

PRoMo

Property Meta Model

Basic Concepts for openAAS

Epple, Palm, Azarmipour
White Paper
Version 0.9
10/10/2016

Lehrstuhl für Prozessleittechnik
Prof. Dr.-Ing. Ulrich Epple
RWTH Aachen
D-52064 Aachen, Deutschland
Telefon +49 (0) 241 80 94339
Fax +49 (0) 241 80 92238
www.plt.rwth-aachen.de

Contents

1	Introduction	1
2	The Property Meta Model	3
3	Properties and their Carriers	4
3.1	Material and Immaterial Carriers	4
3.2	Identification of Carriers and Properties	4
4	General Properties	6
4.1	Existence as a property by itself	7
4.2	Semantic Specification Model	7
4.3	Unified Specification Schema	8
4.4	Unified Identification Schema	8
5	Property Value Statements	10
5.1	Unified Expression Logic	11
5.2	Unified Expression Semantic	12
5.3	Type Assigned Statements	13
5.4	Formal Reasoning	13
6	Administration of property value statements	15
7	Summary	16
	Literatur	17

1 Introduction

All systems, including concrete living entities or non-living objects, such as living organisms, products, devices, plants or abstract objects such as algorithms, methods, processes or relations, possess properties by which they are described. Every property is an intrinsic part of its thing. We say: the thing carries the property. As shown in Fig.1.1, properties can be specialized into

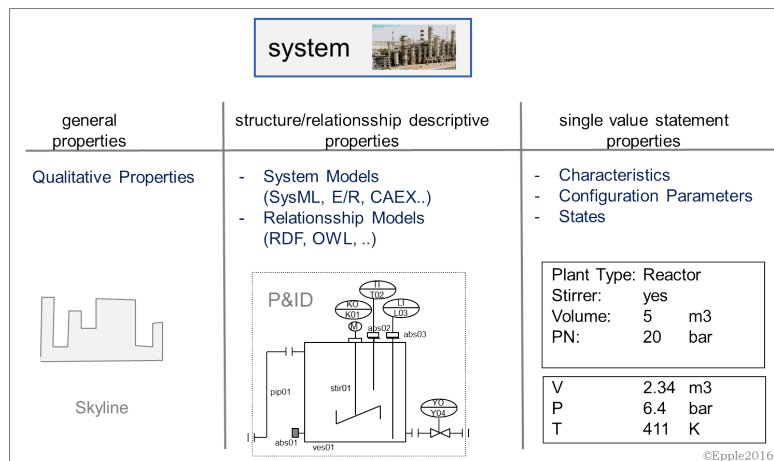


Figure 1.1: Property Categories

different kinds of formalized types. Structure/relationship descriptive properties allow a formalized description of structures and relationships based on a system meta model. Single-value statement properties allow a formalized and standardized description of characteristics, configuration parameters and states by a single quantitative value. There is a usual distinction between states and characteristics due to the changeability of their values: The value of a state can change dynamically within the observation period. The value of a characteristic is permanently fixed or at least not changed within a defined observation period and therefore can be used to characterize its carrier.

Systems exchange information by exchanging data via communication. To be able to interpret the exchanged data, the involved systems need a common semantic interpretation concept. There is a plurality of semantic concepts which can be applied. Which fits best depends on the exchange purpose and the kind and grade of formal standardization of the fact description in the involved systems. As shown in Fig.1.1, properties play an important role in the exchange semantic. To describe the structure of a system, models like SysML, CAEX, E/R, etc. can be used as a common conceptual base. To exchange information about general relationships between semantic elements, concepts like Linked Data or OWL-constructs can be applied. In the context of industrial IT-systems in many cases only information about single valued properties

like characteristics, configuration parameters and states have to be exchanged. For all these cases semi semantic exchange concepts can be developed which allow the exchange and usage of information without an "understanding" of the specific individual semantic.

This paper focuses on the exchange of information about Single Valued Properties (SVPs). A general semi semantics will be presented which allows a common standardized exchange of and reasoning about statements to SVPs.

The paper is based on the publications [1] and [2].

2 The Property Meta Model

As shown in Fig.2.1, the property meta model consists of the thing by itself and three model parts:

- the carrier classification model
- the property type description model and the
- property value statement model

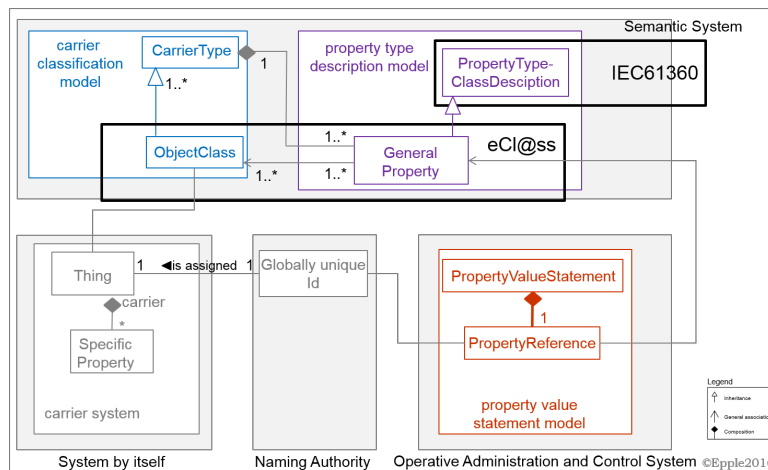


Figure 2.1: Architecture of the Property Meta Model

For the information exchange and usage within the operative system, only the property statement model and the possibility to reference the properties by a unique identification schema are needed. The property type description model allows the description of the meaning of a property type as a general term by standard description patterns. The semantics of a property type becomes clear only within the context of a specific carrier type. The property itself is a part of its carrier. The carrier is the 'thing by itself' and outside the model.

3 Properties and their Carriers

Every property is exclusively assigned to a specific subject matter, it is so called *property carrier*. We say: the subject matter has a property. A property is a singular term and exists exactly once. To focus on this fact, a property is denominated as a *special property*. At every point of time, every *special property* has a defined value.

3.1 Material and Immaterial Carriers

A subject matter can be a material as well as an immaterial thing. In both cases it can have properties and play the role of a *property carrier*. Material things have physical properties which belong to them intrinsically. Intrinsically means, that the physical systems has the properties independent from any modelling. For example, mycar (as a material thing) has its horsepower and its weight with actual physical values. Information about these actual values can be gained by measurements, model related calculations, from construction documents, or on the base of manufacturer specifications. But the physical values exist, independent of the fact if we have any information about them or not. Besides the material things, immaterial things can carry properties too. Immaterial carriers belong to the information world. They represent for example general terms or model elements like instances, classes, types and so on. Subject matters like type declarations or instance specifications contain statements concerning properties of their related physical objects. Of course they have their own properties too. The statement concept will be explained later, here just an example to explain the relationships: The financial administration has an information model in which all registered vehicles are recorded. Every registered vehicle is represented by an instance in this model. Every vehicle is associated with a tax number. This is a characteristic that has nothing to do with the physical vehicle but with its instance in the information system of the financial administration. In this case, the instance is the property carrier. By deregistering or selling the vehicle, this instance will be deleted with all its associated properties. However, the physical vehicle with its physical properties will exist further without any change. In contrary to the physical properties, the properties of the immaterial carriers are known as element of the information world by principle. Within the instance description of the administration, we also find notes about the weight and the horse power of the related physical vehicle. These notes implement *statements* about special properties of the physical vehicle but not properties by themselves. The concept of the *property statements* will be explained later.

3.2 Identification of Carriers and Properties

To exchange information about a property, it is necessary that every partner is able to identify the property. This requires an identification schema which allows the unambiguous identification of

a property within the whole system and an efficient way to transfer the necessary identification data between the communication partners. The identification of a property is realized in two steps:

- the identification of the property carrier and
- the local identification of the property within the namespace of the carrier.

An unambiguous identification of all property carriers requires a global unique identification schema for all kind of matters under consideration. Such an identification system is a general requirement. Especially in the technical and the commercial field, there exist *URIs* (Uniform Ressource Identifiers) to identify the objects of interest. Hierarchical schemas to order and identify technical objects like the elements of rolesystems (in buildings, factories, plants, machines..), produced products or even immaterial assets like plans etc. are well established and can be used. It seems feasible to find an appropriate solution for the unique identification of the property carriers. The main problem is the second part, the unique naming of the property within the scope of the carrier. The local name of a property is determined together with its general property definition. It is easy to organize unique names within the scope of a carrier type. The problem is the multiple inheritance schema. As we will see, the property set of a carrier comprises multiple general property sets from different carrier types. To overcome this problem, additional naming concepts are needed. In the next chapters some possible solutions will be discussed.

4 General Properties

The meaning of a property will be described in a description model. The concept of this model is explained in Fig.4.1.

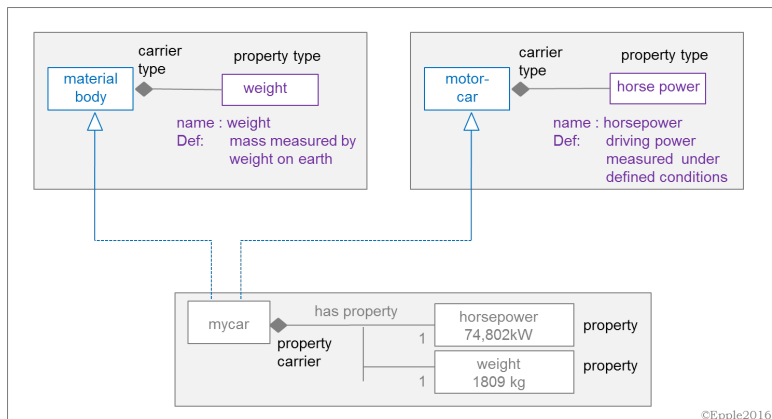


Figure 4.1: Example Showing the System of Properties, Carriers, Property Types and Carrier Types

In this example, the thing by itself is the physical car 'mycar'. It exists physically and has physical properties. It has these properties intrinsically if we know them or not. Here we use the 'horsepower' and the current 'weight' as examples. The semantics definition what a 'horsepower' or a 'weight' is, is given in the type models. Typically the carrier inherits its properties from different carrier types. In this example, 'mycar' inherits its property 'weight' from the general carrier type 'material body', its property 'horsepower' from the standards of what a 'motorcar' is. It is very usual, that a carrier inherits properties from most diverse carrier types of different domains.

As shown in the example and in Fig.2.1, the definition of a property takes place within the semantic description of a carrier type. The object, defining a property is called a *general property*. It encompasses

- the postulation of the existence of a property,
- its semantic specification and
- its name/identification.

4.1 Existence as a property by itself

On a meta level the possession of a *general property* can itself be interpreted as a *special property* of a class indicating its derivation from the respective *property carrier type*. The possession of a *general property* 'horsepower' is a *special property* of the class 'motorcar'. That means, possessing the property 'horsepower' indicates a derivative of the class 'motorcar'.

4.2 Semantic Specification Model

Semantically a *general property* can be specified from scratch or it can be specified on the base of predefined *quantity prototypes* and *quantity sorts*. As shown in Fig.4.2, especially physical properties are typically specified on the base of *quantity prototypes* and *quantity sorts*. *Quantities*

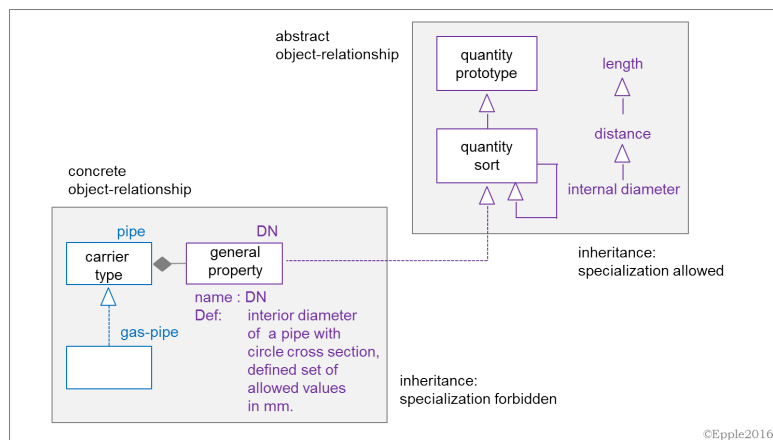


Figure 4.2: Prototypes with General Object Reference and General Properties with Concrete Object Reference

and *general properties* can be distinguished by their reference. The reference of a *quantity* is abstract but the reference of a *general property* is direct, that means, a *general property* defines exactly a certain property in each instance of the property carrier type. Physical properties are derivatives of physical quantities. They inherit the scales, the units, the methods to be measured, etc. Quantities can be derived from other quantities by specialization. A distance is a special length (measured between two abstract objects), an internal diameter is a special distance (measured between two abstract vis-a-vis walls) and DN is the characteristic diameter in the context of a concrete industrial pipe.

A main characteristic of the specification process is the prohibition of the specialization of *general properties*. It is related to the definition of a *general property*. A *general property* will be defined once in its carrier type. The specification is complete, self-contained and can not be modified afterwards. So it is forbidden to specialize a *general property* in derived classes of its carrier type. If a *general property* 'range of measurement' is defined in 'measurement device', then no specialization of it is allowed in the class 'flow meter'. 'Flow meter' inherits the *general property* 'range of measurement' from 'measurement device' unchanged.

Primary single valued properties can be handled independently from each other and individually. Adding or removing a property has no effect on other properties. They remain still valid. This allows multiple inheritance for the classes that are described by properties without causing consistency problems. The class system can be designed and varied without any restrictions by the property model.

4.3 Unified Specification Schema

The specification of a general property has different goals: It shall help humans to understand the semantic meaning of the general property, it shall contain formal elements which allow an automatic read out of characteristic parameters by the digital computer system and it shall provide an administrative shell to maintain the specification within a library of property types. To tackle these goals, the specification will contain likewise formal, semi formal, semi informal and informal elements. The IEC61360 [3] is an international standard to describe standard data element types. Even if we don not use the special coding and domain specific classification schema, we can use the general principles to structure the specification. Especially we can use the defined attributes to specify a general property. As shown in Fig.4.3, the IEC61360 orders the attributes into the categories: identifying, semantic, value, administrative and relationship. We can interpret the attributes on a metalevel as properties which describe a general property. As

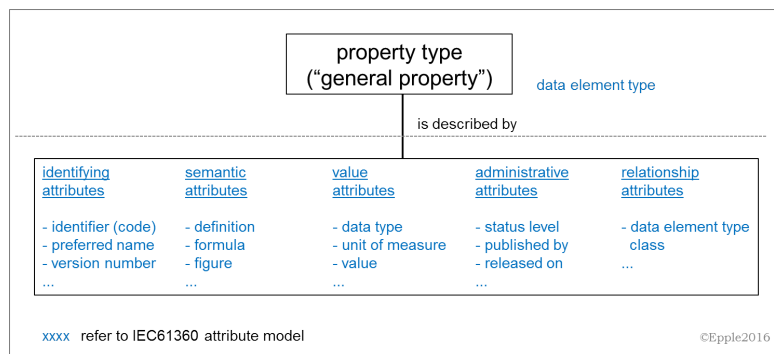


Figure 4.3: IEC61360 Template for the definition of a *general property*

shown, the attributes defined in the IEC61360 fit very well with the attributes needed to define a general property. This is still true, if we include states and configuration parameters into our single-valued-property family.

4.4 Unified Identification Schema

One of the most critical elements of the property meta model is the schema to identify a property within the context of its carrier. For this local identification, the name of the general property shall be used. As stated before (see Fig.4.1), a property carrier inherits properties of different property carrier types of typically different domains. Thus, even if it is possible to guarantee a

unique naming concept within a property carrier type, it is not guaranteed, that we get unique names within the property carrier. Different strategies can help to overcome this problem:

1. Identification together with the Property Carrier Type.

To avoid the naming problem caused by multiple inheritance, a global unique identification of a general property can be build by a combination of the name of the property carrier type plus the name of the general property. This global identifier can be used as a local name to identify a property within the scope of its carrier. The advantage of this solution is the fact, that the reference to its property carrier type is explicitly visible in the ID of a general property. A basic requirement is a unique naming schema for property carrier types.

2. Direct Global Unique Identification.

In this concept, an own independent unified naming schema for general properties is implemented. Every general property has its own unique ID. This concept is for example realized in the IEC61987 [4] and in the eCl@ss library [5]. eCl@ss separates the world of the property carrier types from the property carrier classes. The property carrier types are not explicitly modeled and appear only implicit and often unnamed in the specification of the general properties. The reference to the semantic of the property carriers is established by an assignment of the general properties to domain specific 'product classes'. Every property carrier is an instance of a specific product class. Basis is a strictly hierarchical (no multi inheritance) abstraction schema of the product classes. In eCl@ss a standardized 4-level abstraction schema is used. In this concept, a general property can be assigned to different product classes.

Independent of the chosen strategy the unified naming schema has together with the classification two functions:

- to allow a unique identification of a property within the context of a property carrier and
- to define which properties are relevant in the context of a property carrier

The second function determines the role of a thing as a property carrier and defines its assignment to the semantic classification schema.

5 Property Value Statements

An essential aspect of the property meta model is the comprehension of property value statements as autonomous objects. As discussed earlier, the property carrier is a concrete object which shows the values of its special properties at any point of time. The current weight of my car is 2016.1 kg and the front wheel, actually mounted at the left side, has a nominal diameter of 14 inches. These are properties of my concrete car. They are as they are, if we measure them or not, if we know them or not. But, independent of the property itself, we can make statements concerning the quantitative trait of a property. We can make any number of statements concerning one special property. In the information world we do not deal with the properties themselves, but with statements concerning the quantitative trait of the properties. In this chapter, we describe the property value statement model. The focus lays on the operative usage of property value statements. Fig.5.1 shows the basic structure of this part of the reference model.

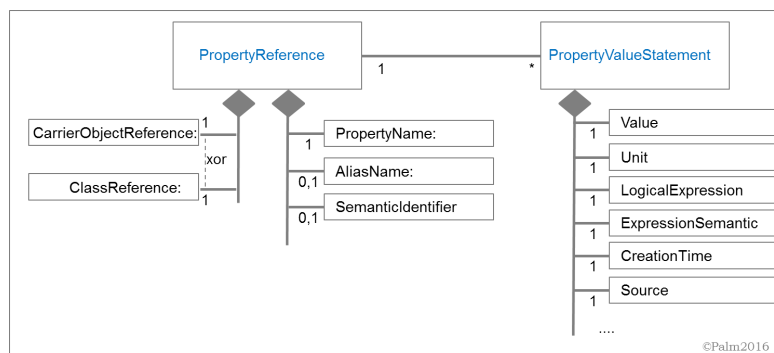


Figure 5.1: Object model of the property statement model

A property value statement defines a logical expression over a value set of a property. It consists of:

- value
defines the value for comparison in the logical expression
- unit
defines the physical unit of the value
- logical expression
defines the kind of logical relationship to specify the value set
- expression semantics
defines the semantic of the expression (requirement, assurance, ..)

- creation time
defines the point in time when the statement is stated
- source
defines the element, generating the statement.

For special purposes additional elements can be added. For example elements to describe accuracy or tolerance bands around a 'measured' statement or an element to describe the measure of quality of the statement etc. Property value statements are the standard elements to store 'knowledge' in a system. The property value statement concept allows the independent stating of any number of statements for one property. New statements do not affect existing ones. On the base of this concept standard functions and communication services can be defined to use and analyze the property statement expressions. Furthermore, by the reference to the property all statements concerning one property can be related automatically. The statement expressions concerning one property build a formalized system which can be analyzed and combined by automated functions.

5.1 Unified Expression Logic

Every statement defines a logical expression over a value set of the property. It is always involved with a comparison of an expression value x with a comparison value y .

- by a binary relationship ($=$, \neq , $<$, $>$, \geq , \leq)
- by a value set (\in , \notin)

The kind of possible expressions depends on the value scale: nominal, ordinal, interval or ratio. Fig.5.2 shows the possible logical expressions for the different scales. For nominal scaled values

logical expression \leftrightarrow	scale			
	nominal	ordinal	interval	ratio
equal ($=, \approx$)	$\{x\} \in \{y\}$	$x = y$	$x \approx y$	$x \approx y$
greater than ($>$)	-	x before y	$x > y$	$x > y$
less than ($<$)	-	x after y	$x < y$	$x < y$
greater than or equal (\geq)	-	x before or equal y	$x \geq y$	$x \geq y$
less than or equal (\leq)	-	x after or equal y	$x \leq y$	$x \leq y$

x value of the property
 y comparative value
 \approx means equal within a tolerance which is described in the property specification

©Epple2016

Figure 5.2: logical expressions defined for the different scales

for example only the operators \in and \notin are allowed. For all interval and ratio scaled variables, the $=$ operator has the meaning of an \approx operator. In these cases, two values are seen to be 'equal'

if they are within a defined tolerance band. The limits of the band can be defined individually as part of the statement or within the scope of the general property specification. Examples: measurement statement: the temperature of the fluid is equal 43.2°C (measured by a special temperature measurement with a known but small measurement deviation), assurance statement: the weight of a product is equal 34.5kg (within the deviation which is allowed for all weighing measurements by trade law).

5.2 Unified Expression Semantic


Every statement defines a logical expression over a value set of the property. The expressions have different meanings which are indicated in the expression semantic:

- requirement expression (R):
The one who makes the statement requires that the property instance fulfills the represented condition. The determination of a configurable setting is also a requirement.
Examples:
 - the car I want to buy shall have four seats minimum
 - the reactor shall be in 'reparation mode'
 - the catalog of the manufacturer requires: the 5V power adaptor needs a power supply of 110V or 230V
 - the electric consumer of my electrical 230V socket must consume less than 10A
- assurance expression (A):
The statement guarantees that the property instance fulfills the represented condition.
Examples:
 - the catalog of the manufacturer assures: Every car of type xxx has five seats.
 - the power supply of the electrical socket in my office is 230V
 - the catalog of the manufacturer assures: the 5V power adaptor consumes less than 0.5A
- measurement expression (M):
The statement gives a read out or measured information of the real 'is-value' of a property at a certain point of time. Examples:
 - measurement information: Weight of my car at 12:24:33 has the measured value 2212.34 kg
 - the current mode of the reactor is 'reparation mode' (read out of the mode state)
- setting expression (S):
The statement states a value which is fixed by a setting process or by construction. Examples:
 - nominal diameter of a pipe is DN20
 - measurement range is set to 0-100 C

- new version is set to V3.2

5.3 Type Assigned Statements

If a property value statement is assigned to a type, then it is valid for all derived types, instances and realizations. When a property value statement is assigned to an instance, then it is valid only for a unique special property. Fig.5.3 shows a typical example of a catalog statement list concerning a product type of a manufacturer. All the statements in the list are assigned to

27-20-04-05		variable area flowmeter			
0173-1#02-AA0677#001	manufacturer name	Krohne	S	∈	
0173-1#02-AA0847#003	description of product type	variable area flowmeter	S	∈	
0173-1#02-AA0734#001	manufacturer product description	VA40	S	∈	
0173-1#02-BAG982#006	nominal width	DN25	S	∈	
0173-1#02-BAB322#005	form	N21.9	S	∈	
0173-1#02-BAA048#006	max measurement error	1 %	A	<	
0173-1#02-BAA036#008	max. process temperature	100 C	R	<	
0173-1#02-BAG943#005	lowest value for volume flow rate	063 l/h	R	>	
0173-1#02-BAG944#005	greatest value for volume flow rate	630 l/h	R	<	

source: <http://www.eclaccontent.com/index.php?language=en&version=9.1> source: <http://www.krohne.com/09/2013-4000416902-TD-VA40/45-R02-en>
©Epple2016

Figure 5.3: example of type assigned statement list in a catalog

a product type, here the type VA40, a variable area flowmeter of the Krohne company. The properties are referenced by their names (second row). Additionally they are referenced by their unique eCI@ss-code (first row). The third row contains the value settings which characterizes the type VA40. In the figure we added two additional rows indicating explicitly the implicit semantic meaning and the logic expression of the statements. As we can see, there are assurance expressions (A), requirement expressions (R) and settings (S) as well.

5.4 Formal Reasoning

With the presented small sets of statement semantics and logical expressions the basic check and select tasks needed in engineering and operation can be formulated formally and executed automatically without any further semantic knowledge. For example, usability checks of devices (fits a device to the role requirements?), checks for plan inconsistencies, tests for interpretation errors, the search for unused reserves etc. can be executed without any further specifications or engineering efforts. Fig.5.4 shows some results we get by the conjunction of different statements. The unified expression logic together with the unified expression semantic build a semi-semantic system which allows an automated reasoning without any further knowledge of the domain specific semantic.

Statement A1	Statement A2	logical result of conjunction
requirement	assurance	<ul style="list-style-type: none"> - ok - not ok
requirement	requirement	<ul style="list-style-type: none"> - A1 part of A2 - A2 part of A1 - A1+A2 stronger than A1 and A2 - contradiction
assurance	detection	<ul style="list-style-type: none"> - ok - not ok
requirement	setting	<ul style="list-style-type: none"> - ok - not ok
setting	setting	<ul style="list-style-type: none"> - fits - does not fit

©Epple2016

Figure 5.4: logical results of the conjunction of two statements

6 Administration of property value statements

Within the engineering and operative environment the property statements play a dominant role. Their values carry the operative semantic knowledge of the product, the process and the site. Their standardized representation, semantic and logic allow their processing by standardized functions and services. This is a powerful base enabling system interoperability. The administration of statements is complex and needs a practical organization schema. The basic characteristics are:

- Every statement is a unique and independent entity.
- Every statement is part of a special view onto a system. Every view can and will generate its own statements. The result is a manifold of statements of different views concerning the same property.
- Typically, statements once stated will never be changed anymore.
- Statements can be stored in the system forever, can be deleted when not needed anymore or can be replaced by a new statement with updated values.

In many cases it is useful to administrate all the statements of a special view within one statement list or an ordered group of statement lists. A vendor catalog is an example for such an ordered group of statement lists. It contains for all the products a vendor offers a product type specific list containing the information the vendor wants to publish to interested customers. Another example is the requirement list of an industrial user. As a result of a planning process, he gets functional role specifications which formulate requirement statements for potential realisation units (The planned pumpfunction N15 requires a pump with...). Especially statement lists with statements referring all the same carrier are very practical. They allow the declaration of the carrier reference once for all the list entries. In summary, the organization of the property value statement lists is closely coupled to the structures of the specific system-, product- process- and life cycle models.

7 Summary

In field of industrial automation, there is a variety of technical IT-Systems that have to exchange information with each other. For example, planning systems of the plants, PCS-planning system, catalog systems of the manufacturer, order systems, process control systems, maintenance systems, optimization systems, etc. The different tasks cause the systems to have different functions, models and semantics. It seems not possible to integrate such a system into one common model. The property model offers the possibility to establish a standardized interoperability concept for such heterogeneous system environments. It allows to exchange information about each single property by standardized services. The base is an agreement about the principle concepts and a common library of general properties. Every system that knows a general property can make statements to any special property or group of special properties of this type. Statements can be requirements, assurances, measurements or settings. Every system that is interested in a special property can inform itself about the statement expression that are applied to this property in other systems. It can be tested if two statements concerning the same property match. This can be done by general basic rules without any understanding of the specific meaning of the property. By considering the value statements on properties as autonomous and standard elements of a semi-semantic model, we get the possibility of an operative information exchange across the heterogeneous system of systems.

Bibliography

- [1] EPPLE, ULRICH: Merkmale als Grundlage der Interoperabilität technischer Systeme. at - Automatisierungstechnik, 59(7):440–450, July 2011.
- [2] MERTENS, MARTIN: Verwaltung und Verarbeitung merkmalsbasierter Informationen: vom Metamodell zur technologischen Realisierung, volume 1201 of Fortschritt-Berichte VDI Reihe 8. VDI Verlag, Duesseldorf, 2012.
- [3] N.N.: IEC61360 Standard data element types with associated classification scheme for electric components. International Standard. IEC61360-1 Ed.3 Part 1: Definition - Principles and methods, IEC, 2007.
- [4] N.N.: IEC61987 Industrial-process measurement and control - Data structures and elements in process equipment International Standard. IEC61987-11:2012 List of Properties (LOP) of measuring equipment for electronic data exchange - Generic structures, IEC, 2012.
- [5] N.N.: eCl@ss library. Consortial Standard. eCl@ss 9.1 , www.eclass.eu, 04/7/2016), eCl@ss e.V., 2016.