UML-to-GML Application Schema Pilot (UGAS-2019) Summary Report

Publication Date: YYYY-MM-DD

Approval Date: YYYY-MM-DD

Submission Date: YYYY-MM-DD

Reference number of this document: OGC 19-009

Reference URL for this document: http://www.opengis.net/doc/PER/ugas2019-ID

Category: OGC Summary Report

Editor: Johannes Echterhoff

Title: UML-to-GML Application Schema Pilot (UGAS-2019) Summary Report

OGC Summary Report

WARNING

This document is not an OGC Standard. This document is an OGC Summary Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Summary Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the

Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Table of Contents

Chapter 1. Background

International Organization for Standardization (ISO) 19109:2015 *Geographic information – Rules for application schema* Clause 2.3 *UML application schema*, defines a Unified Modeling Language (UML) metaschema for feature-based data. That General Feature Model (GFM) serves as the key underpinning for a family of ISO standards employed by national and international communities to define the structure and content of geospatial data.

ISO 19136:2007 *Geographic information – Geography Markup Language (GML)* Annex E U*ML-to-GML application schema encoding rules*, defines a set of encoding rules that may be used to transform a GFM-conformant UML concept model to a corresponding XML Schema (XSD) based on the structure and requirements of GML. The resulting XSD file(s) may be used to define the structure of, and validate the content of, XML instance documents used in the exchange of geospatial data among open-source, commercial, consortia, and government systems. This methodology for deriving technology-specific encodings of GFM-conformant UML concept models based on formalized sets of encoding rules has been successfully applied in other contexts than GML (e.g., technologies based on Resource Description Framework (RDF) encodings) and has come to be generically known as “UGAS”.

ShapeChange (https://shapechange.net/) is an open-source implementation of the UGAS methodology that is regularly employed by the international standards community in the application of ISO standards to the resolution of issues involving data model standardization, content validation, and exchange of geospatial data. The U.S. National System for Geospatial Intelligence (NSG) Application Schema (NAS) standardizes an NSG-wide model for geospatial data that is mission-agnostic and technology-neutral. From it, using Model Driven Architecture (MDA) techniques, technology-tied Platform Specific Models (PSM) may be automatically derived and directly employed in system development. ShapeChange is a key technological underpinning that is heavily employed in development and management of NAS-related technology artifacts.

Chapter 2. Summary

The UGAS-2019 Pilot was undertaken in order to improve and enhance the present capabilities of ShapeChange in order to meet current and emerging technology requirements associated with the NSG Application Schema (NAS). Those requirements fell into four areas, as follows:

1. Develop UML model representations to enable enhanced support for NGA analytic tradecrafts, achieved through the addition of support for UML property stereotypes;
2. Engineering analysis and then enhancements to make greater use of Object Constraint Language (OCL) expressions in UML models and derived technology-specific schemas;
3. Develop enhancements to ShapeChange that enable the retention of a greater amount of information through all stages of ShapeChange-based transformation; and,
4. Develop enhancements to the related Profile Management Tool (PMT) in order to satisfy specific requirements for defining profiles of the NAS.

The UGAS-2019 Pilot accomplished key objectives in each of these areas, as follows.

1. Added support in ShapeChange for a new property stereotype for metadata alongside the existing property stereotype voidable. In consequence of employing both property stereotypes the complexity of the NAS – caused by the Meta- and Reason-classes previously required in order to support attribute-specific security markings and data quality metadata – was reduced significantly (over 9,000 UML classes were eliminated). A revised NAS will be much more usable by Subject Matter Experts and maintainable by domain modelers.
2. Implemented expressive OCL constraints accompanied by efficient Schematron assertions, enabling broader application of data content validation mechanisms with NAS-based XML instance documents, resulting in higher levels of data integrity and reduced content errors. Completed the accompanying OCL-to-OWL analysis, enabling enhancement of the NSG Enterprise Ontology (NEO) to achieve increased expressivity in knowledge representation and the ability to implement data integrity checks in a linked-data environment.
3. ShapeChange XML (SCXML) can now be used as a format for lossless exchange of NAS model content. The format has been methodically revised to result in a consistent structure throughout, is now fully supported by ShapeChange, and all standard model loading parameters now apply to models loaded from SCXML.
4. The Profile Management Tool has been enhanced to fully support handling of UML association classes in application schemas. With additional improvements, such as the addition of a number of profiling parameters, the PMT is now suited for the specification and maintenance of NAS profiles by mission specialists.

2.1. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

Name	Organization	Role
Johannes Echterhoff	interactive instruments GmbH	Editor
Clemens Portele	interactive instruments GmbH	Contributor

2.2. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 3. References

The following normative documents are referenced in this document.

- ISO 19109:2015 Geographic information – Rules for application schema
- ISO 19136:2007 Geographic information – Geography Markup Language

Chapter 4. Terms and definitions

4.1. Abbreviated terms

CFP	Call for Proposals
GFM	General Feature Model
GML	Geography Markup Language
GML-SF	GML Simple Features
GUI	Graphical User Interface
IP	Innovation Program
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
MDA	Model Driven Architecture
NAS	NSG Application Schema
NEO	NSG Enterprise Ontology
NSG	U.S. National System for Geospatial Intelligence
OCL	Object Constraint Language
OGC	Open Geospatial Consortium
OWL	Web Ontology Language
PMT	Profile Management Tool
PSM	Platform Specific Model
RDF	Resource Description Framework
SCXML	ShapeChange XML
SHACL	Shapes Constraint Language
SWRL	Semantic Web Rule Language
SQL	Structured Query Language
UGAS	UML to GML Application Schema
UML	Unified Modeling Language
XML	Extensible Markup Language
XPath	XML Path Language
XSD	XML Schema
XSL	Extensible Stylesheet Language
XSLT	XSL Transformation

Chapter 5. Overview

The UML-to-GML Application Schema Pilot (UGAS-2019) initiative continued work conducted in previous OGC Innovation Program (IP) activities, such as OGC Testbed-12, -13, and -14, in the area of application schema modeling and processing. The goal of the UGAS-2019 Pilot was to design and implement solutions for multiple tasks. The following list provides brief descriptions of these tasks, preceded by the according deliverable number.

- D100 - Analyze property stereotypes for metadata assignments and design the necessary changes for ShapeChange to support such a stereotype.
- D101 - Enhance the XML Schema target of ShapeChange to support the conversion of the property stereotype for metadata.
- D102 - Improve the conversion of OCL constraints to Schematron assertions, using the XSLT2 query binding, and thus leverage the improvements of the XPath v2.0 language (compared to XPath v1.0).
- D103 - Complete the analysis from OGC Testbed-14 of translating OCL constraints to OWL expressions.
- D104 - Enhance the ShapeChange XML input loader.
- D105 - Enhance the ShapeChange XML format (SCXML) to support the lossless exchange of NAS content.
- D106 - Improve the handling of association classes by the ShapeChange Profile Management Tool (PMT).
- D107 - Enhance the PMT to support additional parameters.

The following chapters document each task and its requirement(s) in more detail, as well as the results delivered by the UGAS-2019 Pilot.

NOTE	All ShapeChange enhancements developed within the UGAS-2019 Pilot will be publicly released as a component of ShapeChange v2.9.0. https://shapechange.net has been updated to document the enhancements.

Chapter 6. Property stereotype for metadata - D100

6.1. Task / Requirement(s)

For this deliverable, the Pilot was tasked to analyze how property stereotypes can be used within an application schema to indicate that the property can be associated with a (specific type of) metadata.

NOTE	As mentioned in the CFP of the UGAS-2019 Pilot, metaclass ValueAssignment, from the General Feature Model (GFM) (defined in ISO 19109:2015), provides an example of such a property stereotype.

As part of this analysis, the implications of using such stereotypes within an application schema - with the NSG Application Schema (NAS) being the primary use case - were to be considered. Once the concept of property stereotypes for metadata association has been established, the goal then was to extend ShapeChange and the ShapeChange XML (SCXML) format as necessary to support it.

6.2. Results

The concept of using a property stereotype for associating metadata with property values has been developed through a series of web meetings and email conversations. A separate document [https://shapechange.net/wp-content/uploads/2019/12/UGAS19-D100_property_stereotypes.pdf] captures the results of these discussions in detail. The results of this work were presented within the OGC Architecture DWG at the OGC TC meeting in Toulouse, in November 2019.

NOTE	The presentation is available on the OGC Portal at: https://portal.opengeospatial.org/files/?artifact_id=90866

The concept is realized via the new property stereotype <<propertyMetadata>>. When added to a property (an attribute or association role) within an application schema, it signifies that metadata can be associated with the values of the property. By assigning tagged value "metadataType" to the property, the exact type of that metadata can be defined by the modeling expert - per property. Doing so supports use cases in which different kinds of metadata are required for different properties, for example attributes and association roles, or properties with a simple and a complex value type.

NOTE	This essentially enables the <<AttMeta>> and <<RoleMeta>> stereotypes mentioned in the CFP of the UGAS-2019 Pilot.

The new property stereotype can significantly reduce the complexity of an application schema, which used the standard modeling constructs defined by ISO 19103 and ISO 19109 to model property value types with metadata. The NAS is a good example of such an application schema. Currently, the NAS uses Meta- and Reason-classes to model that a property has a set of values or a reason why no value exists, and also metadata for the property value - see Figure 1.

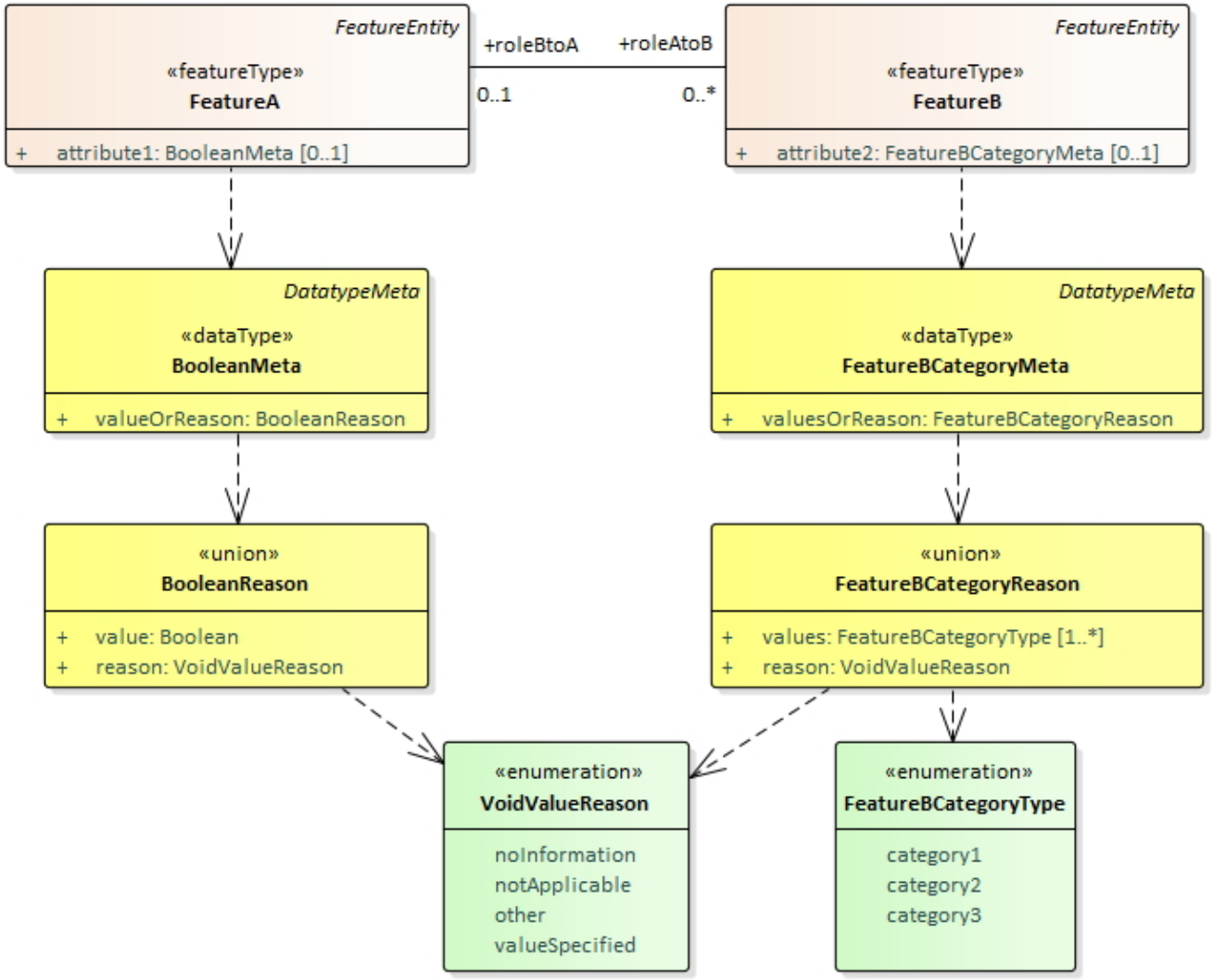

Figure 1. Current approach for modeling property metadata in the NAS, using specific Meta- and Reason-classes

With the new property stereotype <<propertyMetadata>> - as well as the property stereotype <<voidable>>, to indicate that a property can be null, with a defined reason - the Meta- and Reason-classes are no longer needed.

NOTE For further details, see the document at https://shapechange.net/wp-content/uploads/2019/12/UGAS19-D100_property_stereotypes.pdf, chapter 2.2.

The model can be revised/simplified to the one shown in Figure 2.

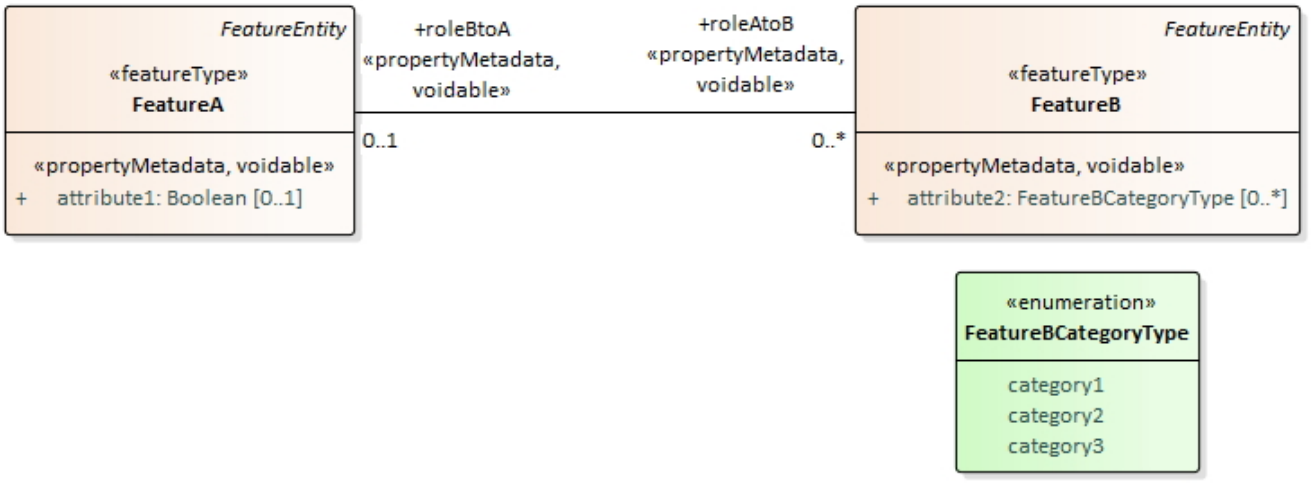

Figure 2. New approach for modeling in the NAS, using property stereotypes <<propertyMetadata>> and <<voidable>>

The resulting model design pattern is much easier to understand and then consistently implement and maintain.

The effect of employing the pair of property stereotypes also applies to implementation schemas, such as XML and JSON schemas.

NOTE More detailed information about encodings of <<propertyMetadata>> can be found in the document at https://shapechange.net/wp-content/uploads/2019/12/UGAS19-D100_property_stereotypes.pdf, chapter 2.3.

These schemas become simpler to implement, easier to understand, and more readily employed in application software.

Additional benefits of replacing the Meta- and Reason-classes with property stereotypes are:

- The actual cardinality of a feature property is directly visible.
- The size of the application schema (in number of classes) significantly decreases; in the case of the NAS more than 9,000 datatypes were eliminated.
- The model clearly separates the domain content model from property metadata.
- OCL expressions can be formulated in a natural way.

NOTE The last point concerning OCL expressions is only relevant for a small set of modeling experts that actually use OCL. Nevertheless, OCL represents a powerful technology which allows these experts to define constraints and conditions that apply to model elements, which cannot be expressed with UML alone. The modeling expert can thus model the universe of discourse in more detail. Conditions represented by OCL constraints can be implemented in different implementation technologies, for example as Schematron assertions, to be evaluated for data encoded in XML. Deliverable D102 was concerned with improving the translation of OCL to Schematron. For further detail, see chapter Completing the translation of OCL constraints to OWL expressions - D103.

Work on D100 also included the design of an OCL language extension. The new operation call *propertyMetadata()* provides programmatic access to the metadata object that is associated with the value of a property with stereotype <<propertyMetadata>>. That means that a modeling expert can explicitly model conditions that involve the metadata that is associated with property values. An example would be that the value of a geometric property must be associated with metadata that provides information on the positional accuracy.

Multiple encodings of stereotype <<propertyMetadata>> were considered during the analysis in D100. The primary focus was on the XML encoding, for which a number of options were considered. The recommended option provides a simple and straightforward solution for encoding the relationship between a property value and a metadata object in XML. That option was implemented in D101 - see chapter Conversion of property stereotype for metadata in the ShapeChange XML Schema target - D101.

Even though not explicitly required for UGAS-2019, the analysis of encodings of stereotype <<propertyMetadata>> encompassed JSON, SQL, GML Simple Features, and Linked Data encodings. The results of D100 provide suggestions on how <<propertyMetadata>> can be realized in these encodings. Since implementation and testing of these suggestions were not in scope of the UGAS-2019 Pilot, such work will need to be conducted in future activities.

Summarizing the key results of D100:

- The new stereotype <<propertyMetadata>>, in combination with stereotype <<voidable>>, can significantly reduce the complexity of an application schema like the NAS, thus improving its usability and maintainability.
- The simplification of application schemas like the NAS enables encodings for use in data storage and/or exchange that are simpler to implement and use. A simple XML encoding of the new concept has been developed.
- The new <<propertyMetadata>> stereotype is also supported in OCL expressions. This approach to modeling property metadata is thus supported in the two languages (UML and OCL) that are primarily used for modeling application schemas.

Chapter 7. Conversion of property stereotype for metadata in the ShapeChange XML Schema target - D101

7.1. Task / Requirement(s)

Once the concept of a property stereotype for metadata had been developed in D100, suitable rule(s) were to be developed, to convert stereotype <<propertyMetadata>> in the XML Schema encoding of an application schema, with the NSG Application Schema (NAS) being the primary use case. The resulting XML Schema should be equivalent, but possibly not identical, to the current NAS XML Schema encoding. The conversion process should take into account property-specific OCL constraints, as well as preceding model transformations.

7.2. Results

ShapeChange was enhanced as follows:

- A conversion rule was added to encode property stereotype <<propertyMetadata>> as an optional XML attribute *metadata* on the XML element that represents the property. The new XML attribute can be used to link to a metadata object, much like other objects can be referenced in an XML encoding using the XML attribute *xlink:href*.
- The Schematron conversion was extended to:
  - Translate the new OCL operation call *propertyMetadata()*.
  - Generate assertions to check that a property either is null, with a defined reason, and no values - or that it only has values. These checks allow enforcing the choice represented by the Reason-classes mentioned in chapter Property stereotype for metadata - D100.
  - Generate assertions to check that a nilReason XML attribute given on an XML element that represents a property has a value equal to one of the enum-literals from a specific enumeration defined in the model. The enumeration can be defined per property (via a tagged value) or via a ShapeChange configuration parameter.

These enhancements fully support the concept of stereotype <<propertyMetadata>> in the XML encoding. They ensure XML data checks and expressiveness equivalent to the current NAS XML Schema.

NOTE	Model transformations do not affect the conversion of stereotype <<propertyMetadata>>. Such transformations can only change which properties have this stereotype, and which kind of metadata may be associated with a property that has the stereotype.

NOTE	The property stereotype <<voidable>> was already supported by ShapeChange before the UGAS-2019 Pilot.

Chapter 8. Conversion of OCL constraints to Schematron assertions, using the XSLT2 query binding - D102

8.1. Task / Requirement(s)

OGC Testbed-14: Application Schema-based Ontology Development Engineering Report (OGC 18-032) [http://docs.opengeospatial.org/per/18-032r2.html] documents recommendations for writing OCL constraints that are intended to be translated to OWL expressions. A major aspect of those recommendations is to significantly increase the use of quantifications (exists(..) and forAll(..)) in OCL expressions. These quantifications are readily supported by OWL, however extensive use of such quantifications can be an issue for the translation to Schematron based on XSLT1 - which is what the OCL to Schematron conversion of ShapeChange currently supports.

NOTE	Schematron is a language for making assertions about the presence or absence of patterns in XML instance documents; it is used to supplement XML Schema (XSD) as a means for testing data integrity and identifying content errors.

The reason is that Schematron with the XSLT1 query binding does not directly support quantifications. More specifically: XPath 1.0, which is used by XSLT1 and thus also used to define Schematron assertions, does not support them. The conversion by ShapeChange represents quantifications through equivalent XPath expressions. Especially the representation of forAll(..) can quickly lead to highly complex and potentially inefficient XPath expressions; a chain of forAll(..) statements would result in a deeply nested tree of count expressions. The situation could significantly improve if XSLT2 was used as Schematron query binding, since XPath 2.0 directly supports quantified expressions.

The goal of D102 therefore was to enhance the current OCL constraint to Schematron assertion conversion capability of ShapeChange to perform the conversion to Schematron based on XSLT2 instead of XSLT1.

8.2. Results

The ShapeChange conversion of OCL constraints to Schematron has been revised. Full use of the XSLT2 query binding now not only allows a straightforward representation of quantifications, but also fully enables evaluation of property values given by reference (given that all objects are encoded within the same XML document, such as a feature collection). When evaluating such references, the Schematron code makes use of XSLT2 keys, which significantly improve the performance of such lookups. Other OCL language constructs can now be translated in a more natural and direct way, too, such as if-then-else-endif and the pattern matching function.

NAS OCL constraints can now be revised to use quantifications, as suggested in the OGC Testbed-14 engineering report. The revised constraints will be easier to translate to OWL expressions. With the new Schematron translation process, the revised constraints can also be translated to efficient Schematron assertions, which can be used for validating NAS data encoded in XML.

Overall, the revision of the Schematron translation, making full use of the language features of the XSLT2 query binding, resulted in a significant improvement of the resulting Schematron: the Schematron assertions are easier to understand, make use of XSLT2 language features to achieve faster execution, and new capabilities (*e.g.*, value pattern matching) useful for the NAS are supported. The implementation of expressive and efficient Schematron assertions enables broader application of data content validation mechanisms with NAS-based XML instance documents, thus resulting in higher levels of data integrity and reduced content errors.

Chapter 9. Completing the translation of OCL constraints to OWL expressions - D103

9.1. Task / Requirement(s)

Testbed-14 included a requirement for the pilot to "Extend ShapeChange to support transformation of NAS OCL Constraints into OWL, SHACL and/or SWRL (or other ontology rule language) encodings to be used with NEO." The results from this effort are documented in Section 5 of the OGC Testbed-14 Application Schema-based Ontology Development Engineering Report (OGC 18-032). The pilot found that conversion of OCL to OWL is not as simple a process as had been originally anticipated. Rather, each OCL construct must be individually analyzed and the rules for its proper translation derived. OGC 18-032 describes the translation rules for twenty- one (21) OCL constructs used in the NAS. It also identifies seven (7) OCL constructs for which translation is not possible and an explanation of why. Due to the unanticipated complexity of this effort, not all fifty-four (54) OCL constructs used by the NAS could be evaluated.

The goal of D103 within the UGAS-2019 Pilot was to complete the analysis of the translation of the remaining 26 types of OCL constraint to OWL class expressions.

9.2. Results

Translations for the remaining types of OCL constraints were developed. Annex B of the OGC Testbed-14 Application Schema-based Ontology Development Engineering Report has been revised accordingly.

NOTE	During the analysis, another opportunity for optimization (of the OCL to OWL translation) has been identified. It is documented in the revised engineering report in section 5.4.

Completion of this analysis enables enhancement of the NSG Enterprise Ontology (NEO) to achieve increased expressivity in knowledge representation and the ability to implement data integrity checks in a linked-data environment.

Chapter 10. Enhancing the ShapeChange XML input loader - D104

10.1. Task / Requirement(s)

Loading content from an SCXML document into ShapeChange ignored most input configuration parameters and elements.

A task for the UGAS-2019 Pilot therefore was to enhance ShapeChange such that loading from SCXML supports the "standard" input configuration parameters and elements; for example, input parameters such as "checkingConstraints" should be honored.

10.2. Results

The analysis conducted for D104 discovered a number of shortcomings of ShapeChange when loading a model from SCXML:

- Several ShapeChange input parameters were not recognized.
- Aliases for tagged values, defined within the input section of the ShapeChange configuration, were not taken into account.
- Descriptor sources defined in the ShapeChange configuration were ignored.
- PackageInfo configuration elements were ignored.
- Stereotypes defined in the SCXML were loaded as-is, ignoring stereotype aliases defined in the ShapeChange configuration and thus not performing stereotype normalization.

All these shortcomings have been fixed. The same model loading behavior now applies to models encoded in SCXML and in other formats, such as an Enterprise Architect repository. SCXML therefore has become a true alternative model encoding.

In addition, the following ShapeChange enhancements not explicitly required for the UGAS-2019 Pilot, but discussed within the project and deemed important and useful for the wider community, have been implemented:

- When loading a model, stereotypes not known to ShapeChange can now be loaded. Before, they were simply ignored/dropped. This enables the NAS to employ new stereotypes and ensure that they will be included in the corresponding SCXML file.
- ShapeChange target implementations now have a means to register conversion rules and standard encoding rules. This supports the implementation of ShapeChange targets by other parties, without having to change the code of ShapeChange itself. This is an important improvement regarding the extensibility of ShapeChange that will benefit NAS-associated uses of ShapeChange.

Chapter 11. Lossless exchange of NAS content using the ShapeChange XML format (SCXML) - D105

11.1. Task / Requirement(s)

SCXML serves as the representation of the NSG Application Schema (NAS) through all stages of ShapeChange-based transformation. It is essential that this representation is complete and that no information is lost during ShapeChange-based transformation. The prototypical use-case is the generation of Enterprise Architect model files based on either the full NSG Application Schema (NAS) or a subset of the NAS.

The NAS is an unusually large application schema that is designed to be technology-neutral. ShapeChange is used to prepare mission-specific subsets of the NAS as well as to transform the NAS model into related models that satisfy technology-specific requirements and constraints. Accomplishing these objectives either singularly or in combination typically requires multiple processing steps that may be separated, or check pointed, by storage of the work-in-progress model using SCXML.

The goal for D104 was to investigate the use of SCXML for the lossless representation of the NSG Application Schema (NAS) at all stages of ShapeChange-based transformation (e.g., after application of the Profiler). Where shortfalls were identified, the pilot should enhance the SCXML schema and/or SCXML-related processing.

11.2. Results

The SCXML schema as well as ShapeChange have been revised and enhanced in a number of ways:

- XML Schema (XSD) version 1.1 is now used to define the SCXML schema.
  - Co-occurrence constraints and requirements regarding SCXML elements and content are now defined using XSD constraints (key, keyref, unique) as well as XSD 1.1 assertions.
  - The SCXML schema now supports open content. This is a useful feature which supports backward and forward compatibility when versioning the SCXML schema. The feature also allows third-party extensions to be added to SCXML documents, without causing validation with the pure SCXML schema to fail.
    - More specifically, open content within an XML Schema means: For an element that is defined by the schema and that has complex content, an instance of this element may contain further elements that are not explicitly defined as content of the element - even if the additional elements have the same namespace as the target namespace of the schema - except for global elements declared by the schema. Deprecated elements that have been removed from the SCXML schema will thus be allowed in instance data.
  - ShapeChange has been extended to support XSD 1.1 based validation of SCXML instance documents.

- The SCXML schema has methodically been revised to result in a consistent structure throughout. Redundant elements such as sc:Property/sc:reversePropertyId have been removed, because it was determined that the information represented by these elements can - and should - be derived from other "more fundamental" elements when determined to be useful to an application. The open content model of the SCXML schema supports the presence of such deprecated elements within older SCXML instance documents. Thus, backwards compatibility is ensured.
- A number of compositors within the SCXML schema have been relaxed from 'sequence' to 'all'. This increases encoding flexibility, and also underlines the fact that the order of elements within such a compositor is insignificant.
- The multiplicity of some SCXML elements - such as sc:descriptors - has been relaxed (from required to optional), to support use cases in which a profile of the SCXML schema is used, and that profile does not use these elements.
- All SCXML schema elements have been documented.
- The SCXML schema has been revised so that association role ownership, as defined by UML 2.4+, can be encoded.
- The human-readable description of a model constraint (e.g. an OCL constraint) can now be encoded in an explicit XML element, and thus be separated from the actual constraint expression.

The revised SCXML schema supports lossless exchange of the NAS model. ShapeChange has been updated to read and write UML models from/to SCXML according to the new SCXML schema.

Another achievement of D105 is that the set of tagged values well-known to ShapeChange has been revised and the documentation on https://shapechange.net/app-schemas/uml-profile/#Tagged_Values has been brought up to date. The updated documentation will help ShapeChange users, including those employing the NAS, in better understanding and correctly applying the numerous tagged values that can be used within ShapeChange processes.

Chapter 12. Improving the handling of association classes by the ShapeChange Profile Management Tool (PMT) - D106

12.1. Task / Requirement(s)

In the Profile Management Tool (PMT), if a UML association class is removed from the application schema profile, the corresponding association (and roles) should be removed as well. Similar considerations apply when adding an association class. Association class management by the PMT shall be a superset of the logical union of the capabilities for management of UML classes and UML associations.

The pilot shall enhance the PMT to address each of these concerns regarding support for association classes.

12.2. Results

The Profile Management Tool has been updated to handle association classes as required. The PMT is now suited for defining profiles of application schemas such as the NAS that use association classes.

Chapter 13. Enhancing the PMT to support additional parameters - D107

13.1. Task / Requirement(s)

The Profile Management Tool (PMT) - and ShapeChange - should support additional profile parameters that were first described in the OGC Testbed-12 ShapeChange Engineering Report: isAbstract, isOrdered, and isUnique.

NOTE	Additional profile parameters to control multiplicity and navigability were already supported by the PMT.

13.2. Results

The PMT and ShapeChange have been enhanced to support the additional profile parameters.

In addition, the PMT has been updated to support the new SCXML schema. For example, the PMT displays ownership of association roles. Furthermore, the following enhancements not explicitly required for UGAS-2019, but discussed within the project and deemed important and useful for the wider community, have been realized:

- Information on isUnique, isOrdered, isComposition, and isAggregation of a property that represents an enum-literal (attribute of an <<enumeration>> class) or a code (attribute of a <<codelist>> class) is now suppressed in the Info panel, because these information elements have no semantic meaning for such literals. This ensures that extraneous or misleading information is not presented to the tool user.
- Since enum-literals typically have maximum multiplicity = 1, any GUI elements for setting the profile parameters isOrdered and isUnique for these model elements are automatically disabled. Profile parameters isOrdered and isUnique are only useful for properties that can have multiple values. This ensures that extraneous or misleading information is not presented to the tool user.
- A new global view setting has been added, which allows displaying the default values of information items such as isOrdered and isComposition, rather than not displaying these items if they are not encoded in SCXML (typically because the default value defined for the item applies). This ensures that tool users are kept aware of the default (implied by SCXML) assignments for these information items.
- The model search functionality has been extended to also search in (the textual values of) tagged values. This is important to users of the NAS where tagged values carry significant semantic content that would otherwise be ignored.

These four enhancements improve the ability for the PMT to be used by NAS Subject Matter Experts to define NAS subsets to meet their mission-specific requirem

Appendix A: Revision History

Table 1. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
Dec 10, 2019	J. Echterhoff	1.0	all	Preliminary summary report

Appendix B: Bibliography

- OGC Testbed-14: Application Schema-based Ontology Development Engineering Report (OGC 18-032), available online at http://docs.opengeospatial.org/per/18-032r2.html
- OGC Testbed-12 ShapeChange Engineering Report (OGC 16-020), available online at http://docs.opengeospatial.org/per/16-020.html