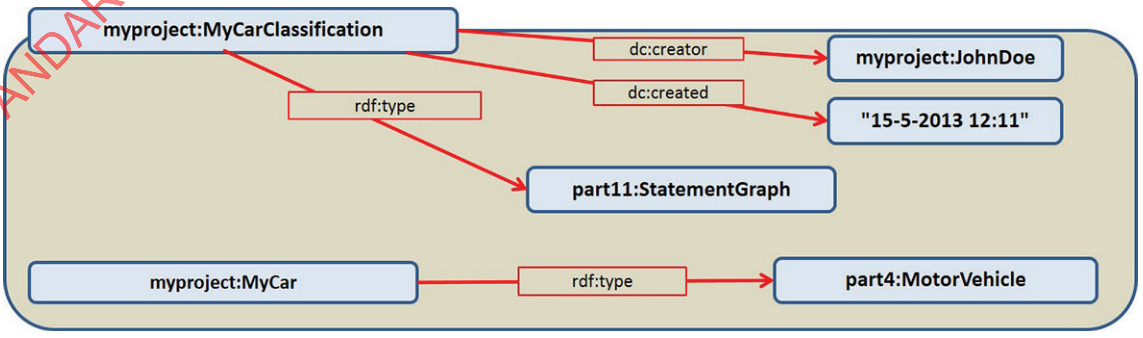

Figure 28 — RDF Named Graph stating that "MyCar" is an instance of the RDL class part4:MotorVehicle"

In Figure 29 the statement is made that URI myproject:MyEngine with label "My Engine" is an instance of the ISO/TS 15926-4, 4 class "part4:Engine".

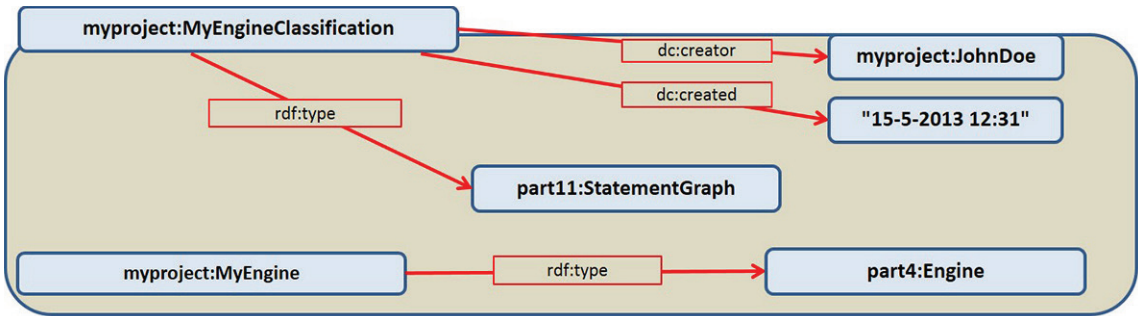

Figure 29 — RDF Named Graph stating that "MyEngine" is an instance of the RDL class "part4:Engine"

In Figure 30 the statement is made by URI myproject:MyCarConsistOfMyEngine that myproject:MyCar consist of myproject:MyEngine by means of the predicate with URI myproject:myConsistOf.

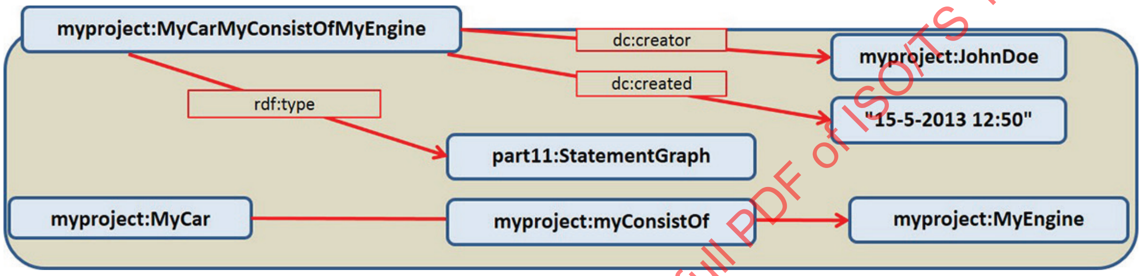

Figure 30 — RDF Named Graph stating that "MyCar consist of MyEngine"

In Figure 31 the statement is made that myproject:MyCar has a property myproject:PowerOfMyCare using the part11:hasProperty relationship.

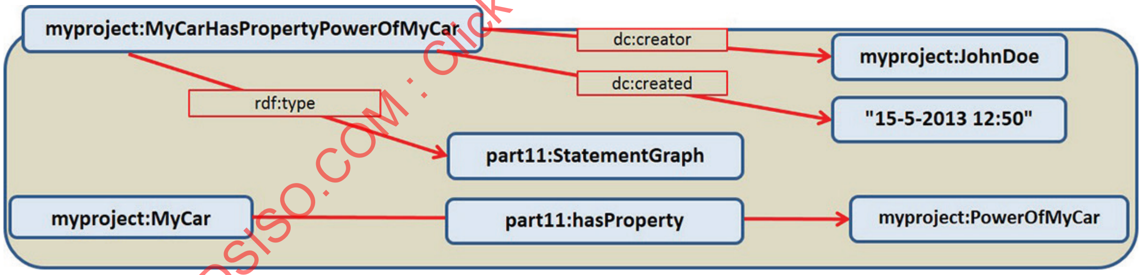

Figure 31 — RDF Named Graph stating that "MyCar" has a property "PowerOfMyCar"

In Figure 32 the statement is made that it is required ("shall have") that myproject:MyCar has a property myproject:PowerOfMyCar.

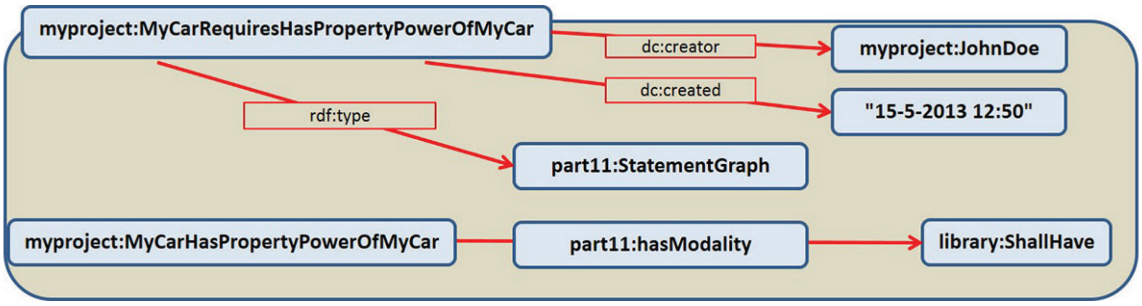

Figure 32 — RDF Named Graph stating that the statement the MyCar "shall have" a property PowerOfMyCar by means of the relationship part11:hasModality

In Figure 33 the statement is made that the property myproject:PowerOfMyCar is an instance of ISO/TS 15926-4 class nominal power.

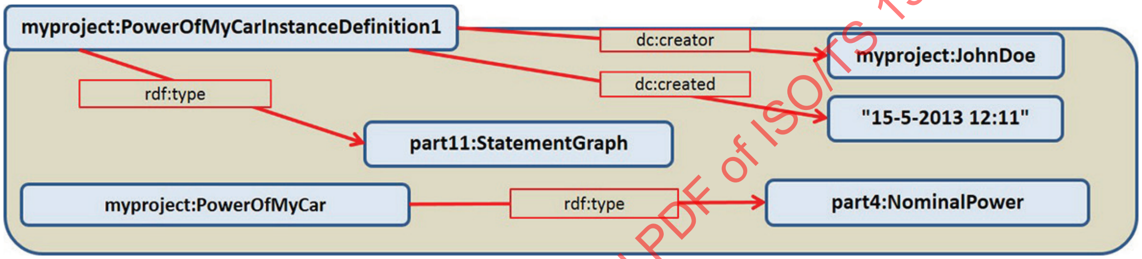

Figure 33 — RDF Named Graph stating that the statement that the property PowerOfMyCar is classified as nominal power according to the RDL class "NominalPower"

In Figure 34 the statement is made that the property myproject:PowerOfMyCar has to be quantified by means of the ISO/TS 15926-4 class kW.

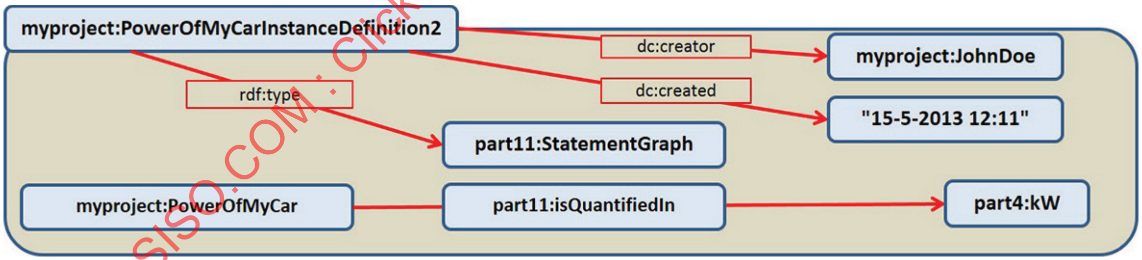

Figure 34 — RDF Named Graph stating that the statement that the property myproject:PowerOfMyCar is expressed in kW according to the RDL class "kW"

In Figure 35 the statement is made that the property myproject:PowerOfMyCar has a magnitude of 240.

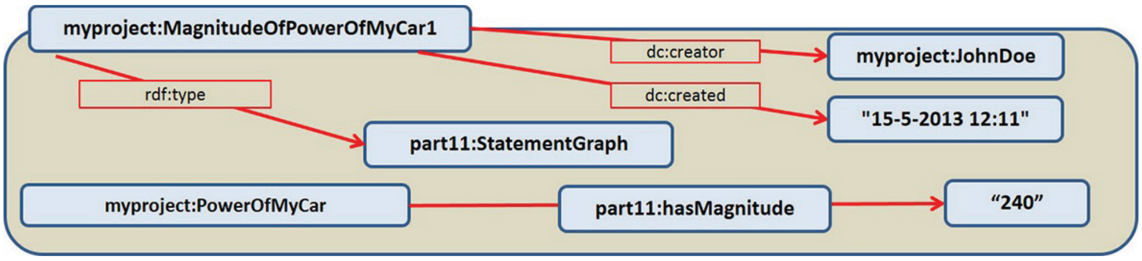

Figure 35 — RDF Named Graph stating that the statement that the property PowerOfMyCar has a magnitude of 240

In Figure 36 the statement is made that the magnitude of the property myproject:PowerOfMyCar (being "240") is a proposal.

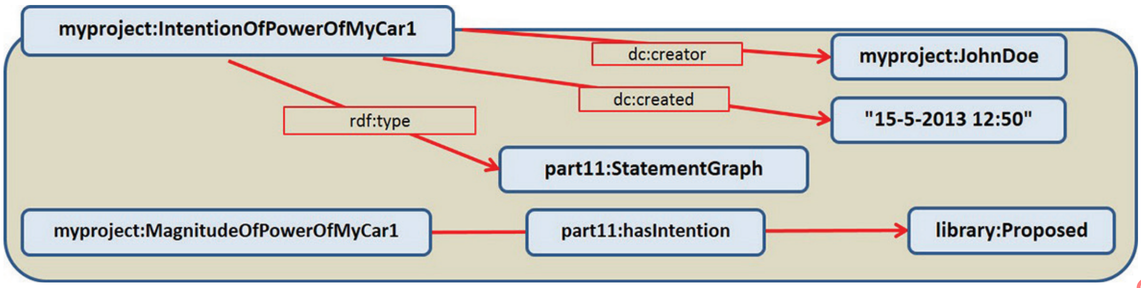

Figure 36 — RDF Named Graph stating that the statement that the magnitude (being "240") of property myproject:PowerOfMyCar is a proposed value

In Figure 37 the statement is made that the magnitude of the property myproject:PowerOfMyCar is an estimated value.

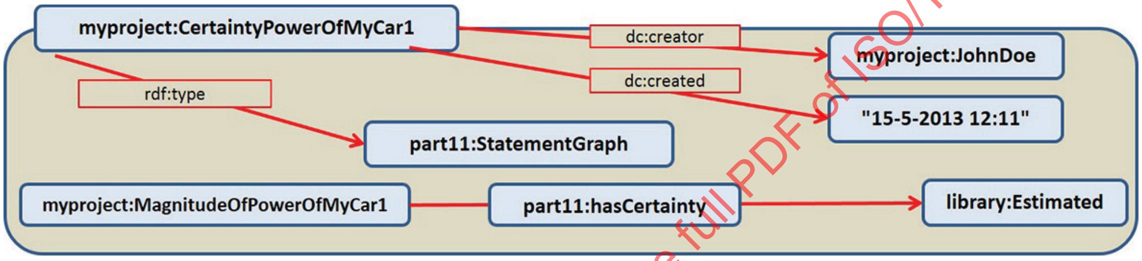

Figure 37 — RDF Named Graph stating that the statement that the magnitude of property PowerOfMyCar is an estimated value

In Figure 38 the statement is made that the magnitude of the property myproject:PowerOfMyCar has been changed to 250. By means of dc:provenance this change is justified. This Named Graph replaces the Named Graph of Figure 35.

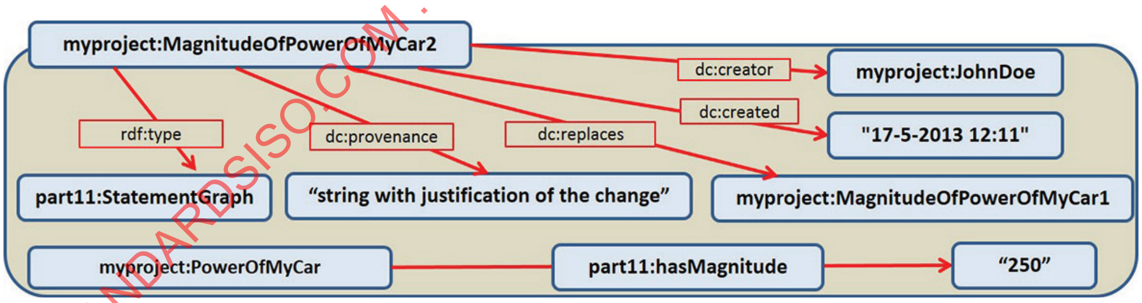

Figure 38 — RDF Named Graph stating that the statement that the magnitude of property powerOfMyCar has been changed in 250 with a justification of the change

In Figure 39 the statement is made that the magnitude (being "250") of the property myproject:PowerOfMyCar is approved.

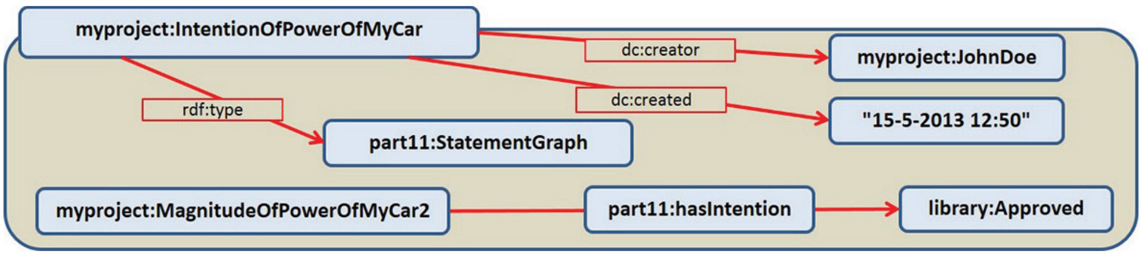

Figure 39 — RDF Named Graph stating that the statement that the magnitude of property PowerOfMyCar is approved

In Figure 40 the statement is made that the magnitude (being "250") of the property myproject:PowerOfMyCar is a calculated value.

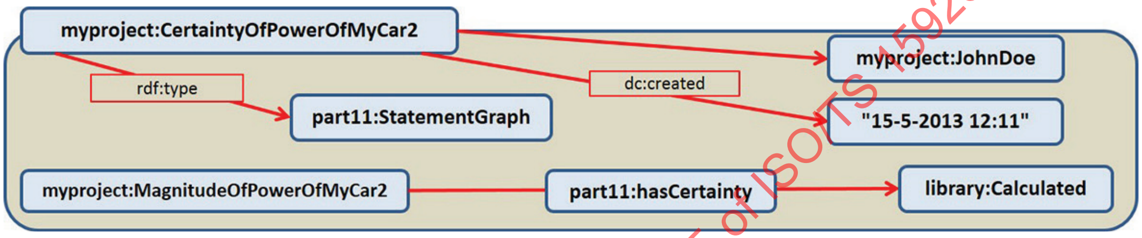

Figure 40 — RDF Named Graph stating that the statement that the magnitude of property PowerOfMyCar is a calculated value

5.5 Example Named Graph methodology within product knowledge modelling

In this clause, an example is given of how this part of ISO 15926 can be used to define product models on class level. The approach is identical to project specific or instance model as shown in the 5.3. First the placeholders for classes of part of the products are introduced. Second the required relations are defined and third the statements are made including the statements about statements.

The example describes how a type of pump can be defined by a party using this part of ISO 15926 in combination with a reference data library (RDL). The type of pump must consist of minimal 2 and maximal 4 pump impellers and has a property power of that type of pump.

In this example are relationships reused from the part11 namespace relationships as defined in 5.4.

5.5.1 Introducing product specific individuals

In Figure 41 the place holder for the product "Pump type A" is introduced by means of an instance of an individual graph.

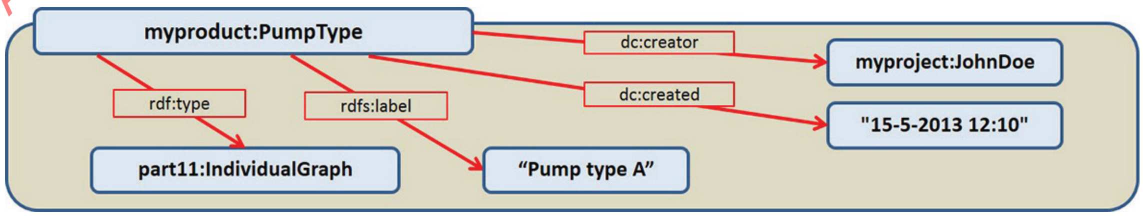

Figure 41 — RDF Named Graph introducing a placeholder for "PumpType"

In Figure 42 the place holder for the product part "Impeller type B" is introduced by means of an instance of an individual graph.

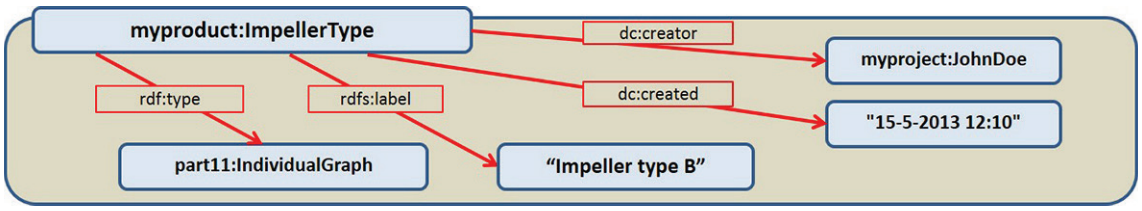

Figure 42 — RDF Named Graph introducing a placeholder for "ImpellerType"

In Figure 43 the place holder for the property "Nominal power of pump type A" is introduced by means of an instance of an individual graph.

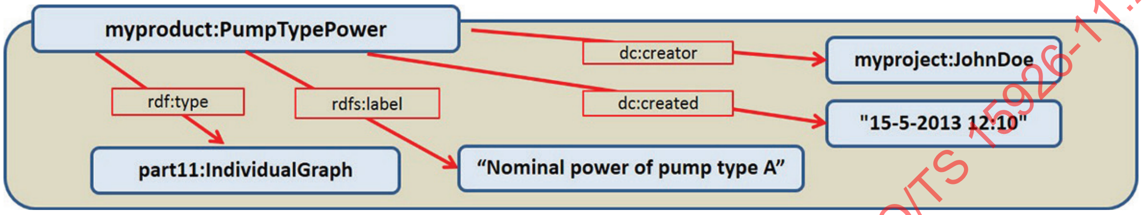

Figure 43 — RDF Named Graph introducing a placeholder for "PumpTypePower"

5.5.2 Introducing project specific relationships

In Figure 44 the relationship "part11:consistsOf" from Figure 20 is specialized to a relationship with a domain that is limited to subclasses of ISO/TS 15926-4 "physical object" and were the range is limited to subclasses of ISO/TS 15926-4 "physical object".

Since rdf:domain and rdf:range both point to instances of classes is in this Named Graph the part11:domainIsRestrictedByClass and part11:rangeIsRestrictedByClass relationships used to specify the domain and range, specific mend for pointing to (sub)classes.

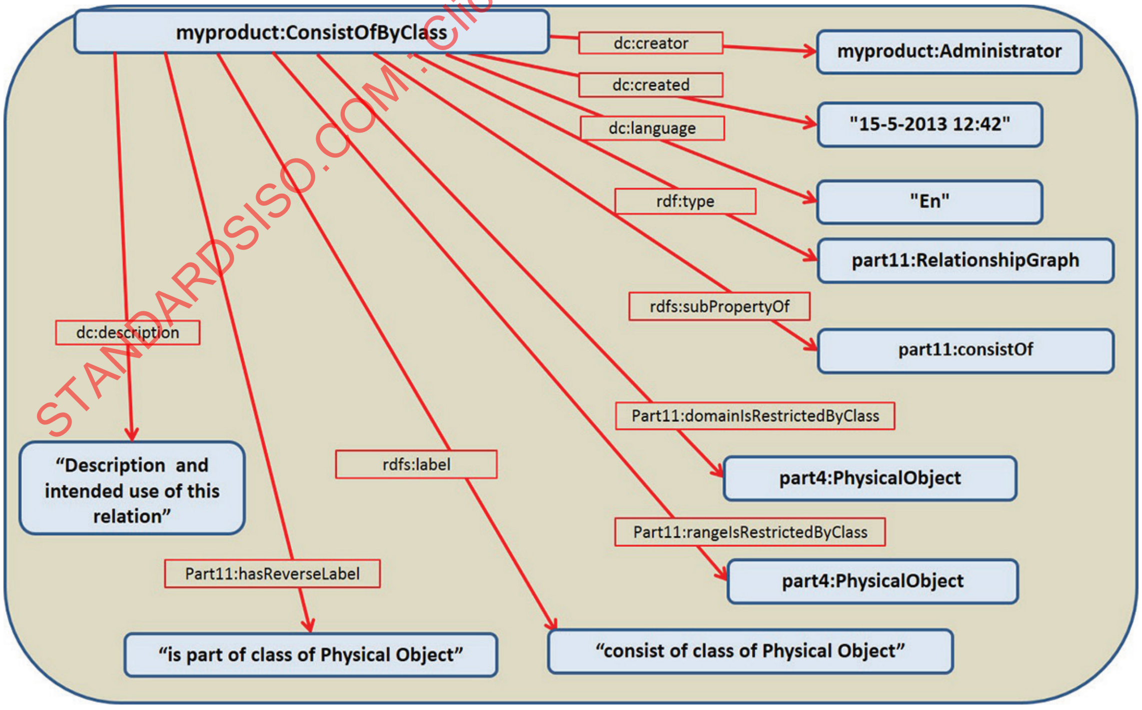

Figure 44 — RDF Named Graph specializing the part11:consistsOf relationship to a myproduct:consistsOfByClassOfPhysicalObject relationship

In Figure 45 the relationship “hasMinCardinality” from the initial set of relationships is introduced in the product model.

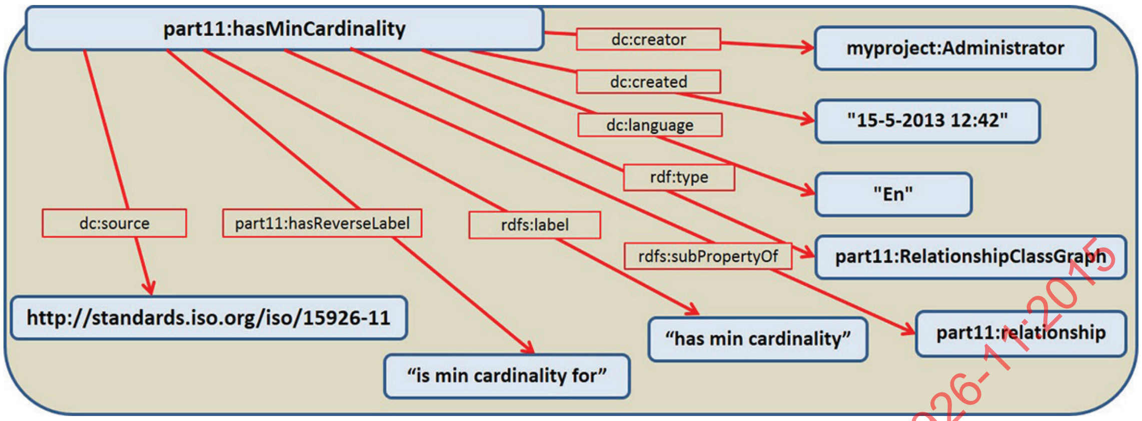

Figure 45 — RDF Named Graph defining the MinCardinality relationship from the initial set of relationships

In Figure 46 the relationship “hasMaxCardinality” from the initial set of relationships is introduced in the product model.

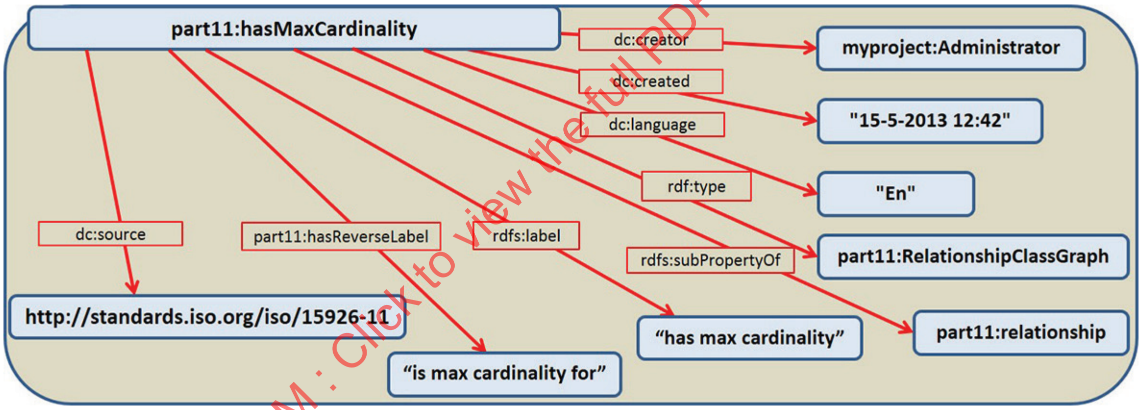

Figure 46 — RDF Named Graph introducing the hasMaxCardinality relationship from the initial set of relationships

5.5.3 Introducing product specific statements

In Figure 47 myproduct:PumpType (with label “Pump type A”) is classified as a subclass of the class “pump” defined in ISO/TS 15926-4.

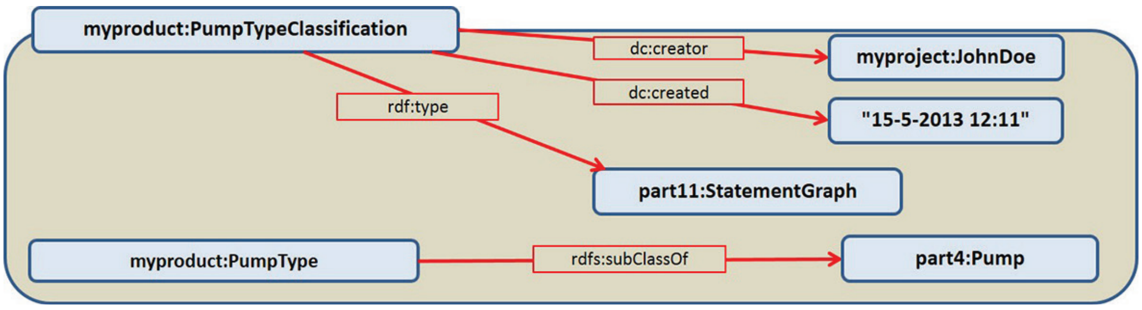

Figure 47 — RDF Named Graph classifying myproduct:PumpType as a subclass of “pump” as defined in ISO/TS 15926-4

In Figure 48 myproduct:ImpellerType (with label “Impeller type B”) is classified as a subclass of the class “impeller” defined in ISO/TS 15926-4

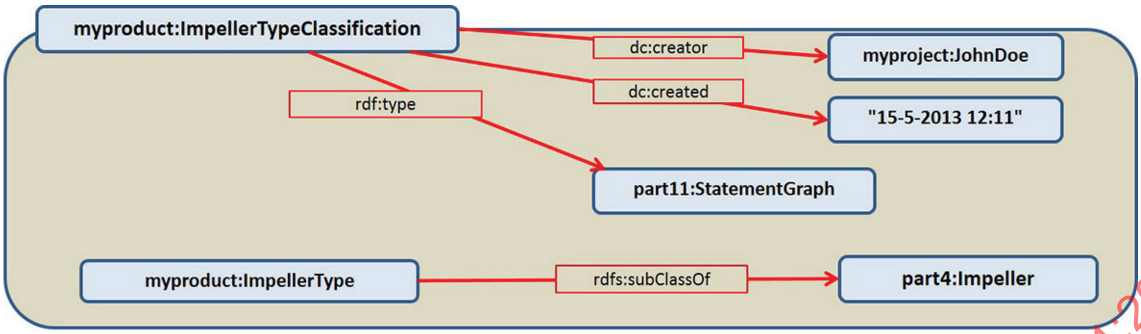

Figure 48 — RDF Named Graph classifying myproduct:ImpellerType as a subclass of impeller from ISO/TS 15926-4

In Figure 49 the statement is made that the class myproduct:pumpType (with label “Pump type A”) consists of the class myProduct:impellerType (with label “Impeller type B”).

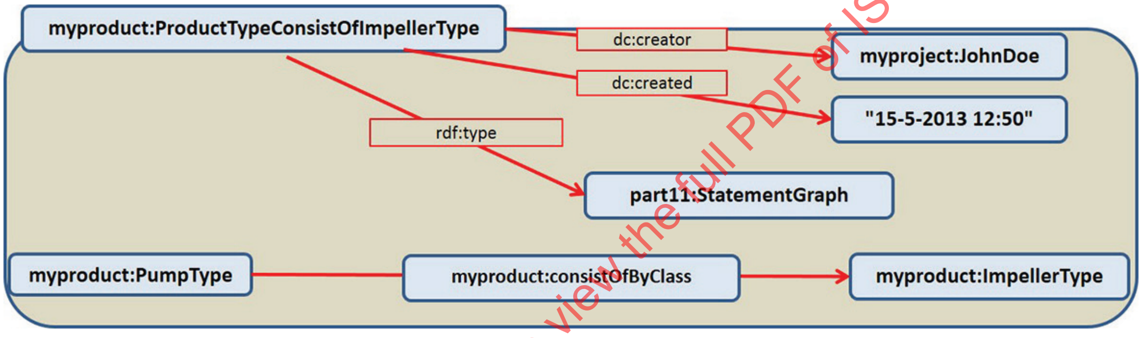

Figure 49 — RDF Named Graph stating that the class myproduct:PumpType typically consist of the class myproduct:ImpellerType

In Figure 50 the statement is made that the statement myProduct:ProductTypeConsistOfImpellerType has a minimum cardinality of two. So an instance of the class myproduct:PumpType (with label “Pump type A”) consist of minimal two instances of myproduct:impellerType (with label “Impeller type B”).

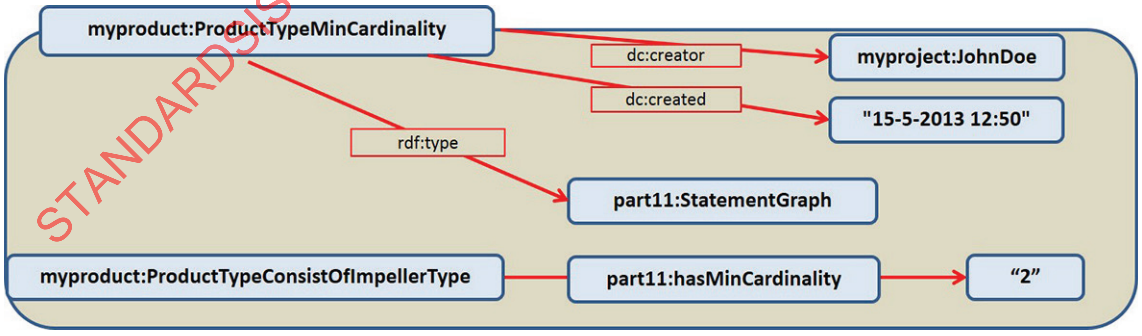

Figure 50 — RDF Named Graph stating that an instance of the class myproduct:PumpType consist of minimal two instances of myproduct:ImpellerType”2”

In Figure 51 the statement is made that the statement myproduct:ProductTypeConsistOfImpellerType has a maximum cardinality of four. So an instance of the class myproduct:PumpType (with label “Pump type A”) consist of maximal four instances of myproduct:ImpellerType (with label “Impeller type B”).

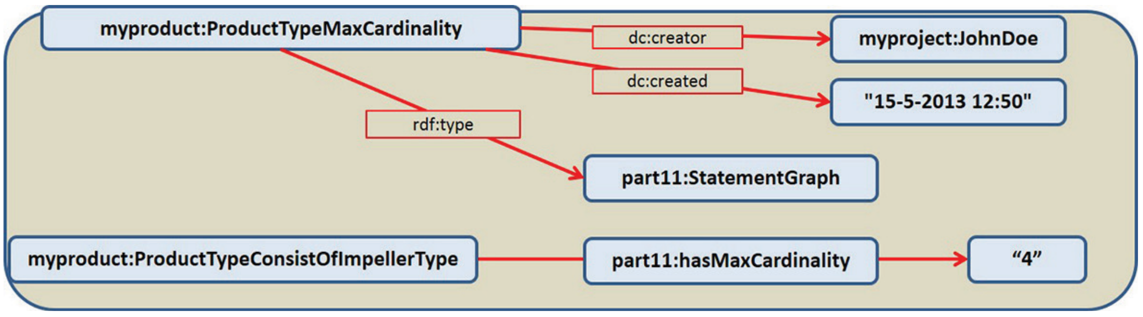

Figure 51 — RDF Named Graph stating that an instance of the class myproduct:PumpType consist of maximal four instances of myproduct:ImpellerType"4"

In Figure 52 the statement is made that myproduct:PumpTypePower (with label "Nominal power of pump type A") is an instance of the class nominal power in ISO/TS 15926-4.

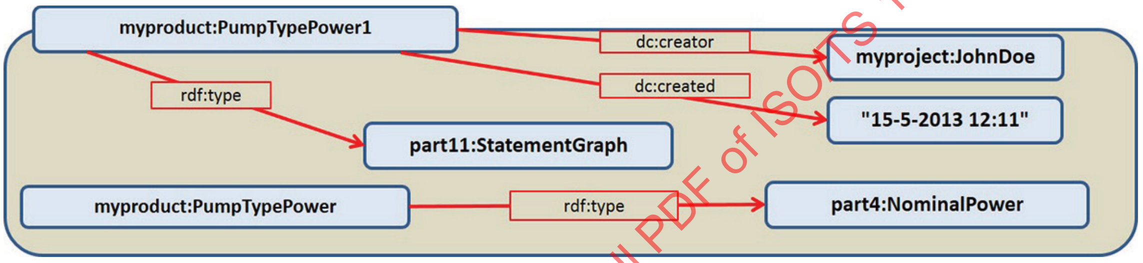

Figure 52 — RDF Named Graph classifying myproduct:PumpTypePower as "Nominal Power" according to the definition of this class in ISO/TS 15926-4

In Figure 53 presents the statement that myproduct:PumpTypePower is quantified by the UoM kW from ISO/TS 15926-4.

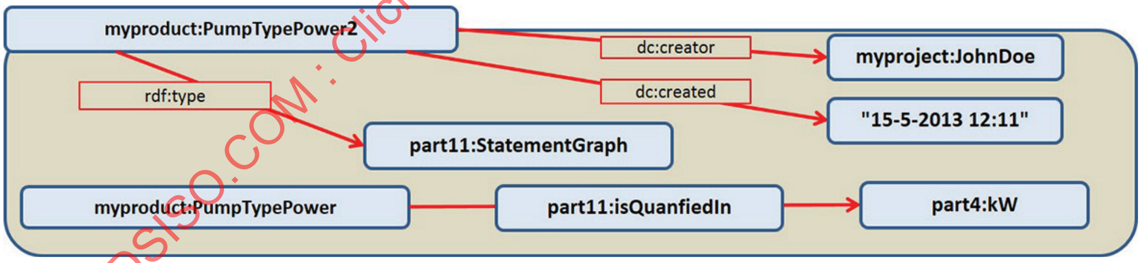

Figure 53 — RDF Named Graph stating that myproduct:PumpTypePower is quantified in kW according to its definition in ISO/TS 15926-4

Figure 54 presents the statement that the quantification of myproduct:PumpTypePower in kW is optional by means of the modality value "can be" as defined in the namespace "library".

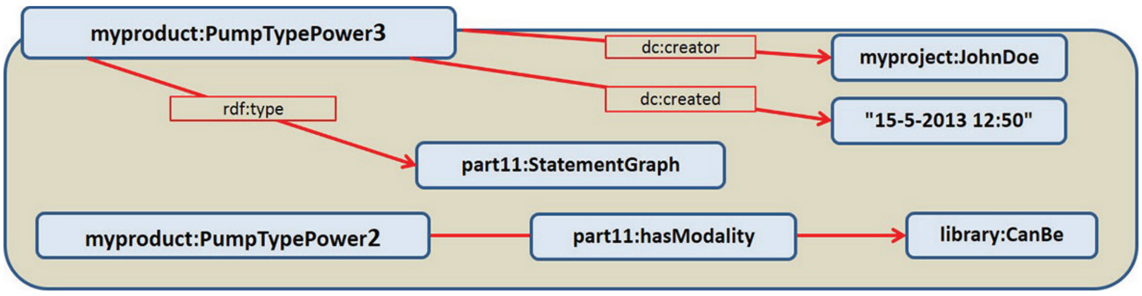

Figure 54 — RDF Named Graph stating that the statement myproduct:PumpTypePower2 has as modality "can be"

In Figure 55 presents the statement that myproduct:PumpTypePower is quantified by the UoM hp from ISO/TS 15926-4.

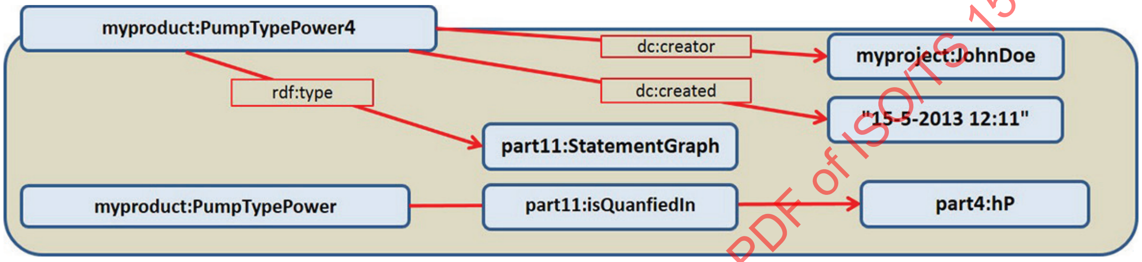

Figure 55 — RDF Named Graph stating that myproduct:PumpTypePower is quantified in hp according to its definition in ISO/TS 15926-4

Figure 56 presents the statement that the quantification of myproduct:PumpTypePower in hp is optional by means of the modality value "can be" as defined in the namespace "library" with respect to the Named Graph "myproduct:PumpTypePower4".

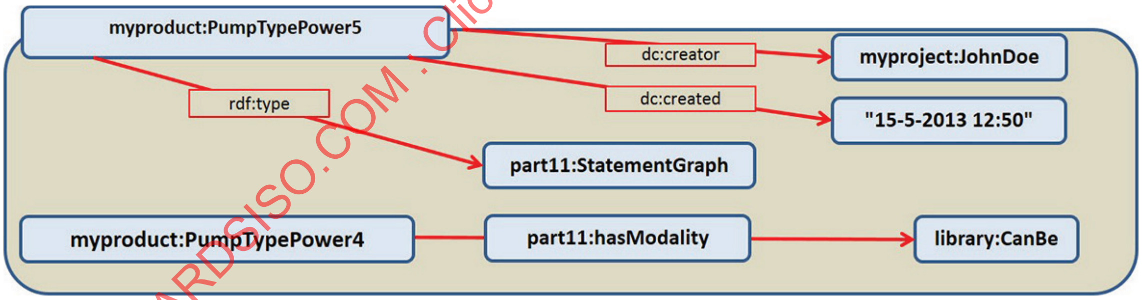

Figure 56 — RDF Named Graph stating that the statement myproduct:PumpTypePower4 has as modality "can be"

6 Reference Data

6.1 Origin of the initial set of relationships

Engineering data in the context this part of ISO 15926 covers not only the output and or input of engineering processes, but also the engineering processes themselves. In this way, this part of ISO 15926 enables the integration and exchange of engineering data together with data, relevant in the context of Systems Engineering data. Other parts of ISO 15926 don't cover sufficiently the area of Systems Engineering aspects in exchanging information over de lifecycle of a plant.

EXAMPLE A requirement, test method and maintenance activity are examples of system engineering data.

In order to achieve a set of relationships, applicable in the context of Systems Engineering, for this part of ISO 15926, ISO/IEC 15288 is used as a primary source.

Within ISO/IEC 15288, 25 processes have been defined in order to realize a system, starting from the statement of purpose (objective) of a system and a set of top level (stakeholder) requirements ending in operating and maintaining that system.

Looking at the generic System Life Cycle Process descriptions given in ISO/IEC 15288 a set of information models have been derived in preparation for the initial set of relationships defined in this part of ISO 15926. This set of information models functions as a reference ontology for Systems Engineering. The information models therefore can also be used as reference models in Systems Engineering projects.

For an example, one of outputs of contractor, information structure can be agreed between client and contractor and specified in an Information Deliverable Manual (IDM). Agreed information structure can be a subset of the presented reference models in this clause (and eventually be modified).

This part of ISO 15926 enables implementation of these Systems Engineering information reference models in an explicit way, defining all the things that are addressed and related in these models. Based on such implementation, one can build configuration and information management systems and herewith communicate Systems Engineering information in an explicit an unambiguous way.

These models also enable product manufacturers and or suppliers to create so called “product knowledge models” of their products in a way that product information can easily be integrated in a Systems Engineering environment based on the same information models.

Every element within a statement (role 1 element, the relationship and the role 2 element) must be traceable to a definition of that element. Therefore this part of ISO 15926 should be used in combination with the ISO 15926 Reference Data Library (RDL, ISO/TS 15926-4) or comparable one. Traceability to a definition can be reached by, every time a new entity is introduced in a statement, making a classification association between the new introduced entity with the corresponding class in the RDL.

In Figure 57 is a RDL structure given based on entities derived from ISO 15926-2 which can serve as a root-class structure for a RDL for Systems Engineering. The shown classes in this figure represent the central root classes used in these information models.

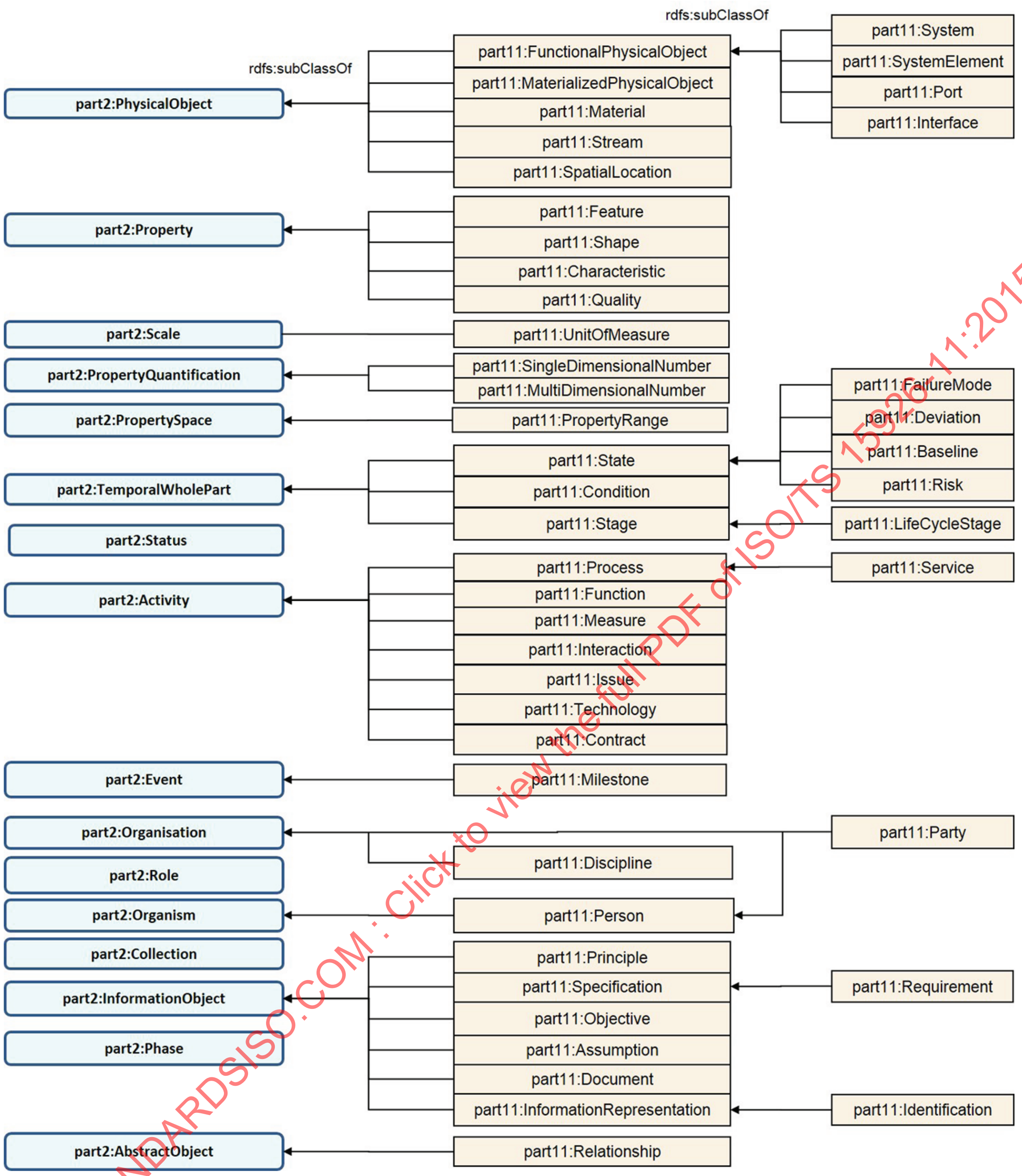

Figure 57 — Example of a root structure (taxonomy) for terms in the context of Systems Engineering as used in the information models in this clause

For each relation within the initial set of relationships is, based on the information models, defined what kind of thing is allowed at the left side of the relationship and what kind of thing is allowed at the right side of the relationship (see columns Domain and Range in the spreadsheet of Annex A).

6.2 Reference information models representing Systems Engineering

The reference information models in this part of ISO 15926 are set up using the principles described in chapter 4.1 Purpose and objectives. The area of Systems Engineering in this ISO part consist of the following information models:

— the process side and the physical side of a system (Figure 58);

— the breakdown structure of the physical part of a system (Figure 59);

— characterizing physical objects by means of properties and status (Figure 60);

— the interactions between system elements and between system elements and the environment and stakeholders (Figure 61);

— the failure mode and effect analysis of a system (Figure 62);

— requirement specification analysis of a system (Figure 63);

— verification of a requirement specification (Figure 64);

— the risk management information model (Figure 65);

— contract change management process (Figure 66);

— work breakdown structure of a system (Figure 67);

— communication flow between parties (Figure 68);

— project organizations (Figure 69);

— information model of a measure (Figure 70);

— information model of an assumption (Figure 71);

— connectivity of physical object (Figure 72);

— document and document versions (Figure 73).

Based on the proposed information models shown below, which must be seen as reference models, an initial set of relations is derived as presented in Annex A: Initial set of relationships.

Each relationship within this initial set can be found in one of the following information models and each statement gives the context and usage of that relationship. Each relationship with the left and right objects represents a statement in the context of Systems Engineering.

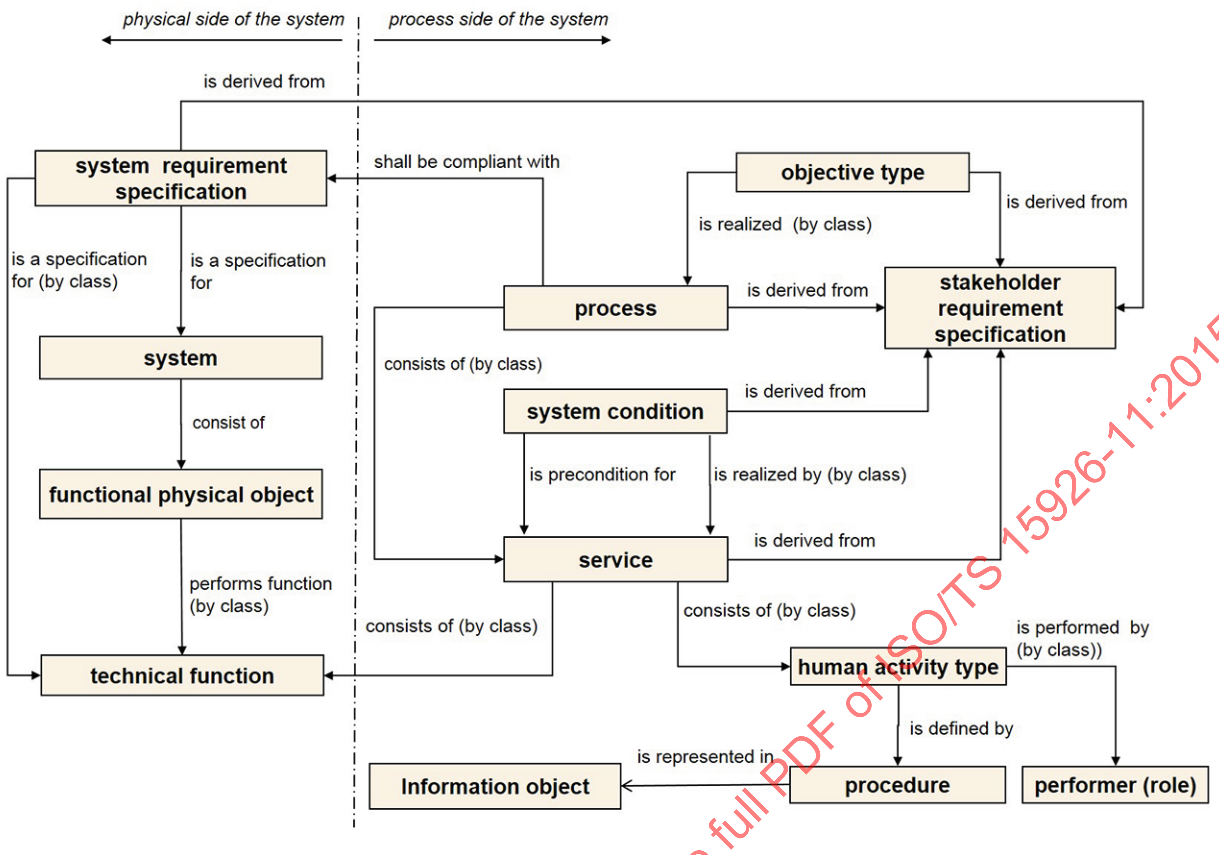

Figure 58 — Information model that represents the process side and the physical side of a system by defining relevant entities and the relationships between these entities concerning the system

Figure 58 describes the entities involved in developing a system. Development of a system in principle starts with stating objectives concerning the system by the stakeholders. There is at least one objective for the target system and one objective for the (enterprise) system which realizes the target system. After defining the objectives, the system life cycle processes “start working” to achieve these objectives.

EXAMPLE The operation process, the requirement definition process and architectural design process are examples of system life cycle processes.

The next step is to define services which must be realized to let the processes perform in such a way that the objectives will be achieved. Services on one hand realize process conditions and on the other hand they need process conditions (in fact pre-conditions).

A service can be realized by one or more technical functions of a functional physical object and/or a human activity. Functional physical objects have one or more technical functions as features which performs (a part of) a service. Services and technical function are both a subclass of activity.

By splitting up a service into technical functions and human activities, a decision is made on the level of automation of the service and separates the responsibility of the operator/maintainer and the designer/builder of the system in achieving the objectives.

Objectives, processes, process conditions and services are derived from stakeholder requirement specifications. Also system requirement specifications are derived from stakeholder requirements. In principle a system consists of functional physical objects which perform technical functions.

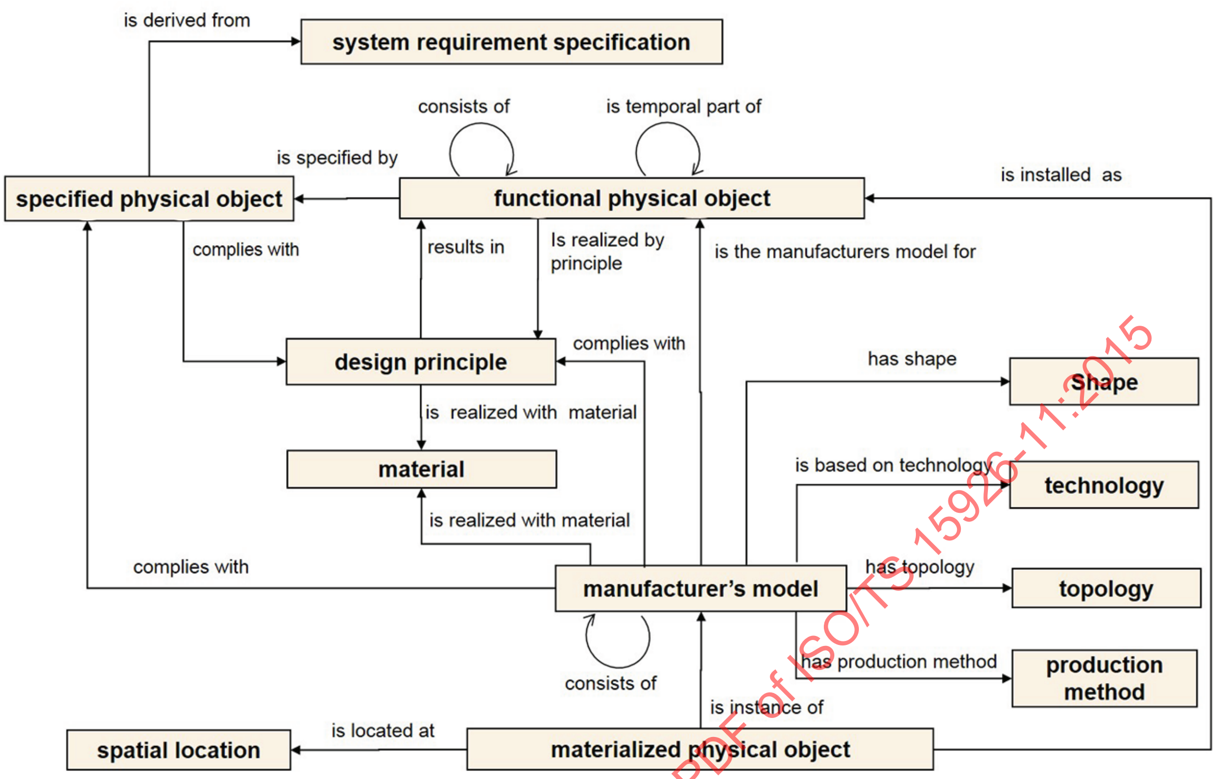

Figure 59 — Information model that represents the breakdown structure of a physical part of a system

Figure 59 shows the information model concerning the system breakdown structure of a system. Within this model four levels of realization of a physical object are recognized:

— functional physical object;

— specified physical object;

— manufacturer's model;

— materialized physical object.

A functional physical object can be defined on several abstractions levels, from sub-system (for example a monitoring system) up to equipment level (for example a pump) or materialized level. On subsystem level a functional physical object is in de breakdown structure followed by a design principle which leads to a set of new functional physical objects. On equipment level a functional physical object is in the breakdown structure followed by a chosen manufacturer's model. The manufacturer's model in fact is a chosen solution for a functional physical object which will be valid during a temporal part of the functional physical object and will have a unique instance during a temporal part of the life cycle of the system. By using the "begin of life" and "end of life" relationships, one can assign a materialized physical object in time to more than one functional physical object.

A manufacturer's model complies with a design principle and is based on a specific technology, has a specific production method and has a topology.

When assigning properties to the various abstraction levels of a physical object one can use the relationship "concerns stage" to assign a specific property to a specific stage since there is in general a relation between the level of abstraction of a physical object and its life cycle stage.

NOTE In the so called "hamburger model" (Functional Unit- Technical Solution model) the functional physical object and the specified physical object are combined in the Functional Unit. The hamburger model is described in the General AEC reference model (GARM), the integration model for AEC applications defined in the early days of STEP.

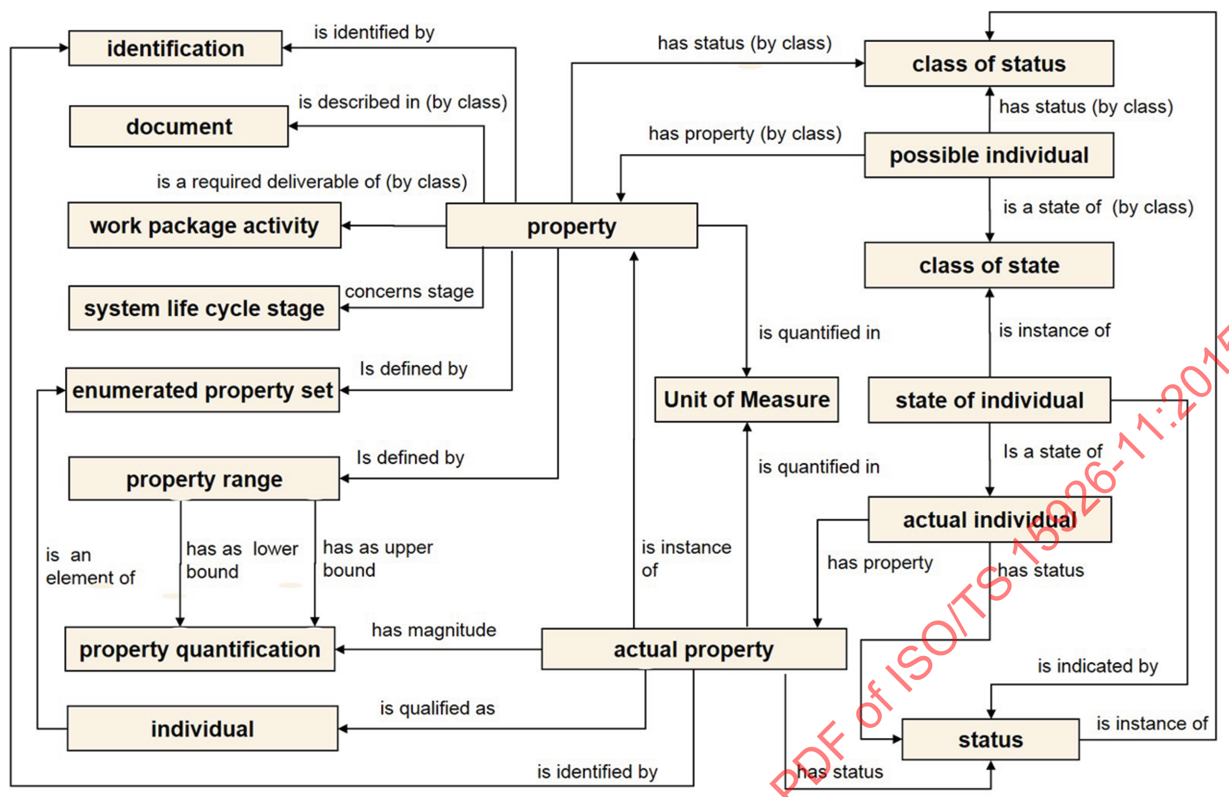

Figure 60 — Information model that represents a simplified usage of properties and status within this part with respect to the data model defined in part 2

Figure 60 shows the information model concerning properties of any element ("possible individual") of a system with a specific approach for unique instances of classes (a whole life individual in a particular state). As long an element of a system (can be a for example physical object, activity or event) exist on class level or on specification level a property is accompanied with a property range defined by an upper and lower boundary and the unit of measure in which the property is expressed. Once a system element is realized and is in a specific state, an actual property has a specific magnitude and is expressed in a specific Unit of Measure.

A property is not always expressed in a unit of measure but can be expressed in a string value being an individual. In that case the property can be qualified by a specific name were, by means of a enumeration were several string values can be defined in an ordered manner, each with a given name as possible value for that property (called the qualification of the property for example IP 44, IP 55 and IP 66 as allowed IP rates in in the context of a property "Protection Rating").

More or less on the same manner the possible states of system element can be defined by defining a class of possible values of the status. On instance level of a system element the state of an individual a particular status can be identified with a name (for example "red" or "green" in case of a colour).

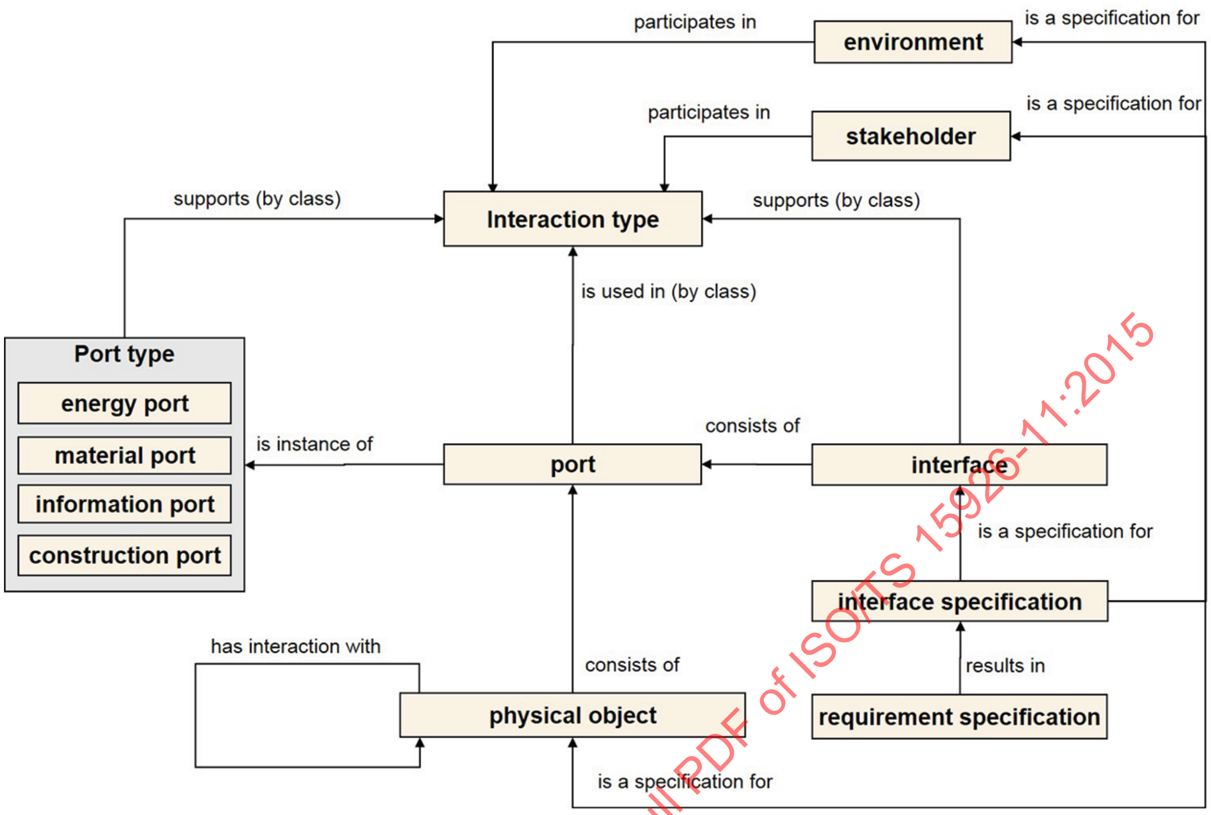

Figure 61 — Information model that represents the interactions between internal system elements and interactions between the system elements and the environment and or stakeholders

Functional physical objects interact with their environment by means of an interface. The interface of a functional physical object exists of ports as shown in Figure 61. There are 4 basic types of ports: material ports, energy ports, information ports and construction (3D) ports. Interactions can be occurs between functional physical objects and it can be between functional physical objects and the environment and or stakeholders.

Based on the models presented in Figure 59, 60, and 61 manufacturers can set up their own product knowledge model that can be used and integrated in the systems engineering process.

Once a functional physical object is realized by a materialized physical object, the ports defined on functional physical object level will also be materialized and becomes specific physical connections.

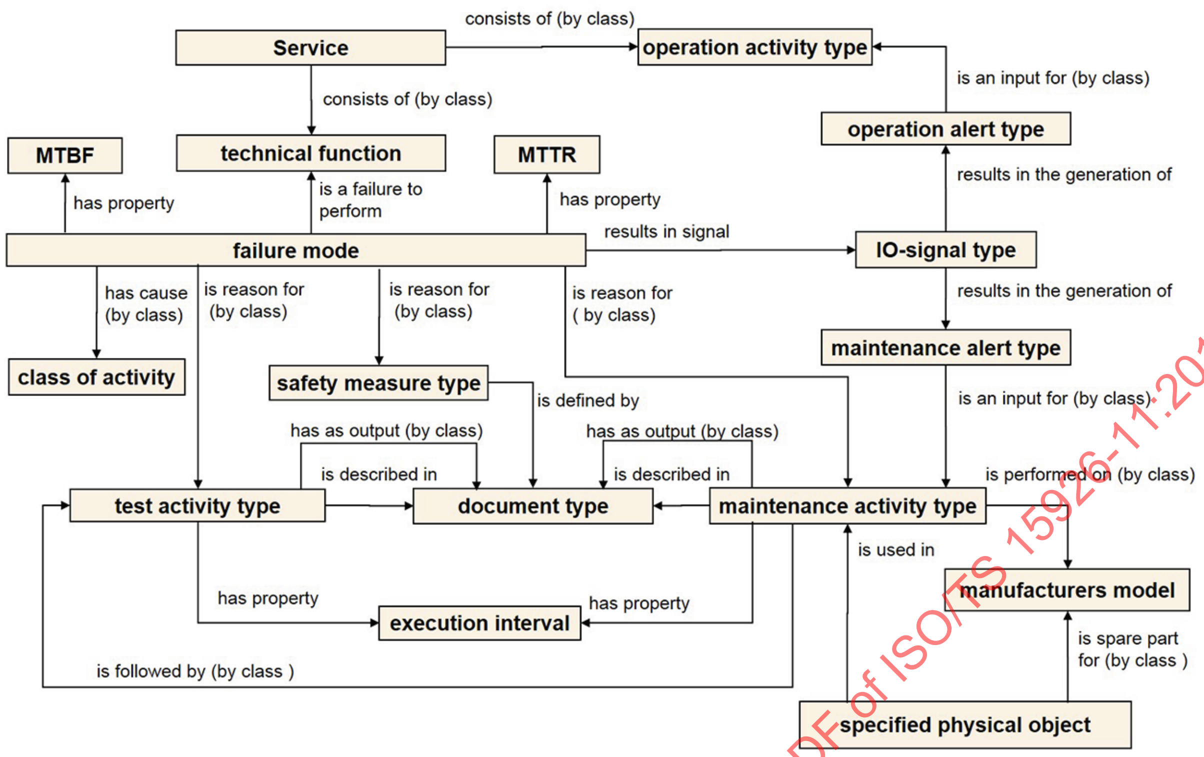

Figure 62 — Information model that represents the failure mode and effect analysis of a system by defining relevant entities and the relationships between these entities

Figure 62 shows the information model concerning failure modes of a system. This model is based on determining possible malfunction of a technical function performed by a functional physical object. A failure mode is a reason for defining one or more maintenance activities, test activities and or a safety measures which are defined in a procedure, contained in a document.

Failure modes that are detected by a control system signal results in a maintenance alert and or an operator alert which will be input in a resp. maintenance or operator activity.

To execute a maintenance activity in relation to a manufacturer’s model, it can be necessarily to use a spare part which will be also a manufacturer’s model on itself.

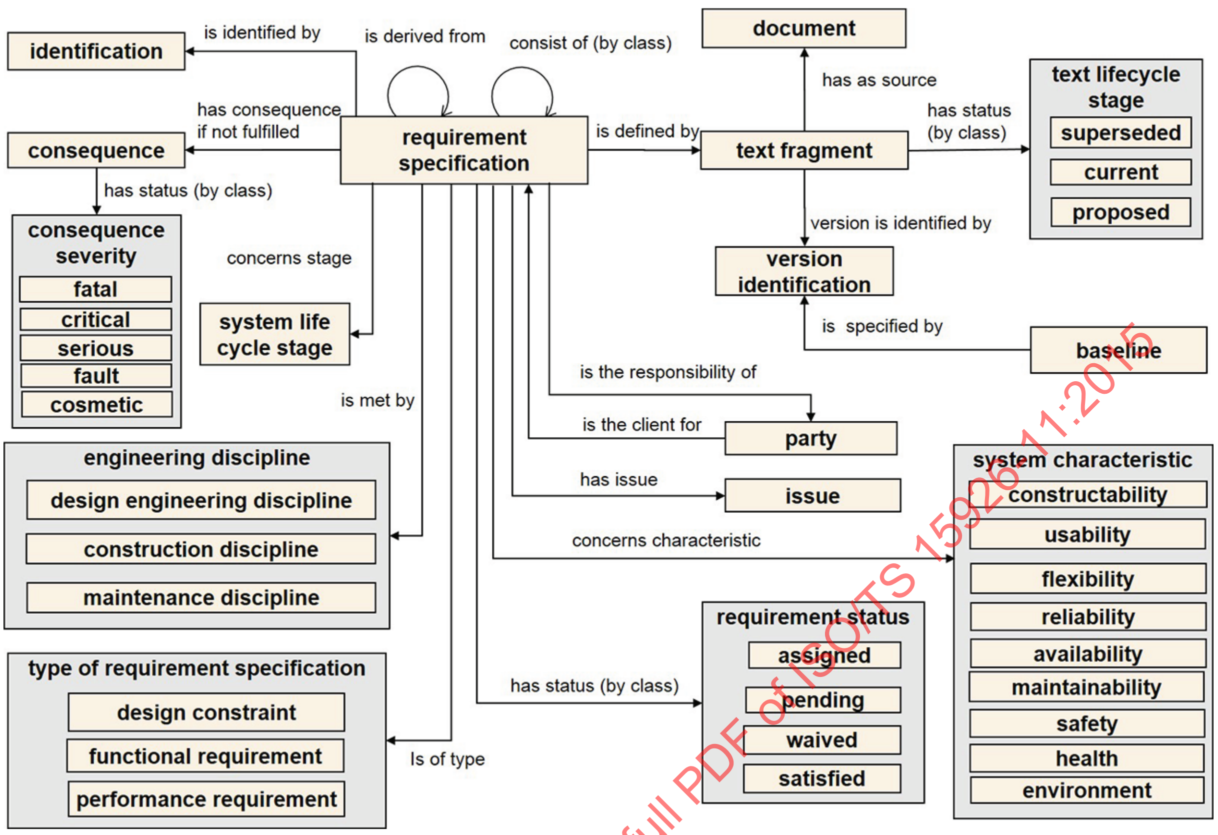

Figure 63 — Information model that represents the requirement analysis of a system by defining relevant entities and the relationships between these entities

Within the requirement analysis (Figure 63) process, requirement specifications are classified in terms of severity, engineering discipline, type of requirement and which system characteristic is addressed by the requirement specification. The requirement specification is allocated to a party which functions as “client”.

In general a requirement specification will be a design constraint, functional requirement or a performance requirement.

A requirement specification is defined in a piece of text “text fragment” that has a status in the context of its lifecycle, has a version and has a source, being a contract or other document.

A baseline can be specified by one or more version elements that are used to identify the version of text fragments.

A requirement specification itself also has a status which can be for example assigned to a system element, pending in order to be assigned, waived when the requirement is no longer relevant, and satisfied in terms of “taking count of” or fulfilled.

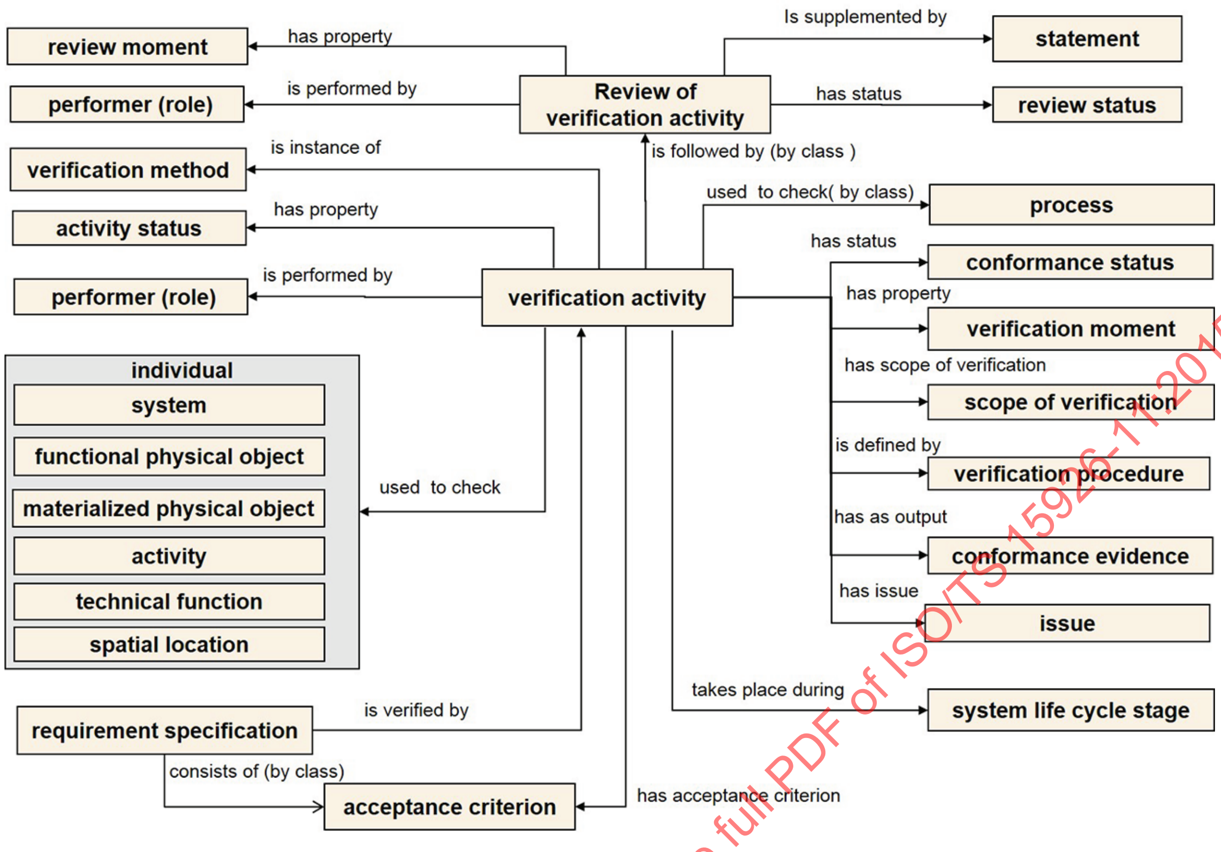

Figure 64 — Information model that represents the verification of a requirement specification by defining relevant entities and the relationships between these entities

The fulfilment of a requirement specification must be demonstrated to the client by a contractor in order to prove that the product or system will be or is built in the right way. Proving that a requirement specification is fulfilled is called verification. In principle each requirement specification (Figure 64) has one or more related verification activities, depending in which system life cycle stages a demonstration is required. One requirement for example can be verified in the design stage and the construction stage, another requirement needs verification just during the maintenance stage of the system.

Each verification activity is related to the specific item that is checked. This can be an individual, specific system, object or activity. Also a process (which is a class) can be verified.

This means if for example a specific requirement specification related to a specific system is checked in tree lifecycle stages of that specific system, there exist three verification activities, each can have their own method (for example review, calculation, simulation, inspection), scope (for example full scale, sample, typical) and procedure (test protocol).

For each verification activity it might be needed to specify an acceptance criterion due to the vague character of the requirement specification. In fact the specified acceptance criterion is in that case an extension of the requirement specification and becomes a part of it.

The verification activity can be depending of the conformance status, a source for a contract deviation for example when the requirement specification points out to appear not feasible.

A specific verification activity has always a status (for example active, finished) and will be performed by a role (for example design manager) and /or a specific person.

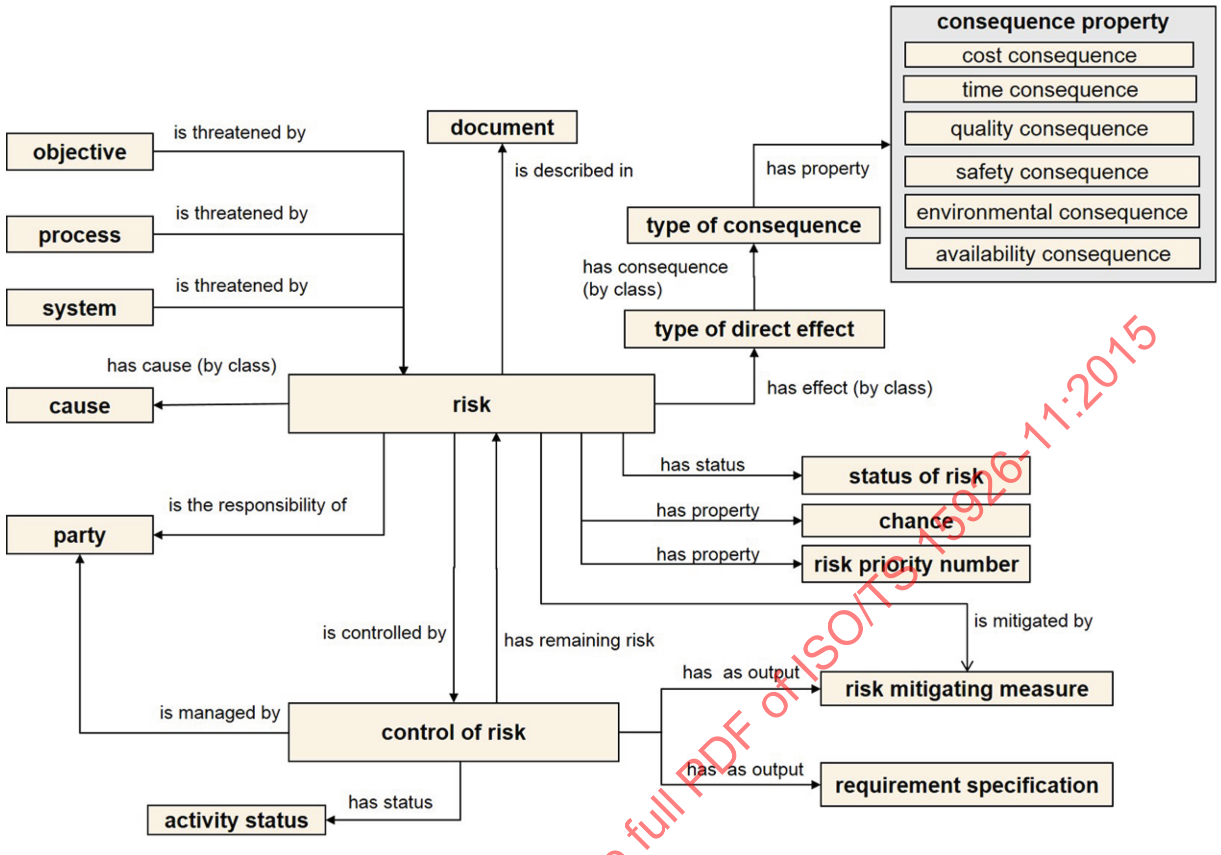

Figure 65 — Information model that represents the information around the risk management process by defining relevant entities and the relationships between these entities

Risk management requires control of information that characterizes a risk and information that comes with mitigating a specific risk.

A risk in principle threatens an objective and the objective related system and or process (Figure 65). A risk has a cause and effect and is de responsibility of a person or organization. The effect leads to consequences concerning cost, time, quality, safety, environment and or availability of the system. By rating these consequences and multiply the mathematical sum of them with the chance there arises the risk priority number which can be used to prioritize a set of risks.

Controlling the risk is managed by a person of organization (both subclasses of party) and ends up in one or more mitigating measures and of one or more requirement specifications. Taking account of fulfilling of these measures and requirement specifications there exist a remaining risk with a new chance and set of consequences resulting in a new risk priority number which must be significant lower than the risk priority number of the primary risk without the measures and or requirements.

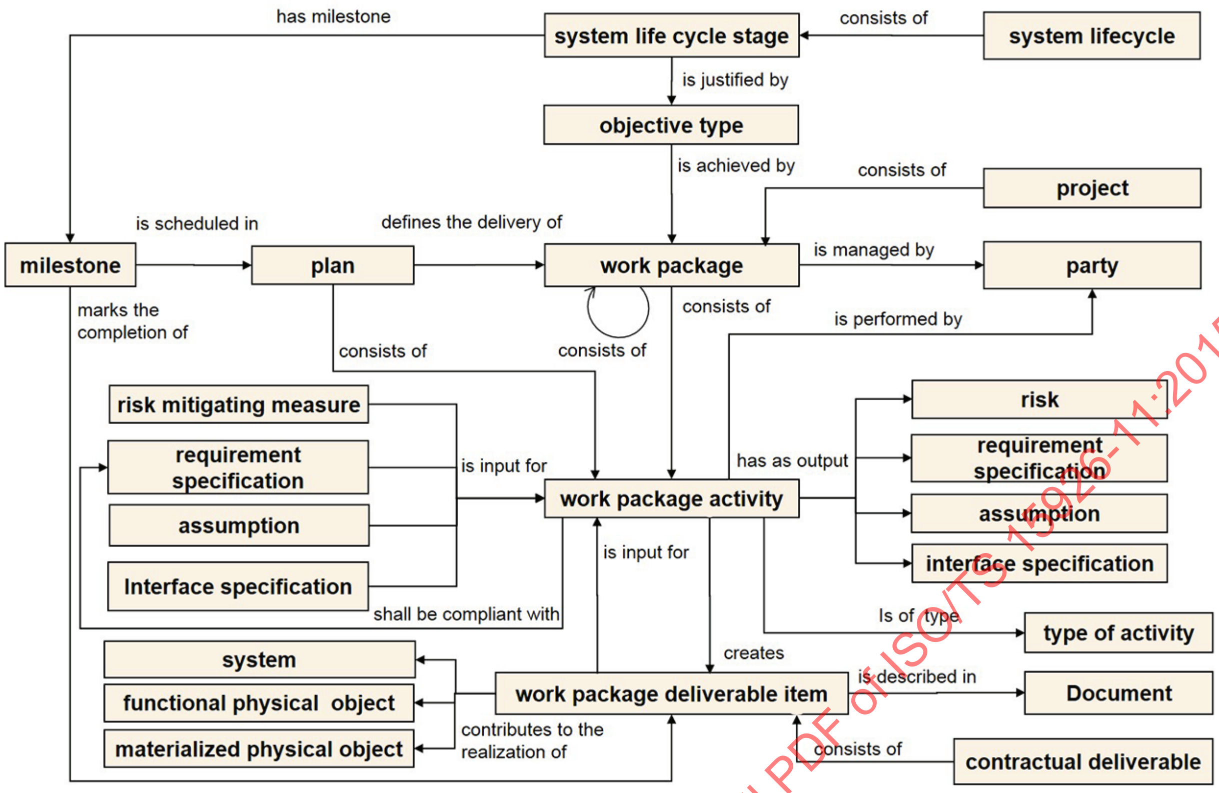

Figure 66 — Information model that represents the contract change management process of a system by defining relevant entities and the relationships between these entities

Figure 66 represents the process of contract change management. A central entity is a contractual delivery which is stated by a principal (for example a client) to be delivered in the contract with the contractor. The contract is between a principal and a contractor where sometimes a third party is involved (stakeholder, preferred supplier, etc.).

A contractual delivery must be compliant with one or more requirement specifications (the "what") and consist of work package deliverable items (see Figure 67). The whole set of work package deliverable items should satisfies the set of contractual deliverables.

In order to be able to phase the work, several marks will be defined where a part of the contractual deliverables will be accepted and some of them will lead to payment. These financial milestones are specific milestones as intended in Figure 67.

An issue around a contractual deliverable can be, after clarifying, a reason for a contract change proposals described by a document that can be ending in a contract extension defined in a document.

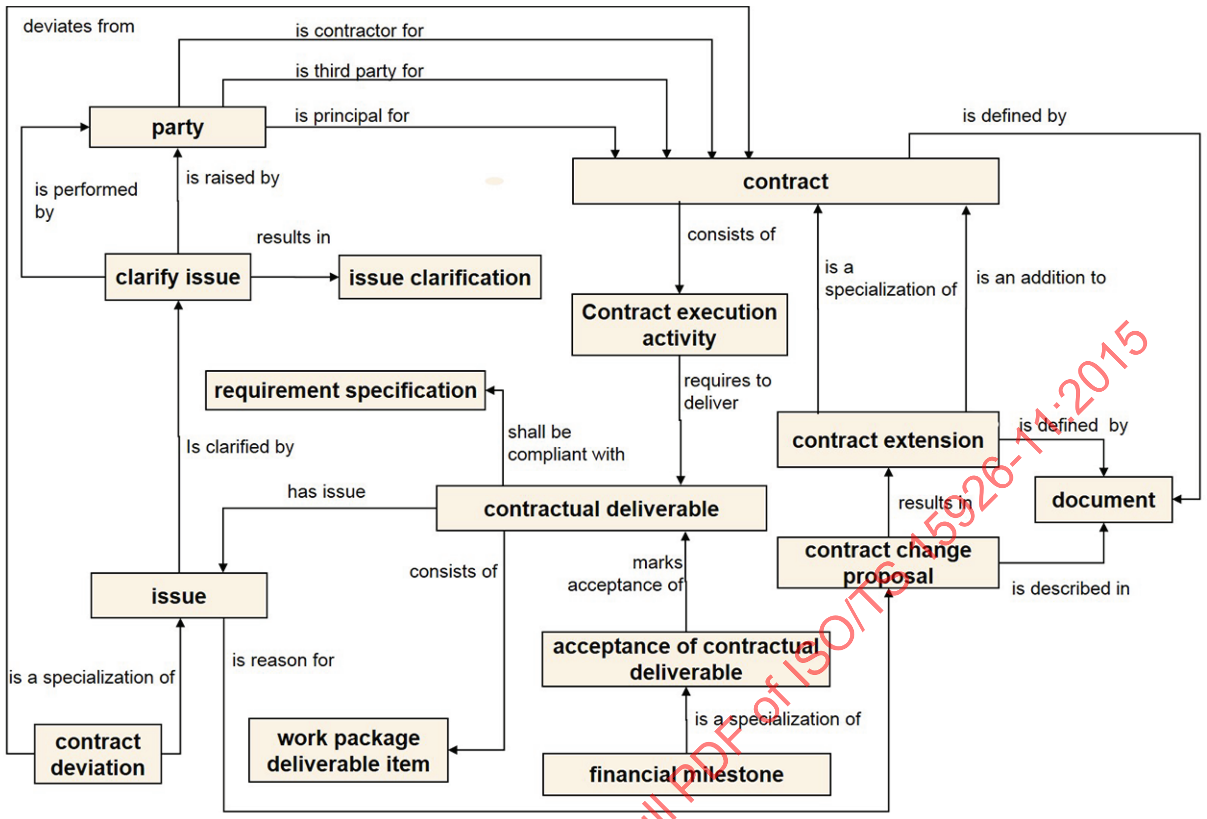

Figure 67 — Information model that represents the work break down structure of a system by defining relevant entities and the relationships between these entities

Figure 67 shows the information model of the work breakdown structure of a system, capturing the whole system life cycle which is typical divided into several stages. Each stage is justified by one or more objectives ("what must have been achieved when the stage is finished"). These objectives are achieved by realizing one or more work packages. A work package exist of one or more activities creating a deliverable contributing to a subsystem, a specific functional physical object or materialized physical object. Both life cycle stages and work packages can have milestones, each of them scheduled and described in a planning.

To execute a work package activity that create a work package deliverable, there will be input needed from the risk management process (measures), requirement analyses process (requirements), design process (assumptions and port interactions specifications. The activity will also lead to new risks, requirements, assumptions and port interactions.

Every work package activity will be in principle an instance of an activity in the context of one of the ISO/IEC 15288 processes

An important entity within SE is a decision. In the presented models decisions are represented by design principles (Figure 59), measures (Figure 70) and assumptions (Figure 71): they all in essence represent decisions. They all have a relation with work package activities and therefore with milestones, so major decisions can be defined by introducing specific milestones.

Trade-off studies including the final design principles can be described in documents that are related to the relevant work packages and Technical Solutions.

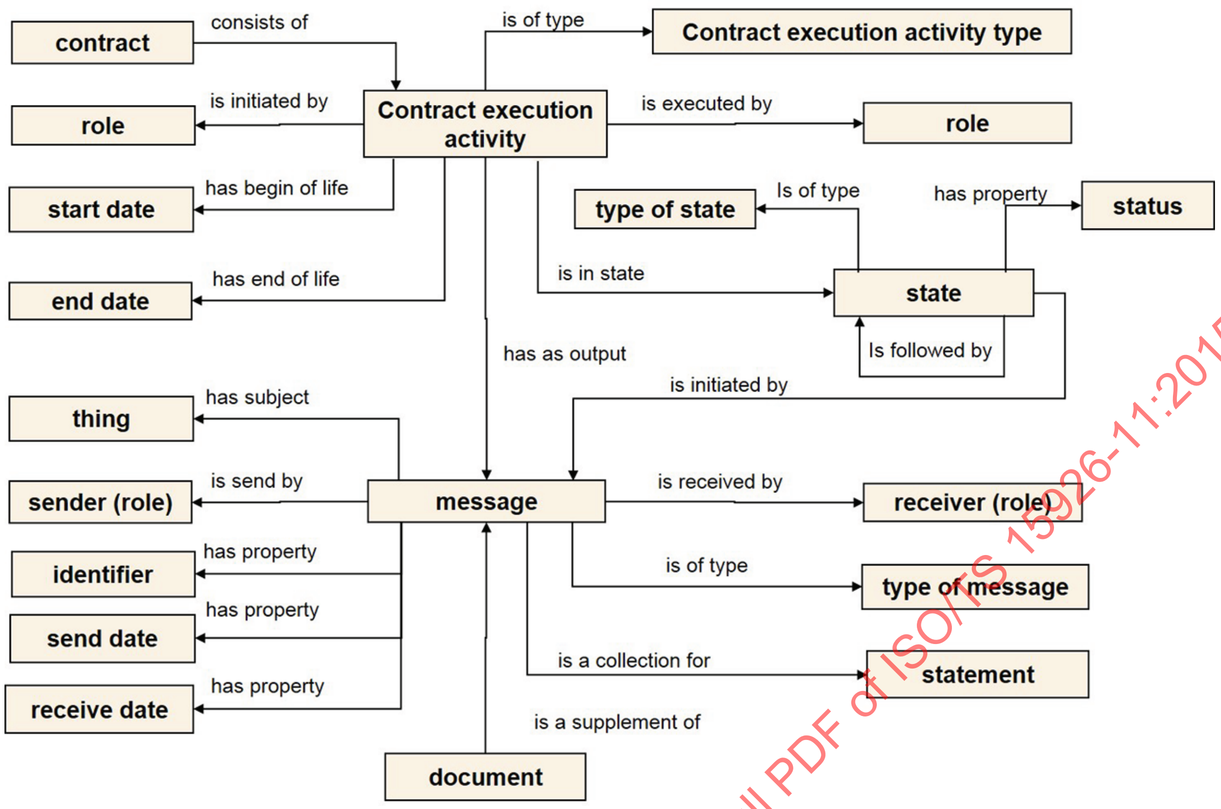

Figure 68 — Information model that represents the communication flow between parties using messages in the context of a specific contract execution activity

Figure 68 represents the formal communication flow between parties involved in a project. Formal communication is defined here as communications in the context of a contract between parties.

A contract consists of one or more contract execution activities of a certain type. A contract execution activity is both initiated and executed by a role of an organization or the role of a person. A contract execution activity will always be in exactly one state and can be of type requested, promised, delivered or accepted. Type of states follows each other in a predefined order. A change in the state of a contract execution activity can only be initiated by means of a formal message as an output of this activity.

A message is of a certain type and is send and received by a role of a party and shall have a “thing” as subject and can be supplemented by a document.

Also a message can consist of a set of statements as defined within this part of ISO 15926.

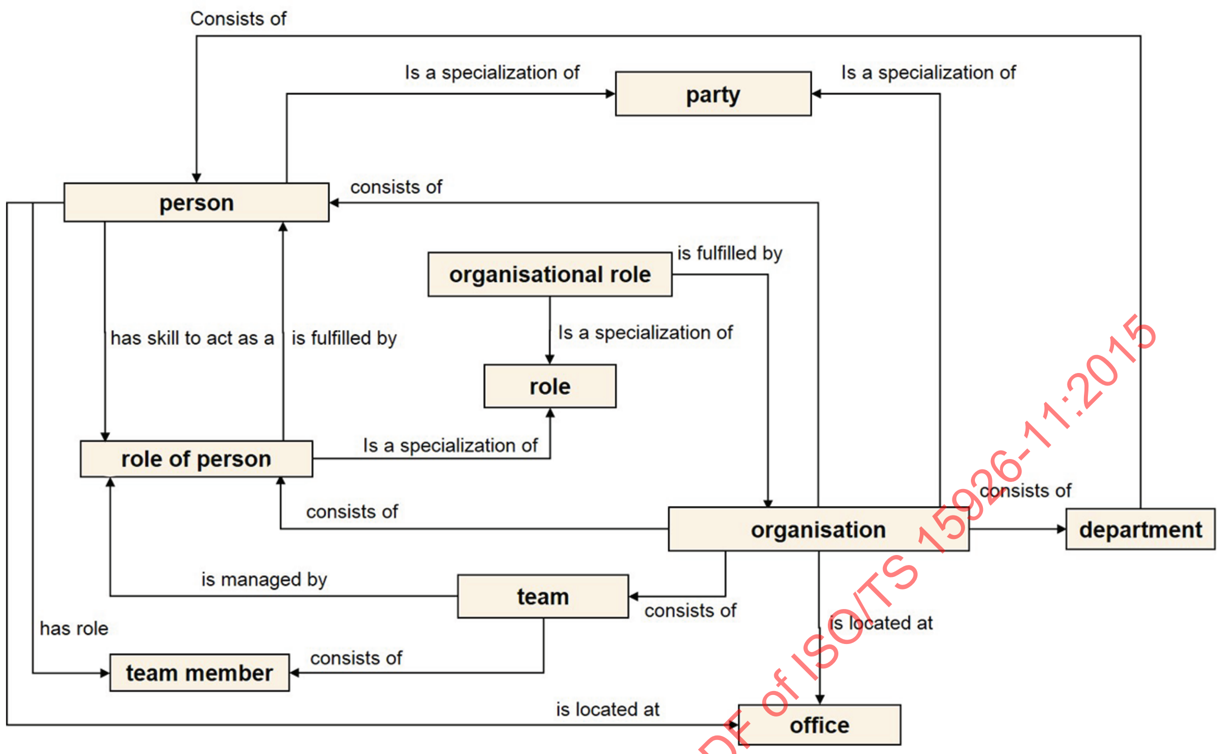

Figure 69 — Information model that represents the organizational terms and how they can be related to each other

Figure 69 represents entities in the context of organization realizing projects. In the other models several times the term party is used were a party can be an organization or a person.

Within this model there has been made a difference between roles of organizations and role of persons.

An organization can consist of persons, role of persons, teams and departments. Teams are managed by a role of a person and consist of several team members each being a role of a person.

Organizations and persons are located in an office.

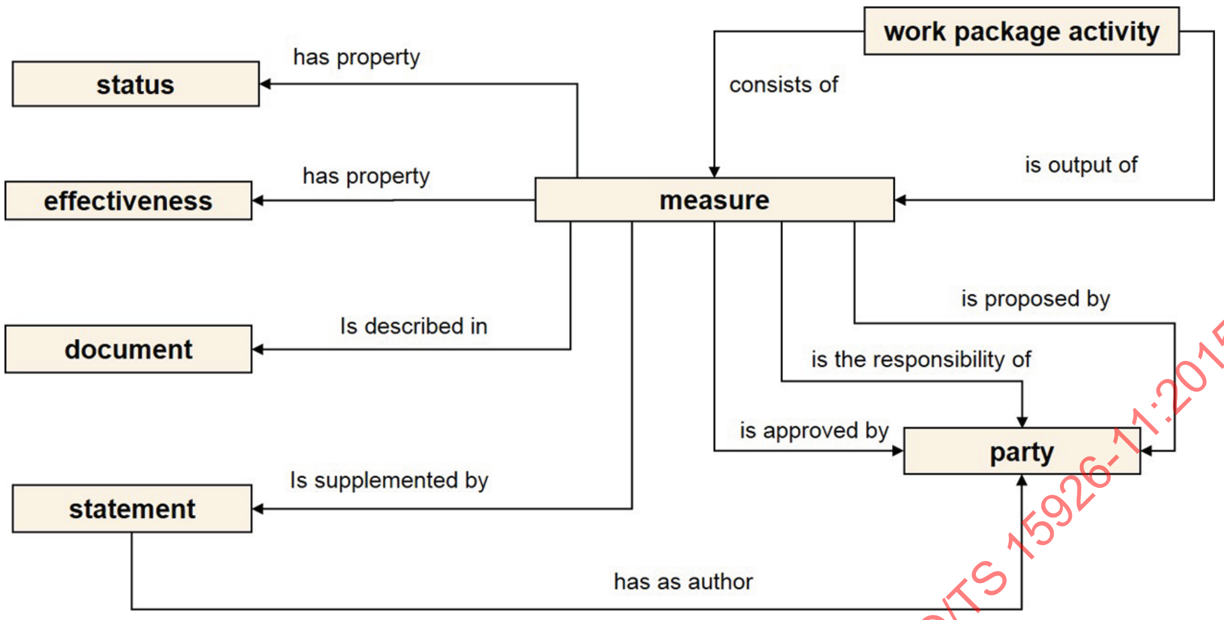

Figure 70 — Information model that represents the information model of a measure (subclass of activity) created and used in the other information models

A type of measure is a 'risk mitigating measure' as introduced in Figure 65. A measure is characterized by a specific workflow: it will be proposed by the party that is managing the controlling activity of the risk. The measure will be approved by those parties that will be affected by the measure. The measure will get a status for example 'approved' and there will be a party that is responsible for executing the measure.

There will be a specific work package activity that will really execute (perform) the measure. When the measure is successfully executed the status of the measure for example to 'executed'. In that case in principle there must be a document that proves the execution of the measure.

During the workflow statements can be made by the various parties to explain the course of the measure. A measure can be classified by a property that represents the effectiveness.

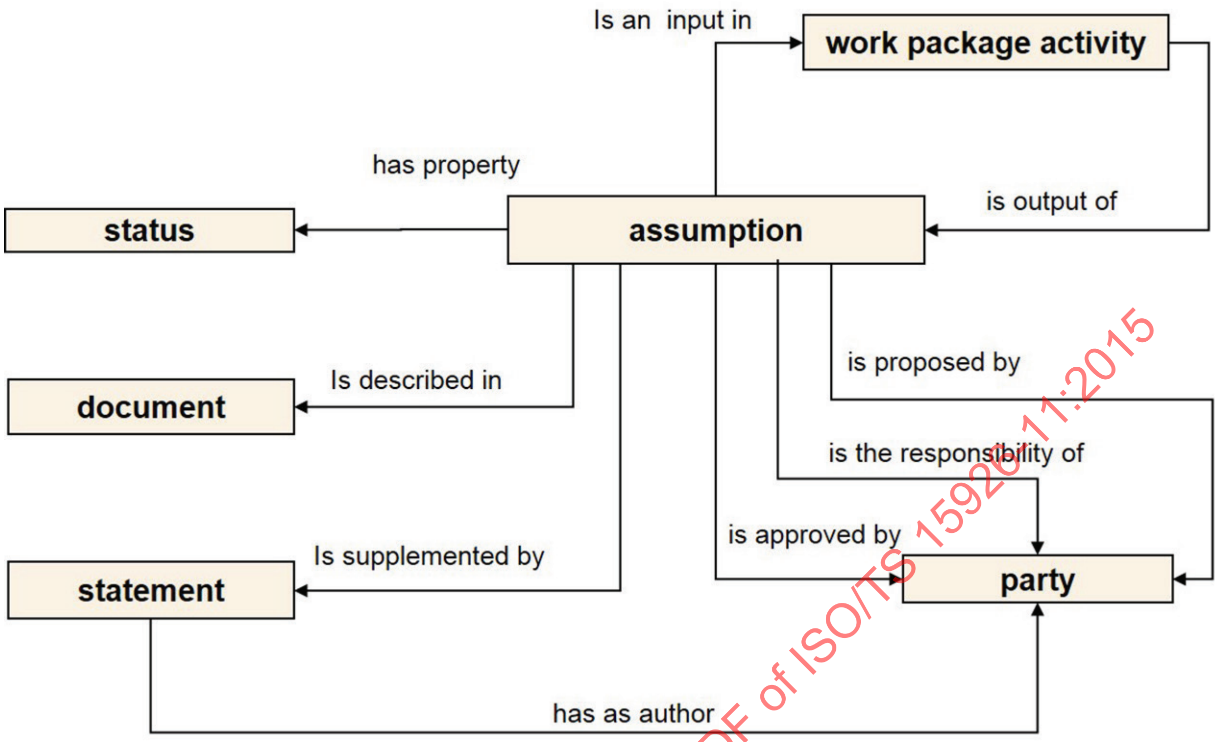

Figure 71 — Information model that represents the information model of an assumption (subclass of information object) created and used in the other information models

The entity assumption is introduced in Figure 67. In Figure 71 the information model is presented of an assumption. An assumption can be used as a registration of for example a design-decision or interpretation of a requirement. An assumption is like a measure also characterized by a specific workflow: it will be proposed by the party that is executing a work package activity for example a design activity. The assumption will be approved by those parties that will be affected by the assumption. The assumptions than will get a status for example 'approved' and there will be another work package activity that will have the assumption as input. When the assumption is successfully processed the status of the activity changes for example to 'executed'. In that case in principle there must be a document that proves the execution of the assumption. During the workflow statements can be made by the various parties to explain the course of the assumption. Assumption can for example be used to clarify requirement specification as output of a requirement analysis activity (a specific work package activity) and to describe a system design of human activity.

Despite of the fact that the information models of both a measurement and assumption looks like the same, they represent each another concept, meaning and role within projects. For this reason both were not combined in one model.

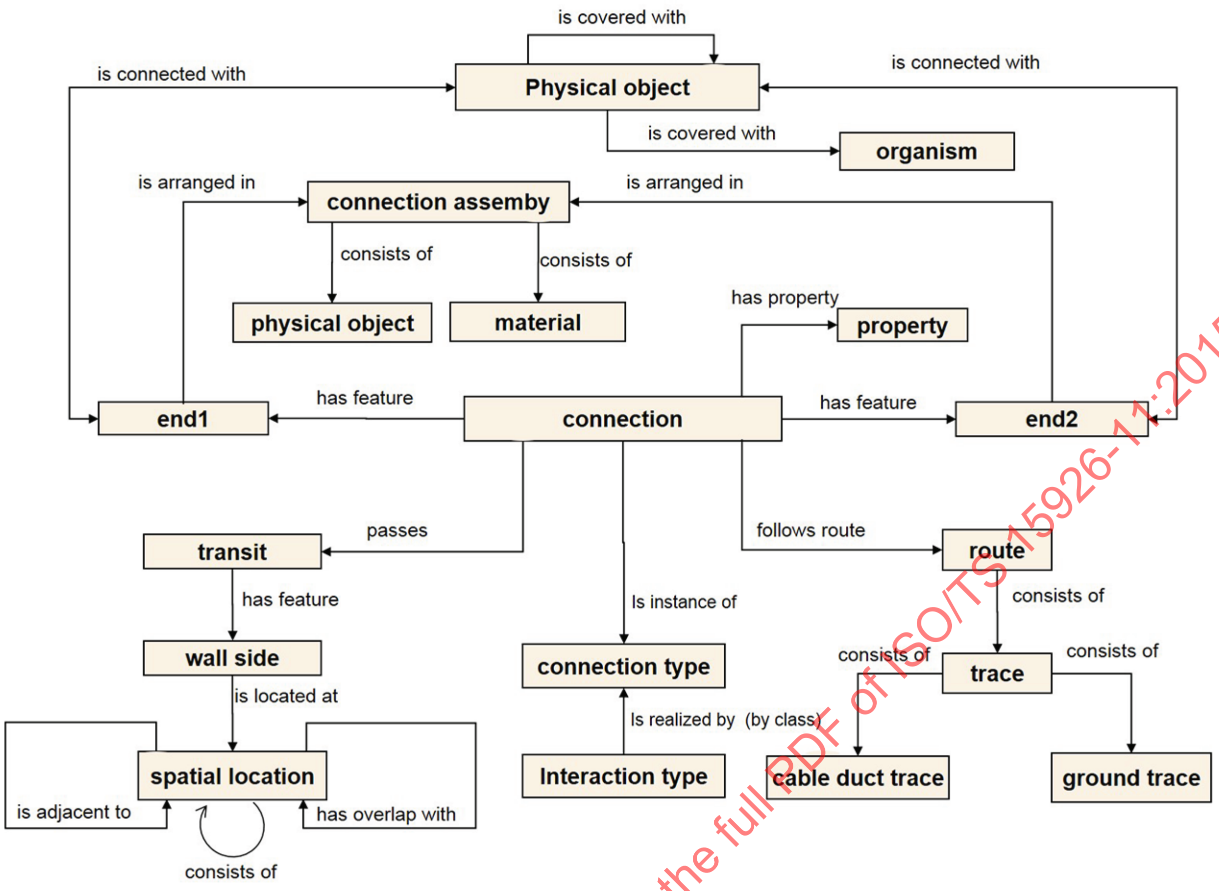

Figure 72 — Information model that represents the connectivity of physical object for example by means of cables and pipes

Figure 72 represents the information concerning connectivity within and between materialized physical objects as parts of a system. This model can handle pipes and cables (the connection type) as a realization of connections. A connection has on both ends a feature called end1 and end2. The needed connection material is represented by a connection assembly consisting of material and physical objects like bolds and nuts.

A connection can have one or more properties specifying the capacity, pressure, diameter, etc. of the connection. A connection can pass a wall that separates two spatial locations by means of a transit of a certain type. A connection follows a route that consists of one or more traces were a trace can consist of ground trace and or duct traces.

Spatial location can also be connected to and even overlap each other what can be made explicit by means of the relationships “is adjacent to” respectively “has overlap with”.

Another form of connectivity is the covering of a physical object with another physical object or organism.

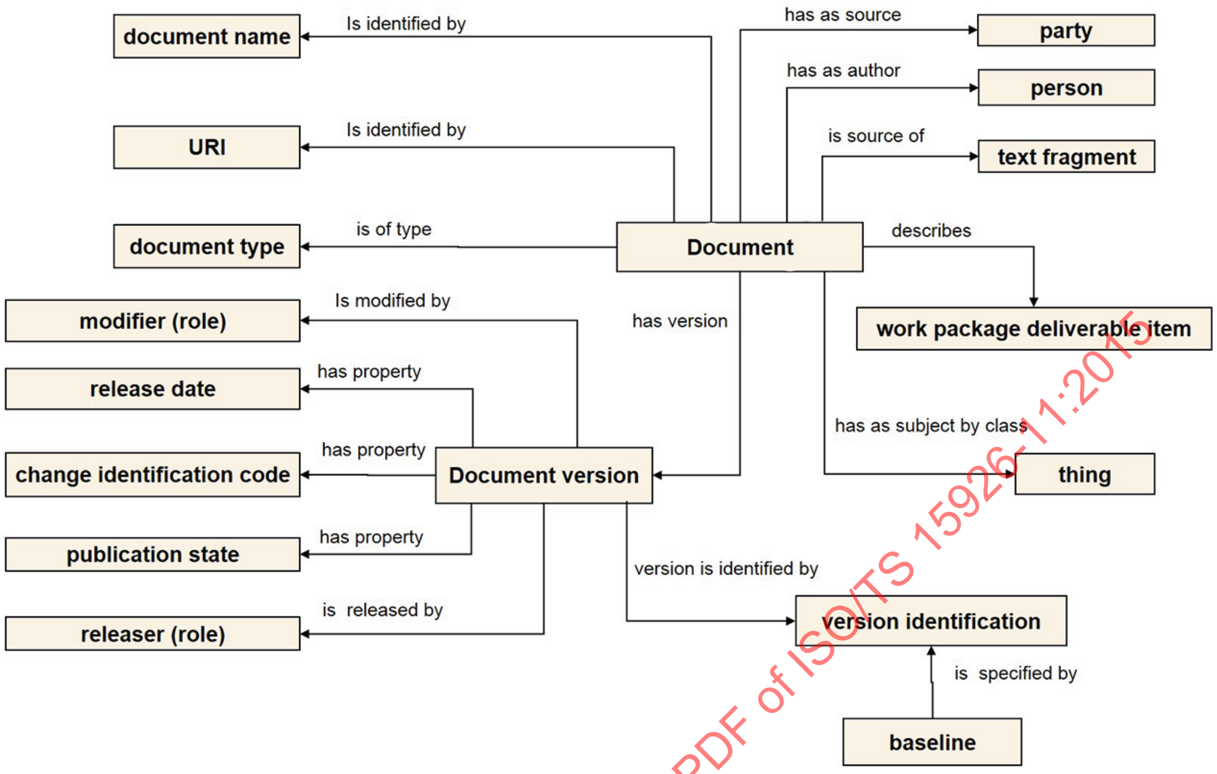

Figure 73 — Information model that represents relevant information about a document

Figure 73 represents the information concerning a document and versions of the document.

The document element has metadata and relationships with sustainable elements such as author, source and subject and document type.

A document in general describes a work package deliverable item such as a system or equipment (see Figure 67).

The main document element has one or more versions which have metadata such as release date, publication state, and releaser and change identification code.

A baseline can be defined by a version identification identifying specific versions of documents.

7 Initial set of reference relationships

To produce for example product models that are compliant to this part of ISO 15926 only relationships shall be used that are within the initial set as given in Annex A or traceable (private) specializations of them.

This initial set is derived from the (reference) information models shown in Clause 6, Figure 58 up to and including Figure 73. The relationships are connected to entities as defined in ISO 15926-2 and based on the principles described in Annex D and Reference [18].

The definition which is given in Annex A originates from the Systems Engineering reference models in Clause 6. Tailoring of the definition of a relationship is allowed as long it stays consistent with the defined domain and range in Annex A. A project specific tailoring would then be subject of the Information delivery manual (IDM) of that project.

In practice the user will start from an object, classified by a (sub) class in the RDL and then find the relationships that are possible for that object in accordance with column domain and range.