UML issues

Summary

This document described several issues with the UML profile defined in ISO 19103 and used/extended by e.g. INSPIRE:

1. The stereotypes dataType and enumeration are invalid according to the UML specification.
2. Reuse of model elements should be indicated with a usage dependency, not a package import
3. Tagged values must only be present on stereotyped model elements.

Some other possible issues are identified as well.

Issue #1: datatypes and enumerations

UML specification

Classifier

To understand this issue, one needs to understand the concept of a **classifier**: a classifier is a classification of instances, it describes a set of instances that have features in common (UML 2011, 51).

Some examples of classifiers are: data type, primitive type, enumeration, interface, class, component, collaboration, use case and actor, see also Figure 1.

Note that an enumeration is a kind of data type, and that a data type is **not** a kind of class. This is stated explicitly in Weilkiens and Oestereich, 16: a simple data type (DataType, Ed.) and a primitive data type (PrimitiveType, Ed.) have nothing to do with a class, except that they share a common superclass, Classifier.

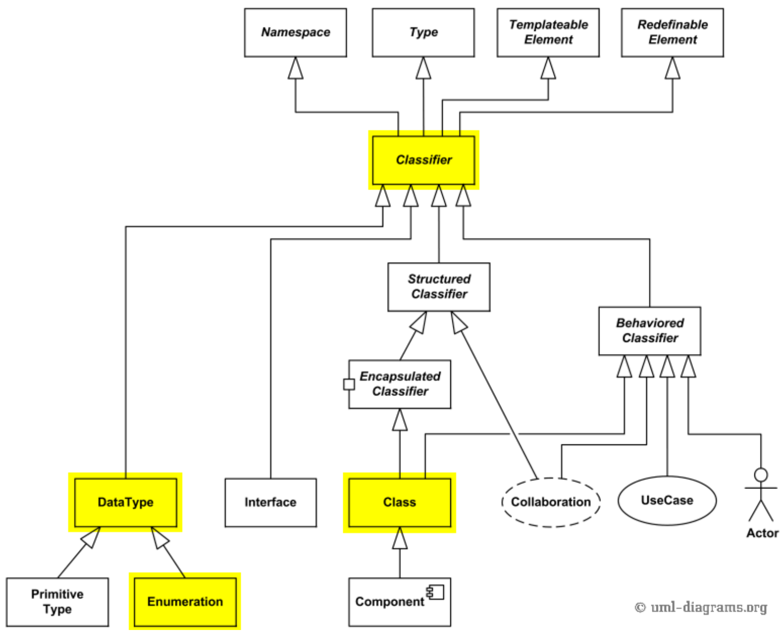

Figure 1: The most important UML classifiers (modified from Fakhroutdinov). The once of relevance to this document are marked with yellow.

The standard notation for Classifier is a rectangle that contains the name with the name of the subclass in guillemets above it (see Figure 2). Each subclass introduces notation variants. For example, the name of the concrete metamodel class («class») is normally omitted (Weilkiens and Oestereich, 36).

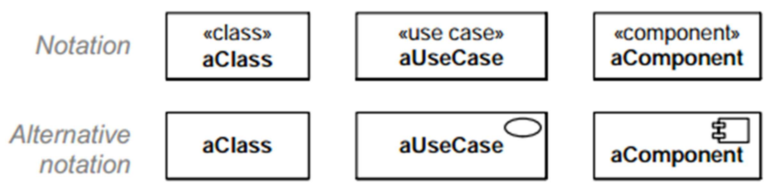

Figure 2: Examples of the notation of Classifier subclasses (Weilkiens and Oestereich, 36). Enterprise Architect uses the alternative notation for classes.

Classes, data types and enumerations

The notation of classes, data types and enumeration is described as follows in the UML specification:

- A class is shown using the classifier symbol. As class is the most widely used classifier, the **keyword "class"** need not be shown in guillemets above the name. A classifier symbol without a metaclass shown in guillemets indicates a class (UML 2011, 49)
- a data type is denotated using the rectangle symbol with **keyword «dataType»** or, when it is referenced by (e.g., an attribute) denoted by a string containing the name of the data type (UML 2011, 60)
- An enumeration may be shown using the classifier notation (a rectangle) with the **keyword «enumeration»** (UML 2011, 67)

The notation of classes is shown in Figure 2, the notation of data types and enumeration is shown in Figure 3. Weilkiens and Oestereich say explicitly about Figure 3 that the labels shown are UML keywords, not stereotypes (Weilkiens and Oestereich, 16).

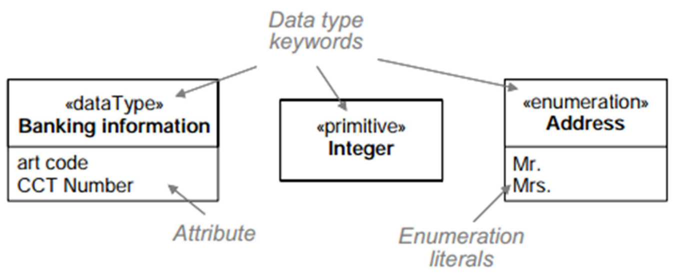

Figure 3: Examples of data types (Weilkiens and Oestereich, 16)

Note that what is described in this and the previous paragraph is valid for both UML 1 and UML 2.

Stereotypes

UML uses the exact same notation for keywords and stereotypes, both are enclosed in guillemets (« »), see UML 2011, 697:

Keywords are always enclosed in guillemets («keyword»), which serve as visual cues to more readily distinguish when a keyword is being used. […] In addition to identifying keywords, guillemets are also used to distinguish the usage of stereotypes defined in user profiles. This means that:

1. Not all words appearing between guillemets are necessarily keywords (i.e., reserved words), and

2. words appearing in guillemets do not necessarily represent stereotypes

ISO/INSPIRE

The UML-profile of INSPIRE, which follows ISO 19103, contains the following stereotypes (INSPIRE 2014, 55-58):

Stereotype	Metaclass	Description
dataType	Class	A structured data type without identity. [ISO/TS 19103]
enumeration	Class	An enumeration

This profile is invalid according to the UML specification, because of the following two reasons:

1. the main reason: the stereotypes dataType and enumeration are used to kind of reinvent concepts that already are present in core UML
2. a more technical reason: the stereotypes dataType and enumeration violate the following part of the specification: UML keywords are reserved words that are an integral part of the UML notation and normally appear as text annotations attached to a UML graphic element or as part of a text line in a UML diagram. These words have special significance in the context in which they are defined and, therefore, cannot be used to name user-defined model elements where such naming would result in ambiguous interpretation of the model (UML 2011, 697).

This issue is probably caused by the fact that UML uses the exact same notation for keywords and stereotypes, which is confusing.

So a correct profile would (1) use the correct UML-metaclasses and (2) define stereotypes having names that obey the following constraint on stereotypes: stereotype names should not clash with keyword names for the extended model element (UML 2011, 680).

Stereotype	Metaclass	Description
~~dataType~~ must not be called dataType	~~Class~~ DataType	A structured data type without identity. [ISO/TS 19103]
~~enumeration~~ must not be called enumeration	~~Class~~ Enumeration	An enumeration

Issue #2: relationships between packages

UML specification

Package Import

A package import is a relationship that allows the use of **unqualified** names to refer to package members from other namespaces (UML 2011, 112). Note that one can always refer to public package members from other namespaces: the public contents of a package are always accessible outside the package through the use of **qualified** names (UML 2011, 110).

Two import options exist:

- public visibility, meaning that the import is transitive (Weilkiens and Oestereich, 26)
- private visibility, meaning that the import is not transitive (Weilkiens and Oestereich, 26)

The predefined **keywords** are **«import»** for a public package import, and «access» for a private package import (UML 2011, 112).

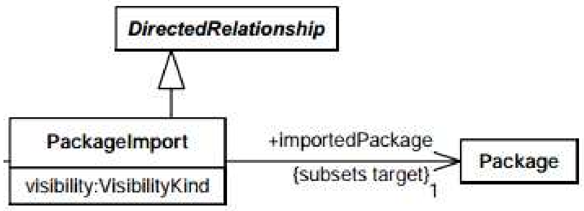

Figure 4: A package import is a kind of directed relationship (modified from Weilkiens and Oestereich, 28)

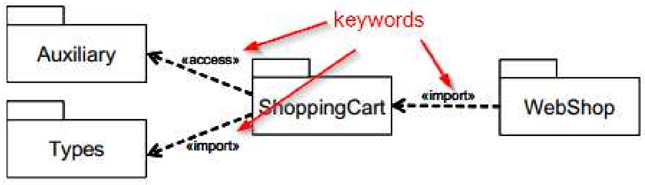

Figure 5: Examples of public and private package imports (modified from UML 2011, 113)

Usage Dependency

A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation. In the UML metamodel, a Usage is a Dependency in which the client requires the presence of the supplier (UML 2011, 139). A usage dependency is shown as a dependency with a «use» keyword attached to it (UML 2011, 139).

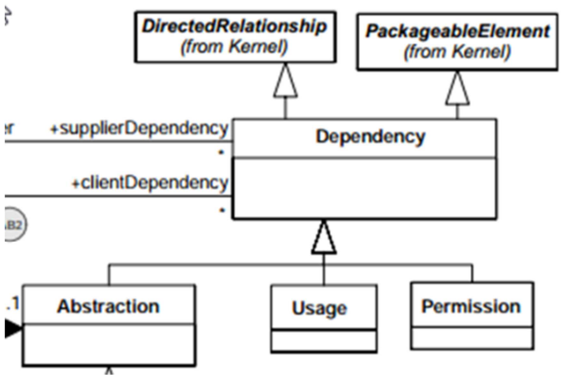

Figure 6: Part of the metamodel for dependencies (modified from Weilkiens and Oestereich, 67). Note that a dependency is a kind of directed relationship. A dependency has nothing to do with a package import, except that they share a common superclass, DirectedRelationship.

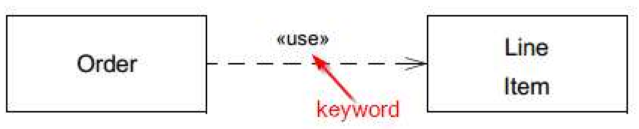

Figure 7: An example of a use dependency (modified from UML 2011, 139)

ISO/INSPIRE

The UML-profile of INSPIRE, which follows ISO 19103, contains the following stereotype (INSPIRE 2014, 55-58):

Stereotype	Metaclass	Description
import	Dependency	The model elements of the supplier package are imported

Naming a stereotype import is against the UML specification, for the same reasons as identified in the first issue about data types and enumerations. A correct way of modelling a package import is to use the package import as built in in UML, as explained above.

However, the actual need for ISO/INSPIRE is reusing classes, data types, etc. defined in other packages, and is not being able to refer to those classes, data types, etc. by means of unqualified names. Usage dependencies fulfill exactly this need, on the condition that classes, data types, etc. have public visibility. No stereotype is needed on a usage dependency in ISO/INSPIRE, since no additional tagged values are required on it.

This issue is probably caused by the fact that in UML 1, an import was a stereotyped permission dependency, an element that is obsolete in UML 2.

Issue #3: Tagged values on elements without stereotype

UML specification

In UML 2, a tagged value can only be represented as an attribute defined on a stereotype. Therefore, a model element must be extended by a stereotype in order to be extended by tagged values (UML 2011, 686).

INSPIRE

The UML-profile of INSPIRE, which follows ISO 19103, defines tagged values on unstereotyped model elements (the last four entries in the table) (INSPIRE 2014, 55-58), e.g. inlineOrByReference on attributes and association roles. According to UML 2, a stereotype with tagged value inlineOrByReference should be defined.

Other issues

Issues with the ISO profile have been identified by T. Kutzner, where the conclusion is as follows: The UML profile in ISO 19103 cannot be regarded a UML profile in the sense of the UML profile definition of the OMG [...] this holds true for every UML schema which uses the ISO 19103 profile (e.g. INSPIRE, AAA), since these schemas make use of the stereotypes “CodeList” and “Union” as well (Kutzner 2012, 56).

Normally a stereotype's name and the name of its applications start with upper-case letters, to follow the convention for naming classes (UML 2011, 681). This is a change from earlier UML versions: UML versions before 2.4 required the first letter of the name of the applied stereotype to be in lower case (e.g. «servlet»). Starting from UML 2.4, the first letter should normally be in upper case (Fakhroutdinov, 2014). This means that e.g. «FeatureType» should be used instead of «featureType».

The UML metamodel does not have a metaclass for attributes. An attribute is a role that a property can take on. In contrast to UML 1.x, UML 2.0 no longer strictly separates attributes and association ends. This means that the representation of an attribute in a class and its representation as a navigable association are the same (Weilkiens and Oestereich, 42). This has an influence on the INSPIRE profile, which defines several stereotypes that extend attributes and association roles (INSPIRE 2014, 55-58).

References

Fakhroutdinov, Kirill. "The Unified Modeling Language". Accessed May 2014 http://www.uml-diagrams.org.

INSPIRE, DTDS. D2.5: *Generic Conceptual Model, Version 3.4*. 2014.

Kutzner, Tatjana, and Andreas Donaubauer. "Critical Remarks on the Use of Conceptual Schemas in Geospatial Data Modelling–A Schema Translation Perspective." *Bridging the Geographic Information Sciences* (2012): 43.

UML, OMG. *2.4. 1 superstructure specification*. document formal/2011-08-06. Technical report, OMG, 2011.

Weilkiens, Tim, and Oestereich, Bernd. *UML 2 Certification Guide: Fundamental & Intermediate Exams*. Morgan Kaufmann, 2010.

Annex: Enterprise Architect

Up to EA 9, enumerations were represented as stereotypes of classes. From EA 10, the UML specification is implemented correctly: enumerations now correspond to full element types instead of stereotyped classes, the release history of EA 10 states.

The UML specification says: If multiple stereotypes are applied, the names of the applied stereotypes are shown as a comma-separated list with a pair of guillemets. When the extended model element has a keyword, then the stereotype name will be displayed close to the keyword, within separate guillemets (example: «interface» «Clock») (UML 2011, 681). Note that the last sentence has been changed slightly in UML 2.5, Beta 2: When the extended model element has a keyword, then the stereotype name(s) will be displayed close to the keyword, within the same or separate guillemets (example: «interface» «Clock» or «Clock, interface»). EA does not show the keyword «dataType» on stereotyped data types, see also the Sparx Forum, and displays stereotyped enumerations as «enumeration, MyStereoType», thus displaying keyword and stereotype name within the same guillemets.