Open Geospatial Consortium, Inc.

Date: 2008-07-02

Reference number of this document: OGC 08-078r1

Version: 0.0.3

Category: Discussion Paper

Editor: Clemens Portele

OGC® OWS-5 ER: GSIP Schema Processing

Warning

This document is not an OGC Standard. This document is an OGC Discussion Paper and is therefore 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, an OGC Discussion Paper should not be referenced as required or mandatory technology in procurements.

Document type:	OpenGIS® Discussion Paper
Document subtype:	NA
Document stage:	Proposed version 0.0.3
Document language:	English

Preface

This document contains a description of the schema tailoring process for application schema development based on the U.S. National System for Geospatial Intelligence (NSG) GEOINT Structure Implementation Profile (GSIP) as developed in conjuction with the Open Geospatial Consortium Interoperability Program initiatives OWS-4 and OWS-5. In particular it discusses:

- Creation of ISO 19109 (Geographic information - Rules for application schema) conformant Application Schemas in UML from in the GSIP, known as the NSG Application Schema (NAS)

- Derivation of GML Application Schemas using the ShapeChange UML-to-GML-Application-Schema conversion tool

- Metadata describing GSIP-based application schemas to support their discovery and assessment using CSW 2.0 services based on the ebXML Registry Information Model

Suggested additions, changes, and comments on this draft report are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document.

The changes made in this document version, relative to the previous version, are tracked by Microsoft Word, and can be viewed if desired. If you choose to submit suggested changes by editing this document, please first accept all the current changes, and then make your suggested changes with change tracking on.

Forward

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.

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 Inc. 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.

Contents

Page

OGC® OWS-5 GSIP Schema Processing ER

1 Introduction

1.1 Scope

This OGC® document describes and discusses the OWS-5 enhancements in the process of creating application schemas in support of the NSG from NGA data based on the GEOINT Structure Implementation Profile (GSIP) which has been based on the NSG Application Schema and accompanying NSG Entity Catalog.

The approach used to create the application schemas starts with the creation of an ISO 19109 conformant application schema in UML from the NSG Entity Catalog. This UML model is then used as input to the ShapeChange UML to GML conversion tool deriving GML application schemas from the UML models in an automated process.

This actitiy is a continuation from OWS-4. Related information is documented in the OWS-4 GSIP Schema Processign IPR and the OWS-5 Data View Architecture ER.

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 Inc. 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.

1.2 Document contributor contact points

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

Name	Organization
Clemens Portele	interactive instruments GmbH
David G. Wesloh	NGA
Paul Birkel	MITRE
Philippe Duchesne	Ionic

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
2008-05-14	0.0.1	Clemens Portele	all	initial document based on OWS-4 IPR (07-028r1)
2008-05-15	0.0.2	Clemens Portele	all	
2008-07-02	0.0.3	Clemens Portele	all	revision based on comments from Paul Birkel

1.4 Future work

Improvements in this document are desirable to resolve the open issues. In addition, the change requests mentioned in this document should be compiled and submitted to OGC and/or ISO/TC 211.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

GEOINT Structure Implementation Profile, Version 2.5 (draft)

NOTE 1 This document can be obtained after publication from http://www.gwg.nga.mil/index.html?content=stds_regs

NOTE 2 The GEOINT Structure Implementation Profile (GSIP) includes the NSG Application Schema (NAS), the NSG Feature Concept Dictionary (NFDD) and the NSG Entity Catalog (NEC).

ISO/TS 19103:2005, *Geographic Information – Conceptual Schema Language*

NOTE 3 This document can be obtained from the International Organisation for Standardisation.

ISO 19109:2004, *Geographic Information – Rules for Application Schemas*

NOTE 4 This document can be obtained from the International Organisation for Standardisation.

ISO 19136:2007, Geographic Information – Geography Markup Language (GML)

NOTE 5 This document can be obtained from the Open Geospatial Consortium Inc. or the International Organisation for Standardisation.

ISO/TS 19139:2007, Geographic Information – Metadata – XML Schema Implementation

NOTE This document can be obtained from the International Organisation for Standardisation.

Catalogue Service 2.0.2, OGC Implementation Standard

NOTE This document can be obtained from the Open Geospatial Consortium Inc.

Catalog Service 2.0.2 ebRIM Application Profile 1.0.0, OGC Implementation Standard

NOTE This document can be obtained from the Open Geospatial Consortium Inc.

Department of Defense Discovery Metadata Specification (DDMS), Version 1.4

NOTE This document can be obtained from. https://metadata.dod.mil/mdr/irs/DDMS/index.html

DGIWG/TSMAD Profile, Profile(s) of ISO 19107 that support two-dimensional topology

NOTE This document can be obtained from https://portal.dgiwg.org/

In the event of a conflict between the text of this document and the references cited herein, the text of this document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

3 Terms and definitions

n/a

4 Conventions

4.1 Abbreviated terms

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
DFDD	DGIWG Feature Data Dictionary
DIGEST	Digital Geographic Information Exchange Standard
FACC	Feature and Attribute Coding Catalogue
GML	Geography Markup Language
GSIP	GEOINT Structure Implementation Profile
ISO	International Organization for Standardization
MSD	Mission Specific Data
NAS	NSG Application Schema
NEC	NSG Entity Catalog

NGA	National Geospatial-Intelligence Agency
NSG	National System for Geospatial-Intelligence
OGC	Open Geospatial Consortium
UML	Unified Modeling Language
WFS	Web Feature Service
XML	eXtended Markup Language

4.2 UML notation

The diagrams that appear in this document are presented using the Unified Modeling Language (UML) static structure diagram based on the rules of ISO/TS 19103 (Geographic information – Conceptual schema language) and ISO 19136 (Geographic Information – Geography Markup Language, GML 3.2.1) Annex E.

5 GSIP-based application schema development

5.1 Goals

A main goal of the application schema processing activity within OWS-5 was to advance the understanding, modeling, publication and use of complex geographic information, like the datasets managed by the National Geospatial-Intelligence Agency, NGA, within a network-centric interoperability architecture.

The approach applied within the National System for Geospatial Intelligence (NSG) is based on a model-driven architecure. Starting from a common understanding of the relevant concepts, modeling takes place in two steps. In the first step, a conceptual, platform independent model is being specified that is the basis for a variety of platform specific storage and exchange models, including GML. In a network-centric architecture these models and the data are provided via services. The published models and related dictionaries allow – in conjunction with other metadata – for an understanding of the data provided via the services. This activity – in conjunction with the activity on the data view architecture – tests the ability of OGC standards to facilitate one of the core scenarios in any spatial data infrastructure – information publication.

5.2 Components of the GSIP

The figure below shows the different layers of GSIP components (middle column) and its environment, which are discussed in more detail in the subsequent subclauses. The figure includes comparable components in related geospatial information communities, applicable ISO standards, etc.

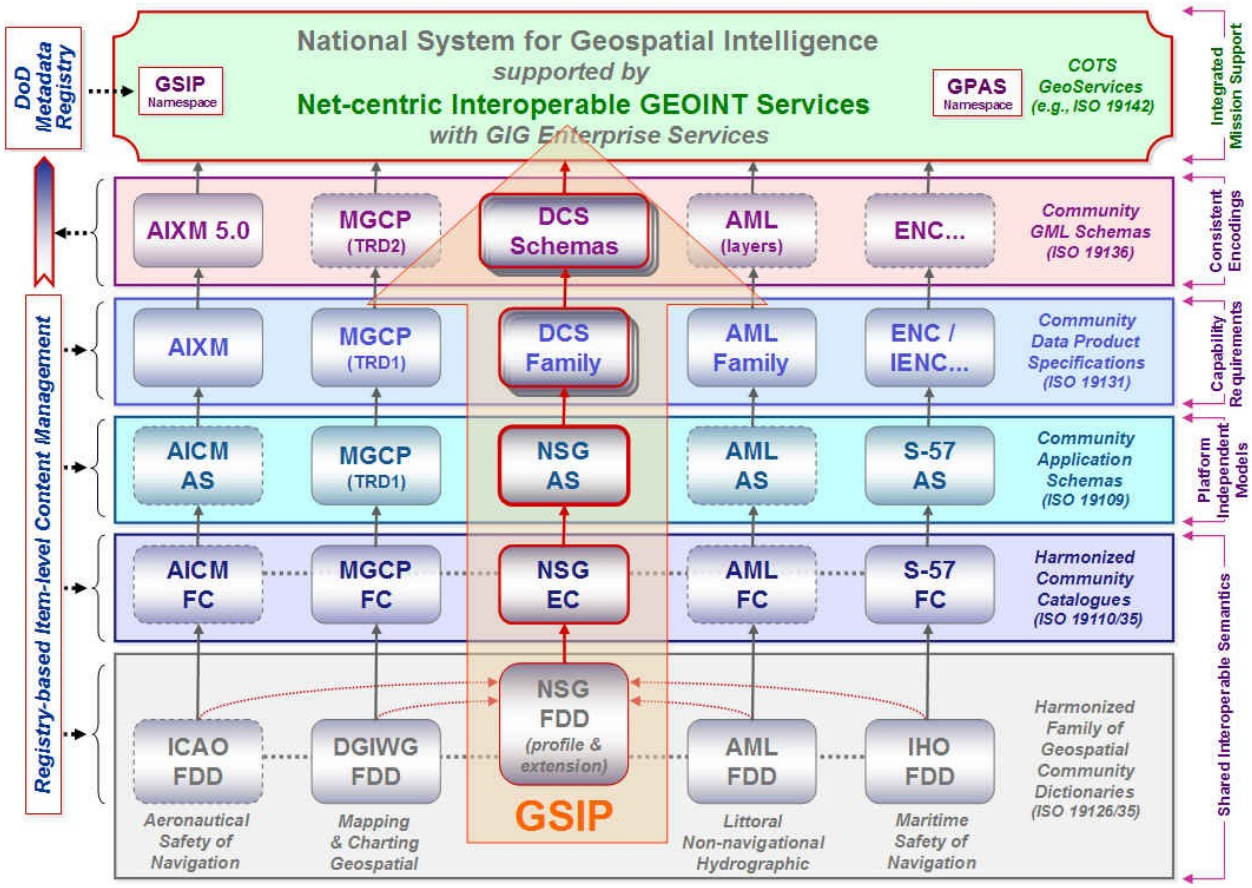


Figure 1 – Net-centric Interoperable GEOINT Services and the GSIP [from the GSIP Fact Sheet]

5.3 Feature data dictionaries

The DGIWG Feature Data Dictionary (DFDD) contains geographic information concepts used by member states of the DGIWG community to characterize aspects of features, i.e. real world phenomena. It is the successor of the Feature and Attribute Coding Catalogue (FACC), a component of the Digital Geographic Information Exchange Standard (DIGEST).

The NSG FDD is the DFDD-based feature data dictionary as specified by NGA/NCGIS. It is a subset of the DFDD without items that are not relevant for NGA, but a number of extensions have been added to represent information used by NGA and its customers; the extensions are often drawn from other data dictionaries.

The proposed ISO 19126 (Geographic information – Feature concept dictionaries) is the underlying abstract specification of the feature data dictionaries.

5.4 The NSG Entity Catalog (NEC)

The NSG Entity Catalog draws from the concepts (feature types, other information entity types, attributes types, enumerants, etc.) defined in the NSG FDD and binds them

together in an entity catalog according to ISO 19110 (Geographic information – Methodology for feature cataloguing).

Objectively, the NSG Entity Catalog will authoritatively specify the GEOINT data elements. Entity catalogs that document the data elements used in a specific community or application, e.g. the Local MSD Data Content Specification, represent implementation profiles that are strict subsets of the complete NSG Entity Catalog (e.g. elements are made mandatory, are removed or associated with other constraints within the scope f the profile).

In order to define application schemas that are an ISO 19109 conformant specification of the feature types and their properties in a formal conceptual schema language, more information is needed than typically contained in a feature catalog. Therefore, the term NSG Entity Catalog is used to denote the extended version of the former NSG Feature Catalog that contains this additional information.

5.5 The NSG Application Schema (NAS)

The NSG Application Schema (NAS) specifies the platform independent model that determines the structure used to represent the semantics specified by the NEC. It imports other existing schemas, like the conceptual schemas from ISO 19100-series of standards. The NAS itself is an ISO 19109 application schema in UML.

The NAS ensures that there is a clear, complete, and internally-consistent NSG geospatial data schema that may be used to derive system-specific implementation schemas in a rigorous manner – this ensures that data integrity is preserved when geospatial data is exchanged between different system implementations within the NSG.

5.6 Implementation specifications / encodings

From the platform-independent NAS, platform-specific representations can be derived. In OWS-5, a GML 3.2.1 application schema is derived to make NSG data available through services like the Web Feature Service.

Additional implementation representations, e.g. for ESRI Shape file, SQL database management systems, Java classes, etc. could in principle be derived as well.

5.7 Scope within OWS-5

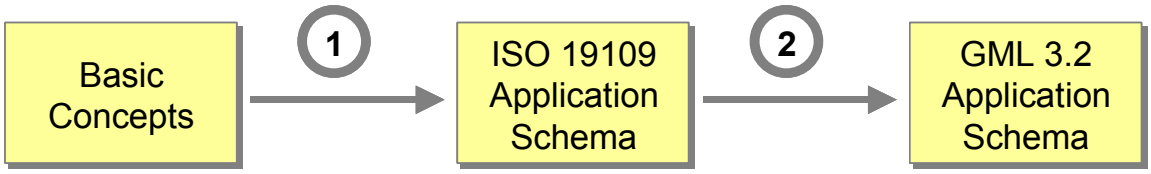
This subclause describes the scope of the work within OWS-5 – as a continuation of the activities in OWS-4 (see OGC Discussion Paper 07-028r1) – in the context of the GSIP:

- Add support for the representation of OCL constraints in the XML environment.
- Add support for the ISO/TS 19139 encoding rules.
- Develop rules for the representation of coverage information in application schemas.

- The ShapeChange UGAS tool has been enhanced as required to address all information encoded in the UML model including support for the IC-ISM XML schema, metadata elements and OCL constraints. The target GML version is GML 3.2.1.

5.8 Application schema creation process

The simplified process is described in the following figure:



Step (1) is executed using an application creating a Rational Rose or Eclipse UML model from the NAS maintained by NGA.

Step (2) is executed using the Open Source UGAS tool "ShapeChange" developed by interactive instruments (ii). More information about the tool including documentation can be found at http://www.interactive-instruments.de/ShapeChange/. The tool documentation includes one document describing the mapping rules from UML to GML as implemented by the tool plus a second document describing the implementation of the ShapeChange tool, its installation and guidelines for using the tool.

6 NSG Application Schema (NAS)

The NAS is an ISO 19109 Application Schema in UML according to the UML profile in GML 3.2.1 Annex E with the exceptions listed in this clause.

In addition, sub-clause 6.5 discusses the modelling of coverages as part of the NAS.

6.1 Use of UML 2 notation

UML notation in accordance with UML 2 is used including the placement of cardinality information for attributes (after the type name instead of after the attribute name) and the naming of all stereotypes uses lowerCamelCase (start with lower case, no blanks).

6.2 Stereotype <<bundle>>

A stereotype <<bundle>> is used for packages in an application schema that is not a leaf package. The <<bundle>> contains either only <<bundle>> packages or only <<leaf>> packages.

6.3 Additional ShapeChange-specific tagged values

In addition to the tagged values specified in GML 3.2.1 Annex E, further tagged values are used to control extensions to the encoding rules specified in GML 3.2.1. These are discussed in clause 7.

6.4 Additional NAS-specific tagged values

In addition to the tagged values specified in GML 3.2.1 Annex E and the ShapeChange extensions, NAS-specific tagged values are coded in the model. These are additional schema metadata and may be mapped, for example, to DDMS metadata elements (see OGC Discussion Paper 07-028r1). However, they are not relevant for the encoding rules discussed in this document.

6.5 Modelling coverages as part of application schemas

6.5.1 Overview

A future version of the NAS will include the modelling of coverages according to ISO 19123. Currently, ISO 19109 does not specify any rules for modelling coverages as part of application schemas (as ISO 19109 was finished before ISO 19123). Therefore, a task of OWS-5 dealt with the modelling of coverage information in application schemas and representing these application schema components in GML. In OWS-5 gridded elevation data (SRTM) was used as an example.

6.5.2 Discussion of approaches

Different approaches were considered, including the following:

- The direct approach: Use the CV-types from ISO 19123 as-is in the application schema and specify additional constraints in the documentation of the application schema.

  In a GML implementation the appropriate GML element would be used directly. Additional constraints could be encoded in Schematron constraints in the GML application schema.

  This approach has advantages and disadvantages:

  + The use of predefined GML element may make it easier to handle such coverages for software applications

  - Semantics of the coverage not really specified in the application schema

  - Range part not clearly specified (both in UML and in XML)

- Subtyping the CV-types: To improve the domain specific representation of coverages in application schemas, the direct approach could be clarified by specifying a subtype of the appropriate CV-type.

  In a GML implementation a coverage element substitutable for the appropriate GML coverage element would be used. The extension would not add any new properties.

  This approach has advantages and disadvantages:

+ The semantics of an elevation feature is more clear by specifying a specific feature type.

+ The coverage element may also serve as a hook for Schematron constraints specific to elevation features.

- Range part still not clearly specified (in particular in XML)

- Software needs to parse the schema to understand that app:ElevationGridCoverage is a grid coverage

- The conceptual elevation model relatively complex due to the use of the generic coverage model.

- Coverage functions as property values: For an elevation grid coverage it may seem plausible to equate the elevation model with a single coverage feature[1]. However, in general, this may be too simplistic. Just like a feature can have multiple properties with geometries as values, why shouldn't a feature (as an abstraction of real-world phenomena) have multiple coverage functions? A road segment may, for example, have a coverage describing the road width as a function of the distance from the start of the segment as well as a separate function describing measurements of physical properties of the pavement.

  I.e., this approach would associate a coverage function with a feature. In the application schema, the value type of a feature property would be a CV-types from ISO 19123 (potentially with additional constraints). In some cases it may be appropriate to use the coverage, which is a feature type, too, as a "standalone" feature type.

  For clarity, this approach should be clarified in an extension of the General Feature Model as specified in ISO 19109 to reflect that coverages functions should in general be values of a feature property[2]. The change could be a new <<metaclass>> GF_CoverageFunctionAttributeType as a subtype of GF_ThematicAttributeType and with a dependency to CV_Coverage.

  In addition, rules for application schemas that include coverage functions should be added to ISO 19109:

---

[1] To some extent, the "coverage is a feature" notion seems to have created an unnecessary separation between "vector features" and "coverages".

[2] This is also noted in the Observation & Measurement standard, part 1 (item b in the "Changes Required to the OpenGIS specification" clause): *'O&M describes a property-value provider model, linked to the ISO 19109 GFM, under which features are the generic carriers of properties. However, ISO 19123 provides a model for describing properties that vary with spatio-temporal location. For consistency between the GFM and the Coverage model, it appears that every coverage must be related to one or more "features" of some type that may logically carry the property whose variation is described. This may be trivial – e.g. the "medium" whose extent matches the domain-extent of the coverage (e.g. atmosphere, ocean, earth) – and may merely be described in the coverage "metadata". But it is nonetheless required to add a notion of "the feature carrying the coverage" to ISO 19123 in order to make it consistent with the GFM.'*

- *An application schema package that uses coverage functions shall follow the rules of ISO 19109 8.2.5 for referencing standardized schemas, i.e. import the coverage schema specified by ISO 19123.*

- *A coverage function shall be defined as a property of a feature type where the type of the property value is a realization of one of the types given in Table 1.*

Table 1 - List of valid coverage types in an application schema

Abstract coverage types	*Discrete coverages*	*Continuous coverages*
CV_Coverage *CV_DiscreteCoverage* *CV_ContinuousCoverage*	*CV_DiscretePointCoverage* *CV_DiscreteGridPointCoverage* *CV_DiscreteCurveCoverage* *CV_DiscreteSurfaceCoverage* *CV_DiscreteSolidCoverage*	*CV_ThiessenPolygonCoverage* *CV_ContinousQuadrilateralGridCoverage* *CV_HexagonalGridCoverage* *CV_TINCoverage* *CV_SegmentedCurveCoverage*

In a GML implementation a feature type would have a property that has a GML coverage element as its value.

Again, this approach has advantages and disadvantages:

+ Meaning of an elevation feature is exposed by specifying a specific feature type.

+ This element may also serve as a hook for Schematron constraints specific to elevation features.

+ Allows to add additional coverage functions as properties of the elevation feature (e.g. a coverage function mapping location to accuracy; this is common for bathymetry).

- Range part still not clearly specified (in particular in XML)

- Software needs to understand coverage functions as feature properties

- Implementation profile: The model from ISO 19123 intentionally contains redundancy (e.g., domain & range as well as pair representations) and very generic range values and range type descriptions. This is good from a conceptual point of view, but in an application schema context typically a more focused representation is required to remove redundancy and use a specific domain/range-value representation. GML has made a choice here, but sometimes other representations are needed to be able to constrain the range part. Thus, coverage functions in applications are probably better described by types **realising** the CV-types instead of **subtyping** them.

  In a GML implementation and if the GML3.2.1 Annex E encoding rules would be applied, the coverages would become separate feature types independent from the pre-defined coverage types. This does raise some issues not only from a GML roadmap point of view, but also from a software tools implementation point of view.

  This approach has advantages and disadvantages, too:

+ Conceptual model including the range part is more clearly specified.

+ This element may also serve as a hook for Schematron constraints specific to elevation features.

- How do software applications identify the coverage function? Or is it sufficient if this is done only for well-known feature types, i.e. application specific?

6.5.3 Approach selected

6.5.3.1 Overview

The approach selected is based on a combination of the last two approaches discussed above and has the following characteristics:

- It treats a coverage function as the value of a feature property and assumes an extension of the ISO 19109 General Feature Model as proposed above.

- It follows an approach that creates an implementation profile of the ISO 19123 types.

- It further distinguishes different encoding mechanisms.

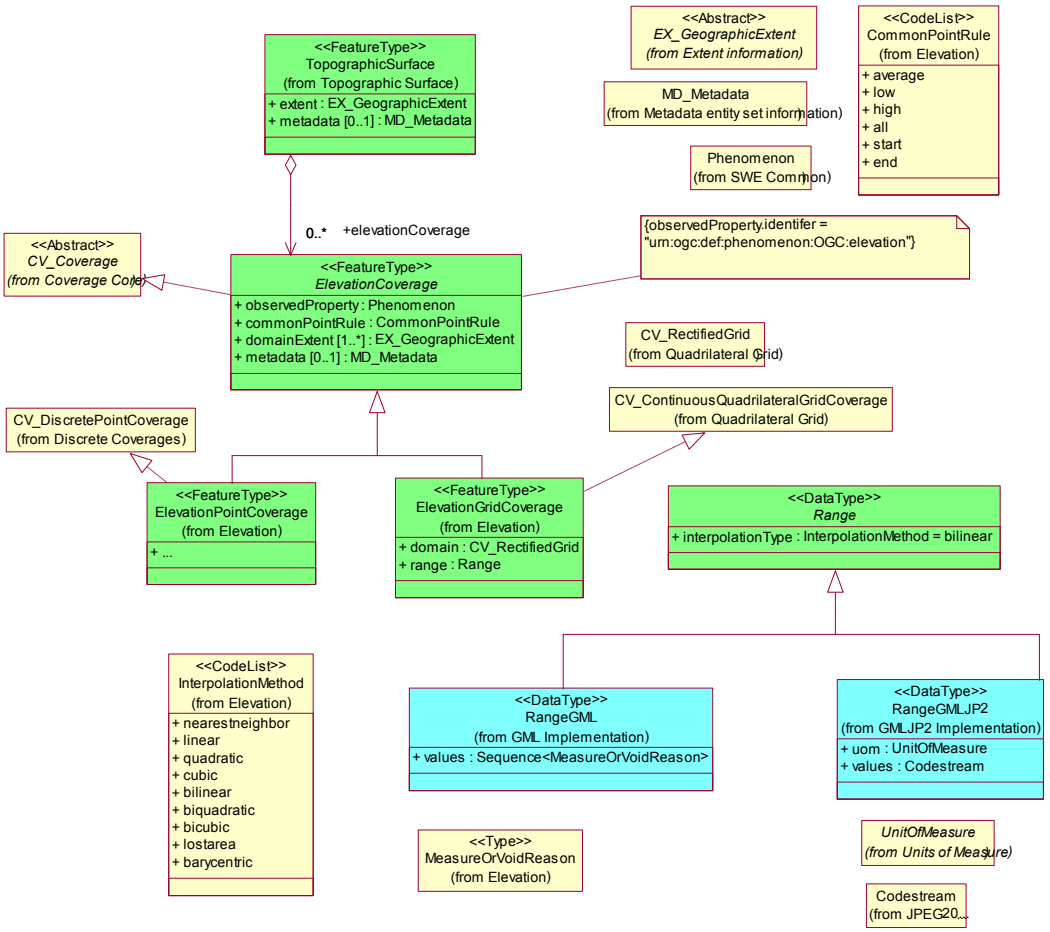
Note: The list of interpolation methods in ISO 19123 obviously differs from the one in WCS 1.1.1. In particular, "bilinear" in ISO 19123 has the same meaning as "linear" in WCS 1.1.1.

Recommendation: Propose changes to ISO 19109 (GFM and rules for coverage functions in application schemas amendment) and WCS (align code list with ISO 19123).

6.5.3.2 Application schema

The prototype application schema is illustrated in the UML class diagram below.

The feature type topographic surface has several properties, of which some are elevation coverage functions. Gridded elevation coverages are modeled in the ElevationGridCoverage type that implements CV_ContinuousQuadrilateralGridCoverage specified in ISO 19123 using a domain property (always a rectified grid) and a range (for example, a JPEG 2000 codestream with some metadata about the interpretation of the values in the codestream).



6.5.3.3 GML application schema

The resulting GML application schema derived from this prototype application schema using the GML 3.2.1 Annex E encoding rules would be:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:cov="http://www.opengis.net/ows5/ics/elevation/test"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/1.0" elementFormDefault="qualified"
targetNamespace="http://www.opengis.net/ows5/ics/elevation/test" version="0.2">
  <import namespace="http://www.opengis.net/swe/1.0"
schemaLocation="http://schemas.opengis.net/sweCommon/1.0.0/swe.xsd"/>
  <import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="http://schemas.opengis.net/gml/3.2.1/gml.xsd"/>
  <import namespace="http://www.isotc211.org/2005/gmd"
schemaLocation="http://schemas.opengis.net/iso/19139/20070417/gmd/gmd.xsd"/>
  <!--XML Schema document created by ShapeChange-->
  <element name="ElevationGridCoverage" substitutionGroup="cov:ElevationCoverage"
type="cov:ElevationGridCoverageType"/>
  <complexType name="ElevationGridCoverageType">
    <complexContent>
      <extension base="cov:ElevationCoverageType">
        <sequence>
          <element name="domain">
            <complexType>
              <sequence minOccurs="0">
```

```
              <element ref="gml:RectifiedGrid"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
          </complexType>
        </element>
        <element name="range" type="cov:RangePropertyType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="ElevationGridCoveragePropertyType">
  <sequence minOccurs="0">
    <element ref="cov:ElevationGridCoverage"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
<element name="TopographicSurface" substitutionGroup="gml:AbstractFeature"
type="cov:TopographicSurfaceType"/>
<complexType name="TopographicSurfaceType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="extent">
          <complexType>
            <complexContent>
              <extension base="gml:AbstractMetadataPropertyType">
                <sequence minOccurs="0">
                  <element ref="gmd:AbstractEX_GeographicExtent"/>
                </sequence>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
        <element minOccurs="0" name="metadata">
          <complexType>
            <complexContent>
              <extension base="gml:AbstractMetadataPropertyType">
                <sequence minOccurs="0">
                  <element ref="gmd:MD_Metadata"/>
                </sequence>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
        <element maxOccurs="unbounded" minOccurs="0" name="elevationGridCoverage"
type="cov:ElevationGridCoveragePropertyType"/>
        <element maxOccurs="unbounded" minOccurs="0" name="elevationPointCoverage"
type="cov:ElevationPointCoveragePropertyType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="TopographicSurfacePropertyType">
  <sequence minOccurs="0">
    <element ref="cov:TopographicSurface"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
<element abstract="true" name="ElevationCoverage"
substitutionGroup="gml:AbstractFeature" type="cov:ElevationCoverageType"/>
<complexType abstract="true" name="ElevationCoverageType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="observedProperty" type="swe:PhenomenonPropertyType"/>
        <element name="commonPointRule" type="gml:CodeType"/>
        <element maxOccurs="unbounded" name="domainExtent">
```

```
            <complexType>
              <complexContent>
                <extension base="gml:AbstractMetadataPropertyType">
                  <sequence minOccurs="0">
                    <element ref="gmd:AbstractEX_GeographicExtent"/>
                  </sequence>
                  <attributeGroup ref="gml:AssociationAttributeGroup"/>
                </extension>
              </complexContent>
            </complexType>
          </element>
          <element minOccurs="0" name="metadata">
            <complexType>
              <complexContent>
                <extension base="gml:AbstractMetadataPropertyType">
                  <sequence minOccurs="0">
                    <element ref="gmd:MD_Metadata"/>
                  </sequence>
                  <attributeGroup ref="gml:AssociationAttributeGroup"/>
                </extension>
              </complexContent>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="ElevationCoveragePropertyType">
    <sequence minOccurs="0">
      <element ref="cov:ElevationCoverage"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
    <attributeGroup ref="gml:OwnershipAttributeGroup"/>
  </complexType>
  <element name="ElevationPointCoverage" substitutionGroup="cov:ElevationCoverage"
type="cov:ElevationPointCoverageType"/>
  <complexType name="ElevationPointCoverageType">
    <complexContent>
      <extension base="cov:ElevationCoverageType">
        <sequence/>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="ElevationPointCoveragePropertyType">
    <sequence minOccurs="0">
      <element ref="cov:ElevationPointCoverage"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
    <attributeGroup ref="gml:OwnershipAttributeGroup"/>
  </complexType>
  <element name="ElevationValue" substitutionGroup="gml:AbstractObject"
type="cov:ElevationValueType"/>
  <complexType name="ElevationValueType">
    <sequence>
      <element name="value" type="gml:MeasureType"/>
    </sequence>
  </complexType>
  <complexType name="ElevationValuePropertyType">
    <sequence>
      <element ref="cov:ElevationValue"/>
    </sequence>
  </complexType>
  <element abstract="true" name="Range" substitutionGroup="gml:AbstractObject"
type="cov:RangeType"/>
  <complexType abstract="true" name="RangeType">
    <sequence>
      <element default="bilinear" name="interpolationType" type="gml:CodeType"/>
    </sequence>
  </complexType>
  <complexType name="RangePropertyType">
    <sequence>
      <element ref="cov:Range"/>
```

```
    </sequence>
  </complexType>
  <element name="RangeGMLJP2" substitutionGroup="cov:Range" type="cov:RangeGMLJP2Type"/>
  <complexType name="RangeGMLJP2Type">
    <complexContent>
      <extension base="cov:RangeType">
        <sequence>
          <element name="uom" type="gml:UnitOfMeasureType"/>
          <element name="values" type="gml:ReferenceType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="RangeGMLJP2PropertyType">
    <sequence>
      <element ref="cov:RangeGMLJP2"/>
    </sequence>
  </complexType>
  <element name="RangeGML" substitutionGroup="cov:Range" type="cov:RangeGMLType"/>
  <complexType name="RangeGMLType">
    <complexContent>
      <extension base="cov:RangeType">
        <sequence>
          <element name="values" type="gml:MeasureOrNilReasonListType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="RangeGMLPropertyType">
    <sequence>
      <element ref="cov:RangeGML"/>
    </sequence>
  </complexType>
</schema>
```

As discussed above, this application does **not** use the GML coverage types, but application specific types that encode the elevation coverage more directly.

6.5.3.4 GML instance

An example of a GML instance as part of a JPEG2000 instance could be:

```
<TopographicSurface xmlns="http://www.opengis.net/ows5/ics/elevation/test"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:gmd="http://www.isotc211.org/2005/gmd" gml:id="_1"
xsi:schemaLocation="http://www.opengis.net/ows5/ics/elevation/test elevation.xsd">
   <extent xlink:href="md.xml#extent"/>
   <elevationGridCoverage>
      <ElevationGridCoverage gml:id="_2">
         <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:elevation"/>
         <commonPointRule>average</commonPointRule>
         <domainExtent xlink:href="md.xml#extent"/>
         <domain>
            <gml:RectifiedGrid gml:id="_3" dimension="2">
               <gml:limits>
                  <gml:GridEnvelope>
                     <gml:low>0 0</gml:low>
                     <gml:high>10000 10000</gml:high>
                  </gml:GridEnvelope>
               </gml:limits>
               <gml:axisLabels>u v</gml:axisLabels>
               <gml:origin>
                  <gml:Point gml:id="_4" srsName="urn:ogc:def:crs:EPSG::4326">
                     <gml:pos>0 0</gml:pos>
                  </gml:Point>
               </gml:origin>
               <gml:offsetVector>0 0.001</gml:offsetVector>
               <gml:offsetVector>0.001 0</gml:offsetVector>
```

```
            </gml:RectifiedGrid>
         </domain>
         <range>
            <RangeGMLJP2>
               <interpolationType>bilinear</interpolationType>
               <uom uom="m"/>
               <values xlink:href="gmljp2://codestream/0"/>
            </RangeGMLJP2>
         </range>
      </ElevationGridCoverage>
   </elevationGridCoverage>
</TopographicSurface>
```

7 GML Application Schema

7.1 Overview

The GML Application Schema is automatically derived from the UML application schema using the ShapeChange UGAS (UML-to-GML-Application-Schema conversion) tool.

7.2 Using the ShapeChange command line interface

After exporting the UML model as an XMI 1.0 file, e.g., "nas_v1.8.2.xml", the ShapeChange tool is executed with the following parameters:

```
java -Xms512m -Xmx1424m -jar ShapeChange.jar -A "NAS" -D "TYPE" -
v "3.2" -o "NAS" "nas_v1.8.2.xml"
```

See the ShapeChange documentation (available at http://www.interactive-instruments.de/ShapeChange/) for details about the parameters.

As a result, the GML application schema and its XML Schema documents are created (as well as GML dictionaries for the definitions in the application schema).

In this process ShapeChange may report some warnings which refer to model elements from the ISO 19100 model and not from the NSG Application Schema.

The GML application schema should be verified after creation with appropriate tools, e.g. Xerces or XSV.

For example, using a local installation of XSV, the validity of nas.xsd can be examined using the following command (the result is documented in nas.log):

xsv.exe -i -o nas.log nas.xsd

7.3 Additional encoding rules

7.3.1 General remarks

This subclause specifies additional encoding rules in addition to GML Annex E.

7.3.2 xsdEncodingRule

To distinguish which encoding rules applies to a package or classifier in the UML model, a new tagged value “xsdEncodingRule” is analysed by ShapeChange. The known values are:

- iso19136_2007: the GML 3.2.1 Annex E encoding rules (default value)
- iso19139_2007: the ISO/TS 19139 encoding rules, see 7.3.8
- iso19136_2007_ShapeChange_1.0_Extensions: the GML 3.2.1 Annex E encoding rules with extensions as specified in this document
- notEncoded: the package or classifier is not encoded

7.3.3 xsdAsAttribute

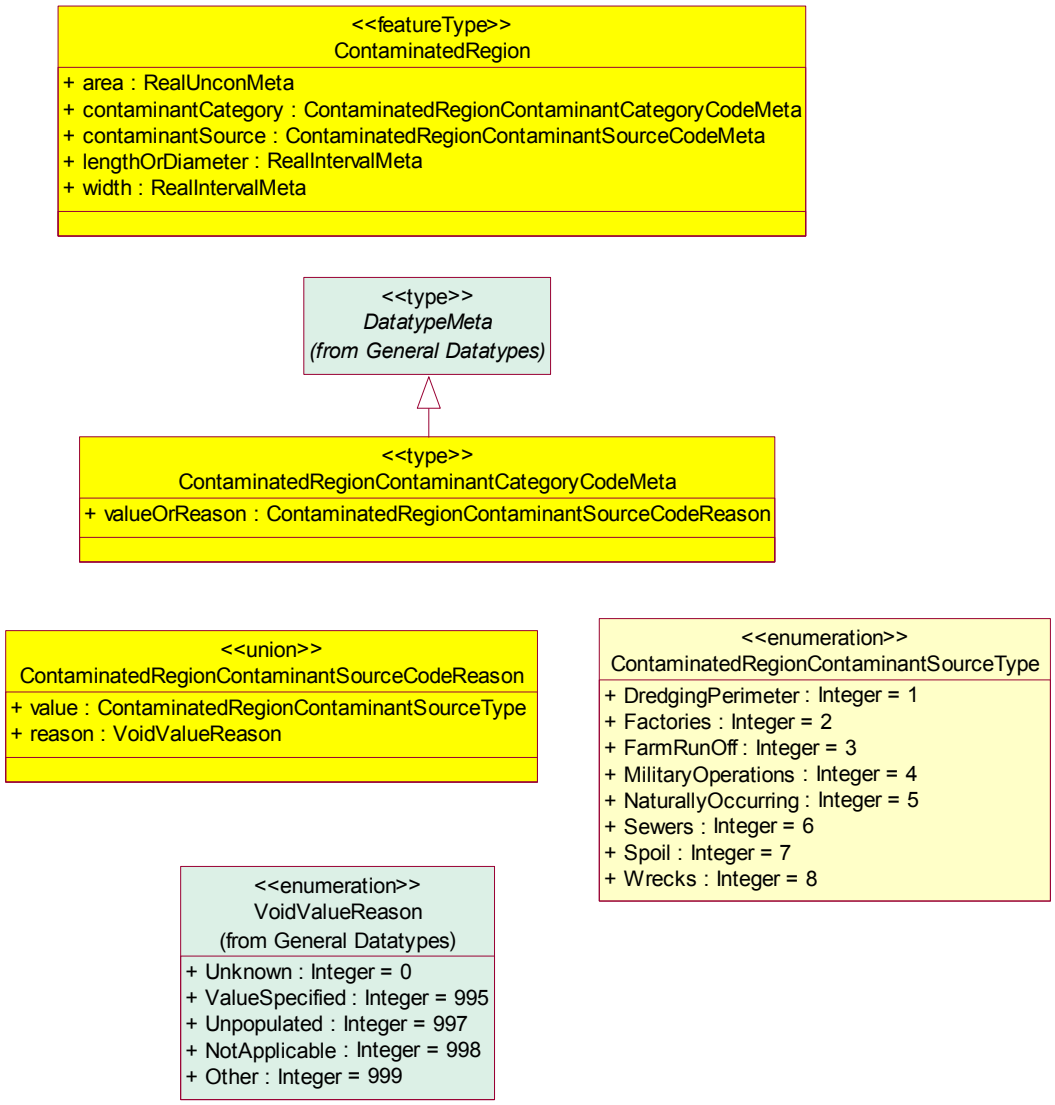
If a UML attribute has a tagged value "xsdAsAttribute" with value "true", has a maximum multiplicity of 1 and is simple values, the UML attribute is represented in XML as an XML attribute.

7.3.4 nillable, nilReasonAllowed and implementedByNilReason

If a UML attribute has a tagged value "nillable" with value "true", the corresponding XML property element would be defined with an XML attribute "nillable" set to "true".

If a UML type has a tagged value "nilReasonAllowed" with value "true", all corresponding XML property types for this property would be defined with an optional XML attribute nilReason as specified by GML.

If a property of the conceptual model is implemented by the nilReason concept of GML, the tagged value “implementedByNilReason” is set. As a result, the following classes



are encoded in the GML application schema as

```
  <element name="ContaminatedRegion" substitutionGroup="nas:FeatureEntity"
    type="nas:ContaminatedRegionType"/>

  <complexType name="ContaminatedRegionType">
    <annotation>
      <documentation>Contaminated Region: A region whose prevailing natural conditions
have been degraded through contamination by harmful or objectionable substances. [desc]
The contamination may be either naturally occurring or the result of human activity. For
example, polluted by sewage or toxic chemicals, obscured by smoke or ash from volcanic
eruptions, or contaminated by exposure to Chemical, Biological, Radiological, and/or
Nuclear (CBRN) agents. [constraint] The associated geometry is either a point
representing the centre of, or a surface representing the extent of, this contaminated
region.</documentation>
    </annotation>
    <complexContent>
      <extension base="nas:FeatureEntityType">
        <sequence>
          <element minOccurs="0" name="area" type="nas:RealUnconMetaPropertyType">
```

```
            <annotation>
              <documentation>Area: The area within the delineation of the
feature.</documentation>
            </annotation>
          </element>
          <element minOccurs="0" name="contaminantCategory"
             type="nas:ContaminatedRegionContaminantCategoryCodeMetaPropertyType">
            <annotation>
              <documentation>Contaminant Category: The category(ies) of contaminants
present in a region.</documentation>
            </annotation>
          </element>
          <element minOccurs="0" name="contaminantSource"
             type="nas:ContaminatedRegionContaminantSourceCodeMetaPropertyType">
            <annotation>
              <documentation>Contaminant Source: The source(s) of contaminants present in
a region.</documentation>
            </annotation>
          </element>
          <element minOccurs="0" name="lengthOrDiameter"
             type="nas:RealIntervalMetaPropertyType">
            <annotation>
              <documentation>Length or Diameter: The dimension of a feature taken along
its primary alignment and generally in the horizontal plane. [desc] The primary alignment
of a feature is its established direction of flow or use (for example: a road, a power
line, a river, a rapid, and/or a bridge). A feature-specific rule may apply. In the case
of a bridge, the length is the distance between the bridge abutments along the bridge
centreline. In the case of a dam, the length is the distance along the dam crest. If no
established direction of flow or use exists then (1) if the feature is irregular in shape
its length is its greatest horizontal dimension (see Attribute: 'Greatest Horizontal
Extent'), else (2) if the feature is regular in shape then a shape-specific rule may
apply: for a rectangular feature, the length of the longer axis; for a round feature, the
diameter.</documentation>
            </annotation>
          </element>
          <element minOccurs="0" name="width" type="nas:RealIntervalMetaPropertyType">
            <annotation>
              <documentation>Width: The dimension of a feature taken perpendicular to its
primary alignment and generally in the horizontal plane. [desc] The primary alignment of
a feature is its established direction of flow or use (for example: a road, a power line
right-of-way, a river, rapid, and/or a bridge). A feature-specific rule may apply. In the
case of a bridge, the width is the distance perpendicular to the bridge centre-line and
generally in the horizontal plane. In the case of a dam, the width is the distance
perpendicular to (across the) the dam crest. If no such direction of flow or use exists
then (1) if the feature is irregular in shape its width is taken perpendicular to the
direction of its greatest horizontal dimension (see Attribute: 'Greatest Horizontal
Extent'), else (2) if the feature is regular in shape then a shape-specific rule may
apply: for a rectangular feature, the length of the shorter axis; for a round feature,
the diameter.</documentation>
            </annotation>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

  <element name="ContaminatedRegionContaminantCategoryCodeMeta"
     substitutionGroup="nas:DatatypeMeta"
     type="nas:ContaminatedRegionContaminantCategoryCodeMetaType"/>

  <complexType name="ContaminatedRegionContaminantCategoryCodeMetaType">
    <annotation>
      <documentation>Contaminated Region Contaminant Category Code or Reason; with
Metadata: A coded domain value denoting the contaminant category type of a contaminated
region, accompanied by the reason that the value may be absent and associated
metadata.</documentation>
    </annotation>
    <complexContent>
      <extension base="nas:DatatypeMetaType">
        <sequence>
          <element name="valueOrReason" nillable="true">
            <annotation>
```

```
              <documentation>Contaminated Region Contaminant Category Code Value: A
contaminated region contaminant category code value.</documentation>
            </annotation>
            <complexType>
              <simpleContent>
                <extension base="nas:ContaminatedRegionContaminantCategoryTypeType">
                  <attribute name="nilReason" type="gml:NilReasonType"/>
                </extension>
              </simpleContent>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

  <complexType name="ContaminatedRegionContaminantCategoryCodeMetaPropertyType">
    <sequence>
      <element ref="nas:ContaminatedRegionContaminantCategoryCodeMeta"/>
    </sequence>
  </complexType>

  <simpleType name="ContaminatedRegionContaminantCategoryTypeType">
    <annotation>
      <documentation>Contaminated Region Contaminant Category Type: A coded domain value
denoting the contaminant category type of a contaminated region.</documentation>
    </annotation>
    <restriction base="string">
      <enumeration value="1">
        <annotation>
          <documentation>Biological: Disease-causing organisms (pathogens), toxins, or
other agents of biological origin (ABO) intended to: incapacitate, injure, or kill humans
and animals; to destroy crops; to weaken resistance to attack; and to reduce the will to
fight.  [desc] A biological agent is a microorganism that causes disease in personnel,
plants, or animals, or cause the deterioration of material.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="2">
        <annotation>
          <documentation>Chemical: The deposit, absorption, or adsorption of chemical
agents on or by structures, areas, personnel, or objects. [desc] A chemical agent is a
substance that is intended to kill, seriously injure, or incapacitate through its
physiological effects.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="3">
        <annotation>
          <documentation>Nuclear and/or Radiological: The emission of radiation, either
directly from unstable atomic nuclei or as a consequence of a nuclear reaction. [desc]
Radioactive contamination is typically the result of a loss of control of radioactive
materials during the production or use of radioisotopes. This includes nuclear fallout
(the distribution of radioactive contamination by a nuclear explosion). Radiological
weapons (&#147;dirty bombs&#148;) use conventional explosives to scatter powdered
radioactive material over the area around the bomb&#146;s explosion.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="4">
        <annotation>
          <documentation>Thermal: The process of contamination by a rapid change in
temperature. [desc] For example, the dumping of hot water into a normally cooler body of
water (or vice versa) or the effect of steam pipes on the temperature of the surrounding
environment (for example: frozen soil that thaws).</documentation>
        </annotation>
      </enumeration>
    </restriction>
  </simpleType>

  <simpleType name="ContaminatedRegionContaminantSourceTypeType">
    <annotation>
      <documentation>Contaminated Region Contaminant Source Type: A coded domain value
denoting the contaminant source type of a contaminated region.</documentation>
    </annotation>
```

```
    <restriction base="string">
      <enumeration value="1">
        <annotation>
          <documentation>Dredging Perimeter: The boundary of an area on the waterbody
bottom (for example: a channel) that has been deepened by dredging.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="2">
        <annotation>
          <documentation>Factories: Locations (for example: buildings) where goods are
manufactured. [desc] Industrial pollutants may be in the form of liquids, gases, and/or
solids.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="3">
        <annotation>
          <documentation>Farm Run-off: The release of pollutants (for example: nitrogen,
phosphorus, sediment, and fecal matter) into waterways from farming and related
agricultural activities.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="4">
        <annotation>
          <documentation>Military Operations: All aspects of military operations
involving the employment of lethal and incapacitating munitions and/or
agents.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="5">
        <annotation>
          <documentation>Naturally Occurring: Naturally occurring pollution (for example:
forest fires and volcanic eruptions) that cause significant deterioration in
environmental quality.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="6">
        <annotation>
          <documentation>Sewers: Artificial channels or conduits, usually covered and
buried, for carrying off and discharging waste, storm water, and/or refuse from buildings
and built-up areas.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="7">
        <annotation>
          <documentation>Spoil: Dredged material that has been deposited on the waterbody
bottom.</documentation>
        </annotation>
      </enumeration>
      <enumeration value="8">
        <annotation>
          <documentation>Wrecks: The ruined remains of a stranded or sunken vessel that
has been rendered useless but continues to leak fluids (for example: fuel
oil).</documentation>
        </annotation>
      </enumeration>
    </restriction>
  </simpleType>
```

7.3.5 asGroup

If a <<union>> class has a tagged value “asGroup” with a value “true” then it is encoded as an XML global group which is referenced wherever a property is defined that has the union class as its value. Note that this is only valid if from the context it is clear how to map the individual values to the conceptual model.

7.3.6 Mixin classes

Due to the fact that several implementation platforms including XML Schema supports only type derivation from a single base type (element substitutablity in XML Schema is restricted to a single element, too), the use of multiple inheritance is currently not supported by GML 3.2.1 Annex E.

However, for conceptual modelling, the ability to define abstract types which capture a set of properties that are associated with a concept is sometimes very convenient.

The following additional rules for such abstract types are therefore supported by ShapeChange:

If a class is a specialization of another class, then this class shall have one of the stereotypes <<featureType>>, <<dataType>>, no stereotype or <<type>>.

The class shall have zero or one supertype with the same stereotype and zero or more abstract supertypes of the stereotype <<type>>.

I.e., disregarding classes with stereotype <<type>>, a generalization relationship shall be specified only between two classes that are either:

- *both feature types (stereotype <<featureType>>),*
- *both object types (no stereotype), or*
- *both data types (stereotype <<dataType>>).*

For every class <<type>> all direct or indirect subtypes shall be either

- *all feature or object types (stereotypes <<featureType>>, no stereotype or <<type>>),*
- *all data types (stereotypes <<dataType>> or <<type>>).*

All generalization relationships between classes shall have no stereotype. The discriminator property of the UML generalization shall be blank.

The abstract mixin class (example in the NAS: GeometryInfo) is encoded as a group with all properties (attributes and navigable association ends) encoded as usual. This group will be referenced from the subtype.

EXAMPLE GeometryInfo and PointGeometryInfo:

```
  <group name="GeometryInfoGroup">
    <annotation>
      <documentation>Geometry Information: An abstract modeling entity serving as a
superclass that collects shared properties (attributes and associations) of modeling
entities that specify geometric representation information about a feature. [desc] For
example, the horizontal and/or vertical metadata, notes, and/or restriction(s) and/or
security control(s) applicable to dissemination of data regarding the geometric
representation of the feature. [constraint] There exists an associated: Event Entity or
Feature Entity</documentation>
```

```
      <appinfo>
        <sc:taggedValue tag="primaryCode">GeometryInfo</sc:taggedValue>
        <sc:taggedValue tag="secondaryCode">ZI029</sc:taggedValue>
        <sc:taggedValue tag="oclExpressions">inv: self.eventEntity-&gt;notEmpty() or
self.featureEntity-&gt;notEmpty()</sc:taggedValue>
      </appinfo>
    </annotation>
    <sequence>
      <element maxOccurs="unbounded" minOccurs="0" name="eventEntity"
       type="gml:ReferenceType">
        <annotation>
          <documentation>Geometry of Event Entity: An event for which this geometry
representation applies.</documentation>
          <appinfo>
            <targetElement xmlns="http://www.opengis.net/gml/3.2">
             gsip:EventEntity
            </targetElement>
            <reversePropertyName xmlns="http://www.opengis.net/gml/3.2">
             gsip:geometry
            </reversePropertyName>
            <sc:taggedValue tag="primaryCode">eventEntity</sc:taggedValue>
          </appinfo>
        </annotation>
      </element>
      <element maxOccurs="unbounded" minOccurs="0" name="featureEntity"
       type="gml:ReferenceType">
        <annotation>
          <documentation>Geometry of Feature Entity: A feature entity for which this
geometry representation applies.</documentation>
          <appinfo>
            <targetElement xmlns="http://www.opengis.net/gml/3.2">
             gsip:FeatureEntity
            </targetElement>
            <reversePropertyName xmlns="http://www.opengis.net/gml/3.2">
             gsip:geometry
            </reversePropertyName>
            <sc:taggedValue tag="primaryCode">featureEntity</sc:taggedValue>
          </appinfo>
        </annotation>
      </element>
      <element name="horizontalCoordMetadata" type="gml:ReferenceType">
        <annotation>
          <documentation>Horizontal Coordinate Metadata: The horizontal coordinate
metadata of this geometry.</documentation>
          <appinfo>
            <targetElement xmlns="http://www.opengis.net/gml/3.2">
             gsip:HorizCoordMetadata
            </targetElement>
            <reversePropertyName xmlns="http://www.opengis.net/gml/3.2">
             gsip:geometryInfo
            </reversePropertyName>
            <sc:taggedValue tag="primaryCode">horizontalCoordMetadata</sc:taggedValue>
          </appinfo>
        </annotation>
      </element>
      <!-- ... -->
    </sequence>
  </group>

  <element name="PointGeometryInfo" substitutionGroup="gml:Point"
   type="gsip:PointGeometryInfoType"/>
  <complexType name="PointGeometryInfoType">
    <annotation>
      <documentation>Point Geometry Information: A modeling entity collecting geometric
representation information about a feature that is modeled as a spatial point. [desc] A
spatial point is a 0-dimensional geometric primitive, representing a
position.</documentation>
      <appinfo>
        <sc:taggedValue tag="primaryCode">PointGeometryInfo</sc:taggedValue>
        <sc:taggedValue tag="secondaryCode">ZI007</sc:taggedValue>
        <sc:taggedValue tag="oclExpressions">
        </sc:taggedValue>
```

```
      </appinfo>
    </annotation>
    <complexContent>
      <extension base="gml:PointType">
        <sequence>
          <group ref="gsip:GeometryInfoGroup"/>
          <!-- ... -->
        </sequence>
      </extension>
    </complexContent>
  </complexType>
```

7.3.7 OCL Constraints

7.3.7.1 Overview

The tagged value “oclExpressions” on types allows capture of additional constraints related to the feature type using OCL. These constraints are typically also shown in the class diagram showing the feature type.

The tagged value “schPatterns” on types is a placeholder for a translation of the “oclExpressions” to Schematron in case the oclExpression cannot be converted automatically to Schematron.

On the GML application schema level Schematron (ISO/IEC 19757-3:2006[3]) is used in most cases as the target language for constraints. Schematron is already used by GML to express constraints that cannot be represented in XML Schema. It is currently considered the most appropriate language to express constraints on the XML level. Tools exist to process Schematron constraints and assert the complicance of an instance document with the specified constraints[4].

It is not feasible to provide a general, full mapping between OCL and Schematron. Thus, the work in OWS-5 focused on typical constraint patterns used in the NAS, for which a representation in the GML application schema has been identified. These patterns have been implemented as part of the extended encodinging rules in ShapeChange.

For restricting the values of types with simple content, xsd:restriction and the appropriate facets are used where possible.

It must be noted that Schematron is based on XPath expressions. This has the effect that Schematron constraints are in practice limited to a single document[5]; in addition, Schematron is not Xlink-aware (without support by additional Xpath functions).

ShapeChange creates one Schematron file per GML application schema in the same directory as the root schema document of the application schema.

---

[3] http://standards.iso.org/ittf/PubliclyAvailableStandards/c040833_ISO_IEC_19757-3_2006(E).zip

[4] for example: http://www.schematron.com/implementation.html

[5] ISO/IEC 19757-4 Namespace-based Validation Dispatching Language (NVDL) might be used in the future to overcome this constraint.

Constraints that cannot be processed by ShapeChange are reported in the log file.

Following are a sample of the various types of OCL statements currently included in the NAS, preceded by an "English equivalent" and followed by a translation into Schematron or XML Schema. Some constraints have some additional discussion about other representations, in particular directly in XML Schema.

7.3.7.2 Constraints on property values

These constraints could in principle also be represented directly in XML Schema by using an anonymous property type for the property element. However, this does not work, if the constraint is specified on a type but the original property is specified on a supertype. To ensure a common, general approach, OCL and Schematron will be used for these cases.

7.3.7.2.1 Constraints on enumeration values

For single-valued properties:

OCL	Schematron
inv: self.<ATT>.value = #<VAL> {or self.<ATT>.value = #<VAL>}*	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="<QUALIFIED_ATT>='<VAL>' {or <QUALIFIED_ATT>='<VAL>'}*"> ...text from constraint column goes here... </sch:assert> </sch:rule>

For multi-valued properties:

OCL	Schematron
inv: self.<ATT>.values->forAll(elem = #<VAL1> {or elem = #<VAL>}*) and self.<ATT>->isUnique (values.elements)	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="(<QUALIFIED_ATT>='<VAL>' {or <QUALIFIED_ATT>='<VAL>'}*) and count(<QUALIFIED_ATT>[.=preceding- sibling::<QUALIFIED_ATT>])=0"> ...text from constraint column goes here... </sch:assert> </sch:rule>

7.3.7.2.2 Constraints on the value or void reason of a property

OCL	Schematron
inv: self.<ATT>.value = <VAL>	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="<QUALIFIED_ATT>=<MAPPEDVAL>"> ...text from constraint column goes here... </sch:assert> </sch:rule> where <MAPPED_VAL> = 'false()' / 'true()' for <VAL> = 'false' / 'true' (other predefined value mappings may be configured).

OCL	Schematron
inv: self.<ATT>.reason = <VAL>	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="<QUALIFIED_ATT>/@nilReason=<MAPPED_VAL> and <QUALIFIED_ATT>/@xsi:nil='true'"> ...text from constraint column goes here... </sch:assert> </sch:rule> where <MAPPED_VAL> = 'notApplicable' etc. for <VAL> = '#NotApplicable', etc. (other predefined value mappings may be configured).

7.3.7.2.3 Constraints on the value of (complex) data types

An example is a constraint that states that in a collection of obstruction heights no two members shall have the same value of their datum. Currently no such constraints are specified, but can be expected in the future. These cases are not covered in OWS-5 (and not yet implemented in ShapeChange). In OCL this constraint could be stated as, for example:

inv: self.obstructionHeight->isUnique(datum.name)

The value of obstructionHeight is a collection of height-with-accuracy-and-datum values and the datums are identified by a name property.

7.3.7.3 Constraints on the type of a property

OCL	Schematron
inv: self.<ATT>->forAll(g | g.oclIsKindOf(<TYPE>) {or g.oclIsKindOf(<TYPE>)}*)	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="count(<QUALIFIED_ATT>/*)= count(<QUALIFIED_ATT>/<QUALIFIED_TYPE>) {+ count(<QUALIFIED_ATT>/<QUALIFIED_TYPE>)}*"> ...text from constraint column goes here... </sch:assert> </sch:rule> This has the potential issue that no subtypes may be used, only the explicitly listed type. Support for subtypes would require Xpath2, but Schematron is based on Xpath.

7.3.7.4 Constraints on a property value of an associated object

OCL	Schematron
inv: self.<ATT>->forAll(w | w.<ATT2>.value = <VAL> {or w.<ATT2>.value = <VAL>}*)	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="valueOf(<QUALIFIED_ATT>)/<QUALIFIED_ATT2>=<VAL> {or valueOf(<QUALIFIED_ATT>)/<QUALIFIED_ATT2>=<VAL>}*">

	...text from constraint column goes here... </sch:assert> </sch:rule> which uses the valueOf() Xpath function from the current WFS 2.0 drafts; valueOf() supports resolution of references.

7.3.7.5 Constraints on the existance of a property value

OCL	Schematron
inv: self.<ATT>->notEmpty() {or self.building->notEmpty()}*	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="count(<QUALIFIED_ATT>) {+count(<QUALIFIED_ATT>)}* >0"> ...text from constraint column goes here... </sch:assert> </sch:rule> Similarily "empty()" would be mapped to "count(...)=0".
inv: count(self.<ATT>) <op> <N>	<sch:rule context="<QUALIFIED_FEATURETYPENAME>"> <sch:assert test="count(<QUALIFIED_ATT>) <op> <N>"> ...text from constraint column goes here... </sch:assert> </sch:rule> where <op> may be, for example, '='.

7.3.7.6 Constraints on the value domain of a property

Note that the following constraints are not represented as such in the NAS version 2.5, but commonly occur; they will be integrated in a subsequent version of the NAS. In the NAS version 2.5 some of them are represented in specific tagged values which are mapped to types with xsd:restrictions and restricting facets. The table below shows the OCL representations and the restricting facets.

OCL	XML Schema - restricting facets
inv: self <op> <N> (on a type inheriting from Number)	restriction of the appropriate base type with a facet "minInclusive", "minExclusive", "maxInclusive", "maxExclusive" depending on the operator
inv: self.uom.uomName=<UOM> (on a type inheriting from Measure)	restriction of the base type with a restriction of the uom attribute to the fixed value of <UOM>
inv: self.size <op> <N> (on a type inheriting from CharacterString)	restriction of the base type with a facet "minLength", "maxLength" or "length"
inv: self->xsdPatternMatch(<PATTERN>) (on a type inheriting from CharacterString; note that since CharacterString does not provide an operation we define a non-standard additional operation "xsdPatternMatch(pattern : CharacterString) : Boolean" which return true if the value of self satisfies the XML Schema pattern. This is formally not correct, but all other alternatives discussed needed non-standard extensions, too.	restriction of the base type with a facet "pattern"

Currently, no support for the facets "totalDigits", "fractionDigits" and "whitespace" has been specified. If required, a similar approach like for "pattern" might be considered.

The "enumeration" facet and related constraints are supported in UML already by types with the stereotype <<enumeration>>. For code list, in principle a constraint could be added enforces the use of a specific dictionary; this could be represented in Schematron, too, as long as the referenced dictionary follows a standard structure (e.g. a ISO/TS 19139 code list dictionary).

7.3.8 Support for the ISO/TS 19139 encoding rules

In an application schema, in general the GML 3.2.1 Annex E encoding rules apply. However, if metadata elements are part of the application schema, then a UGAS tool also needs to support the ISO/TS 19139 encoding rules. As a result, support for the ISO/TS 19139 encoding rules has been added to ShapeChange.

7.4 Profile of ISO 19107

The spatial schema to be used in connecion with NAS feature types is restricted to the profile described by the DGIWG/TSMAD Profile of ISO 19107.

8 Known issues

8.1 Data Content Specifications vs. application schemas

While the process described in this document works for the NAS, the following two goals for Data Content Specificaitons (DCS) are incompatible with each other, in particular when using XML as the encoding platform:

1. Data according to a DCS, e.g. Local MSD, is also valid NAS data. I.e., if a generic NAS-supporting software that offers no specific support for Local MSD receives Local MSD data, it should be able to detect this data as NAS data and use the data accordingly.

2. Data according to a DCS, e.g. Local MSD, uses only a subset of the model elements of the complete NAS schema. I.e., when a data provider that offers Local MSD data publishes the schema of the data, he should only report model elements from the Local MSD profile as he will never provide any information on those elements that are not part of Local MSD.

Addressing item 2 would require a separate Local MSD application schema (this was the approach taken in OWS-4). However, this results, for example, in an Airspace feature in the NAS namespace (in UML as well as in XML Schema) and another in the Local MSD namespace. As a result, client software will not be able to identify that an Airspace feature in Local MSD conforms also to the requirements for Airspace features in the NAS. Since the first goal is considered more important, the approach documented in this

document and OGC document 08-077 (Local MSD Implementation Profile) specifies only a single application schema, the NAS.

In order to enable clients talking to a WFS that advertises data in the NAS namespace to identify whether the WFS provides full NAS data, Local MSD data or data according to some other DCS, OGC document 08-077 (Local MSD Implementation Profile) specifies the requirement that the WFS shall reference dataset metadata and identify the DCS specification using a standardised citation. This is not a convincing solution as the information is quite hidden and will not be supported by any existing clients.

The real issue here is that the concept of a DCS is not part of the ISO/TC 211 or OGC abstract specifications. Data conforms to the application schema (in this case, the NAS), not a DCS.

The approach consistent with the ISO/TC 211 and OGC reference models seems to be that each DCS is a separate application schema, but specified as subtypes or realisations of the NAS types. However, mapped to XML Schema, this does not really simplify the schema (see the issues below) – in fact, it would make the schema more complex as the NAS and the DCS schema has to be parsed. In addition, software components would need to parse the GML application schemas to identify that the DCS application schema is a profile of the NAS and currently few products seem to support such a capability.

An open issue related to this is the lack of a formal specification of the profile of the complete NAS that is part of a DCS. This is similar to the requirements for formal specifications of metadata profiles of ISO 19115. For the XML platform, such profiles could in principle be represented using Schematron.

8.2 Modelling issues

8.2.1 Rules for coverages in application schemas

Open issues related to coverages that have not been addressed in OWS-5 include:

- Mechanisms need to be specified how to identify the domain and range properties of the ElevationGridCoverage in the UML model (stereotypes? tagged values?) and in the GML application schema.

- Discuss approach within OGC and ISO/TC 211 with the intention of clarifying the guidelines for the use of coverage functions in application schemas (and, as a result, encodings like GML).

8.2.2 Codes and dictionary references

In enumerations, there is no reference where the description of the meaning of the value of the enumerants can be found. So either the application knows where to find the dictionary (i.e. is “GSIP-aware”) or this could be encoded as explicitly. Since in this case the stability of the list of values is not/cannot be enforced in the schema itself, this would

result in the use of the stereotype <<codeList>> instead of <<enumeration>> and the tagged value “asDictionary” set to ”true”.

EXAMPLE:

```
<Bench>
   <conspicuousGroundCategory codeSpace=”URI of the NSG dictionary in
   the appropriate version”>Visual</conspicuousGroundCategory>
</Bench>
```

Note that the current encoding rules do not provide a mechanism to constrain the use of a specific codelist, although such a constraint could in principle be added using another tagged value with a mandatory value for the codeSpace attribute.

The current verson of the NAS subdivides the documentation and can be parsed. For example, the entry for *Visual* is: "Visual: Conspicuous visually. [desc] Conspicuousness by radar unspecified." Here, "[desc]" indication the end of the text representing the definition and the start of text providing an additional description. While this approach is convenient for some UML tools (e.g., a single Documentation string is displayed in a small panel) the NAS 2.5 also includes this information as a set of tags separating the documentation into its five components: *name, definition, description, note*, and (for datatypes) *structureSpecification*.

In general, the code list dictionaries should reside in a registry so that they can be dynamically accessed.

8.3 Schema complexity

Concerns about the complexity of the NAS schemas were raised by some OWS-5 participants. The following issues and recommendations were reported:

8.3.1 Schemas file structure

Due to the over all size and complexity of the NAS it is structured into UML packages with necessary dependencies, some of which are currently mutual. The corresponding NAS GML-based schema is organized into a set of files that involve a lot of 'include' statements, sometimes leading to circular dependencies in the case of package mutuality. Many of those statements can be avoided by a different modularisation of the application schema. Although circular dependencies and redundant include statements are valid as per the XML Schema specification, such constructs may make it more difficult to understand and process the schemas.

Simplifying the schemas (by removing redundant include statements, or reducing the number of XML Schema documents) could ease the issue without changing the content model.

8.3.2 Object model complexity

Regardless of the above issue, the object model of the NAS is very large and complex in order to meet the varying requirements of NSG participants and their missions. This can

be an issue, mainly in terms of performance, for applications (like the catalogue, or OGC mapping clients) that need to access the WFS and parse the schemas.

However, solving this issue would probably imply changing the object model, to optimize it for a WFS usage. This has many more implications than the previous issue.

Due to functional requirements for, e.g., object or attribute level metadata, it appears that this issue may be difficult to avoid.

8.3.3 Potential WFS improvements

8.3.3.1 DescribeFeatureType requests return only the relevant set of schemas [Minor Improvement]

If a WFS returns the complete set of schema components of a namespace as a response to a DescribeFeatureType operation, this can cause performance problems in case of large schemas like the NAS.

However, the schema structure may play a role too here, since the current schema structure apparently makes it difficult for WFS implementations that are not tracking dependencies between schema components to isolate the subset of schema components relevant for a specific feature type.

8.3.3.2 Optimized schemas [Major Improvement]

The size/complexity of the schemas, even for a single feature type, is very large. An option might be to publish the data via two different schemas: one offering the full schema and the offering the dataset according to a simplified schema (that may still be sufficient for many applications). However, the question would be what is the set of data that should consitute the simplified schema.

Note that this is different from the discussion in 8.1 above, which is about a subset of the feature types and feature properties of the complete NAS, while this aspect is about a simplified representation of the feature properties, e.g. without the property metadata that may not be relevant for specific purposes and contexts.