

eCH-0118 GML-Kodierungsregeln für INTERLIS

Name	GML-Kodierungsregeln für INTERLIS
Standard-Nummer	eCH-0118
Kategorie	Standard
Reifegrad	Experimentell
Version	1.00
Status	Genehmigt
Genehmigt am	2011-06-08
Ausgabedatum	2011-06-14
Ersetzt Standard	
Sprachen	Deutsch
Autoren	eCH-Fachgruppe INTERLIS <i>Claude Eisenhut, Eisenhut Informatik AG</i> <i>Michael Germann, infoGrips GmbH</i> <i>Dr. Peter Staub, swisstopo</i>
Herausgeber / Vertrieb	Verein eCH, Mainaustrasse 30, Postfach, 8034 Zürich T 044 388 74 64, F 044 388 71 80 www.ech.ch / info@ech.ch

Zusammenfassung

Dieser eCH-Standard definiert Kodierungsregeln, um aus einem INTERLIS-Datenmodell ein GML-Transferformat abzuleiten. INTERLIS ist eine Schweizer Norm zur Beschreibung von konzeptionellen Datenmodellen und zum modellbasierten Datenaustausch. GML ist ein Standard zur Kodierung von geografischen Daten, basierend auf XML.

Zunächst werden grundlegende Konzepte der verwendeten Sprachen INTERLIS und GML sowie Kodierungsregeln und XML-Anwendungsfälle erläutert (Kapitel 4). Nach grundsätzlichen Aspekten zur XML-Objektkodierung folgt eine Darstellung der Unterschiede zwischen der GML-Kodierung und der Kodierung von INTERLIS-spezifischem XML (Kapitel 5 und 6).

Der Hauptteil dieses Standards beschäftigt sich mit der detaillierten, umfassenden Definition von Schema-Kodierungsregeln (Kapitel 7) sowie Instanz-Kodierungsregeln (Kapitel 8).

Im Anhang A ist das vordefinierte Basis-XML-Schema zur GML-Kodierung von INTERLIS eingefügt. Dieses Dokument ist neben dem vorliegenden Standard separat als XML-Schemadatei verfügbar. Die vorliegende Spezifikation wird durch ein einfaches Anwendungsbeispiel abgeschlossen (Anhang B).

Inhaltsverzeichnis

1	Status des Dokuments	5
2	Einleitung	6
2.1	Überblick	6
2.2	Voraussetzungen.....	6
2.3	Allgemeine Ziele und Einschränkungen.....	6
3	Normative Referenzen	7
4	Begriffe, Konzepte	8
4.1	INTERLIS	8
4.2	Geography Markup Language (GML)	8
4.3	Kodierungsregeln	9
4.4	Anwendungsfälle von XML	9
5	Grundsätzliches zur Kodierung von Objekten in XML	11
5.1	Objekteigenschaften: Attribute im INTERLIS-Modell	11
5.2	Assoziationen	11
5.3	Typdiskriminator	11
5.4	Vererbung.....	11
5.5	Semantische Transformation.....	11
6	Unterschiede zu den INTERLIS-XML-Kodierungsregeln	12
6.1	Kodierungsregeln	12
6.2	Aufbau eines Transfers	12
6.3	Lesen von erweiterten/übersetzten Modellen.....	12
6.4	Zeichenvorrat.....	12
6.5	Transferarten.....	13
7	Schema-Kodierungsregeln	14
7.1	Vorbemerkung.....	14
7.2	Zeichenkodierung	14
7.3	Allgemeine Regeln.....	14
7.4	Abbildung der INTERLIS-Modellelemente auf XML-Schema-Namen	14
7.5	Abbildung von Basistypen.....	15

7.6	Allgemeiner Aufbau des XML-Schemas	15
7.7	Kodierung von Wertebereichen	16
7.8	Kodierung von Themen	17
7.9	Kodierung von Klassen, Strukturen und Assoziationen.....	19
7.10	Kodierung von Sichten.....	20
7.11	Kodierung von Beziehungen.....	20
7.11.1	Eingebettete Beziehungen	21
7.11.2	Nicht eingebettete Beziehungen.....	23
7.12	Kodierung von Grafikdefinitionen.....	25
7.13	Kodierung von Attributen	25
7.14	Abbildung der INTERLIS-Typen auf XML-Schema-Typen	28
7.14.1	Übersicht.....	28
7.14.2	Kodierung von Zeichenketten	28
7.14.3	Kodierung von Aufzählungen.....	29
7.14.4	Kodierung von numerischen Datentypen.....	32
7.14.5	Kodierung von formatierten Wertebereichen	32
7.14.6	Kodierung von Gefässen	33
7.14.7	Kodierung von Klassentypen	34
7.14.8	Kodierung von Attributpfadtypen	34
7.14.9	Kodierung von Koordinaten	34
7.14.10	Kodierung von Linienzügen.....	35
7.14.11	Kodierung von Einzelflächen.....	36
7.14.12	Kodierung von Gebietseinteilungen	36
7.14.13	Kodierung von Typen zur Objektidentifikation	38
7.14.14	Kodierung von weiteren Kurvenstück-Formen.....	39
8	Instanz-Kodierungsregeln	40
8.1	Kodierung mehrerer Behälter (TOPICS) in einer Transferdatei	40
8.2	Kodierung von Objekten und Strukturelementen	40
8.3	Kodierung von Objekt- und Behälteridentifikationen	40
8.4	Kodierung von Beziehungen.....	41
8.5	Kodierung von Geometrien.....	41
9	Haftungsausschluss/Hinweise auf Rechte Dritter	42

10	Urheberrechte	42
	Anhang A – Vordefiniertes Basis-Schema	43
	Anhang B – Beispiel.....	45
	INTERLIS 2-Modell.....	45
	INTERLIS 1-Modell.....	46
	GML-Applikationsschema.....	47
	INTERLIS 1-Transferdaten.....	49
	INTERLIS 2-Transferdaten («INTERLIS-XML»).....	50
	GML-Daten.....	51

1 Status des Dokuments

Das vorliegende Dokument wurde vom Expertenausschuss **genehmigt**. Es hat für das definierte Einsatzgebiet im festgelegten Gültigkeitsbereich normative Kraft.

2 Einleitung

2.1 Überblick

In diesem Dokument werden die Regeln definiert, um aus einem INTERLIS-Modell ein GML-Applikationsschema abzuleiten.

2.2 Voraussetzungen

Dieses Dokument setzt INTERLIS- und GML-Kenntnisse voraus. Für eine vertiefte Auseinandersetzung sei auf die Quellen [1], [2] verwiesen. Der Text ist aber mit Beispielen ergänzt, so dass dem Inhalt auch vom Nicht-Experten gefolgt werden kann.

2.3 Allgemeine Ziele und Einschränkungen

- Es wird eine möglichst genaue Modellabbildung angestrebt
- Es entsteht kein Datenverlust gegenüber einem Datentransfer mit INTERLIS 1 oder 2 (ITF oder XTF)
- Das resultierende GML-Applikationsschema lässt sich durch GML-konforme Software nutzen
- Das resultierende GML-Applikationsschema enthält nicht alle Beschreibungsmerkmale des INTERLIS-Modells, weil GML diese Möglichkeiten nicht kennt. Auf der Stufe Modell (Datenbeschreibung) gibt es also einen Verlust, d.h. das GML-Applikationsschema ist kein Ersatz für das INTERLIS-Modell.
- Das resultierende GML-Applikationsschema eignet sich sowohl für den Massendatentransfer als auch für Geowebdienste (insbesondere WFS).
- Dieses Dokument verwendet INTERLIS 2.3 und GML 3.2.x. Da INTERLIS 1 mehr oder weniger 1:1 nach INTERLIS 2.3 übersetzt werden kann, gilt es aber entsprechend auch für INTERLIS 1.

Anmerkung: Durch die Verwendung von GML entsteht kein Datenverlust. Gewisse Fähigkeiten von XTF (INTERLIS 2-XML-Transferformat), die in GML nicht vorhanden sind (z.B. inkrementeller Transfer), werden jedoch nicht nachgebildet. Das resultierende Transferformat ist somit kein vollständiger Ersatz für XTF.

Die GML-Basistypen werden nach Möglichkeit nicht erweitert, um eine möglichst breite Nutzung der Daten zu ermöglichen.

Das resultierende GML-Applikationsschema ist nach Möglichkeit GML Simple Feature (GML-SF) konform, um eine möglichst breite Nutzung der Daten zu ermöglichen. Wenn das INTERLIS Modell Elemente enthält die sich nicht 1:1 nach GML-SF abbilden lassen (z.B. Kreisbogen), ist das resultierende GML Applikationsschema nicht GML-SF-konform.

3 Normative Referenzen

- [1] eCH-Fachgruppe Geoinformation (2006): *eCH-0031: Geoinformation: INTERLIS 2 – Referenzhandbuch*. eCH-Standard.
Online <http://www.ech.ch> → Standards → eCH-0031 (2010-09-06)
- [2] Open Geospatial Consortium OGC (2007): *OpenGIS Geography Markup Language (GML) Encoding Standard*. OGC 07-036, Version 3.2.1.
Online http://portal.opengeospatial.org/files/?artifact_id=20509 (2010-09-06)
- [3] World Wide Web Consortium W3C (2004): *XML Schema 1.1*. Standard.
Online <http://www.w3.org/XML/Schema> (2010-09-06)
- [4] ISO/IEC 14977 (1996): *Information technology – Syntactic metalanguage – Extended BNF*. International standard.
- [5] ISO/IEC 10646:2003 (2003): *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*. International standard.
- [6] eCH-Fachgruppe INTERLIS (2011): *eCH-0117: Meta-Attribute für INTERLIS-Modelle*. eCH-Standard.

4 Begriffe, Konzepte

4.1 INTERLIS

INTERLIS ist eine konzeptionelle Datenbeschreibungssprache und wird im Bereich der Geodatenmodellierung und des Geodatenaustausches verwendet [1].

Das INTERLIS-Referenzhandbuch [1] besteht aus zwei Teilen:

- eine Datenbeschreibungssprache, und
- XML-Kodierungsregeln.

Werden die XML-Kodierungsregeln auf ein mit Hilfe der Datenbeschreibungssprache definiertes Datenmodell angewendet, entsteht ein zum Datenmodell passendes Transferformat.

Für die XML-Kodierungsregeln wird im vorliegenden Dokument eine Alternative definiert.

Im Anhang B ist ein Beispiel-Datenmodell in INTERLIS inkl. einer Transferdatei gemäss diesem Datenmodell eingefügt.

4.2 Geography Markup Language (GML)

GML [2] ist ein Satz von Basis-XML-Schemas [3] und ein Satz von Regeln, wie diese Schemas in eigenen Schemas (sog. *Applikationsschemas*) für die Definition von Transferformaten zu verwenden sind. Aus der Kenntnis der Basis-Schemas und der Regeln für Applikationsschemas lässt sich somit auch eine Datenbeschreibung herleiten.

Anmerkung: Weil die Regeln für GML-Applikationsschemas definiert sind, kennt man nicht nur die zulässige Folge von XML-Elementen (aufgrund des XML-Schemas), sondern weiss, welche XML-Elemente Klassen darstellen und welche Elemente Attribute dieser Klassen sind. Diese Beschreibung ist weniger genau als mittels INTERLIS, darum geht es aber in diesem Dokument nicht primär.

Die INTERLIS 1-FMT-Datei ist auch eine Formatbeschreibung. Im Gegensatz zu einem XML-Schema handelt es sich dabei aber um keine formale Beschreibung.

Im Anhang B ist ein Beispiel-Applikationsschema inkl. einer Transferdatei gemäss diesem Schema eingefügt.

Anmerkung: Die GML 3.1-Schemas validieren teilweise nicht! Das ist nicht nur ein Problem von GML; siehe dazu auch <http://schemas.opengis.net/gml/3.1.1/readme.txt>

In der GML 3.2-Spezifikation [2] steht dazu:

«21.2.6 Property Type Derivation [...]

NOTE As derivation-by-restriction of property types has created problems with commonly used XML parsers in the past, all instances of such derivations have been removed from the GML schema. It is recommended to avoid derivation by restriction in property types in application schemas, too.»

4.3 Kodierungsregeln

Kodierungsregeln definieren, wie aus einem fachlichen Datenmodell durch die Anwendung von Regeln das Transferformat abgeleitet werden kann (modellbasierter Datentransfer). Die Regeln sind dabei so formuliert, dass sie auf ein beliebiges Datenmodell angewendet werden können.

4.4 Anwendungsfälle von XML

Es gibt grundsätzlich die folgenden Anwendungsfälle, die das Design einer XML-Struktur stark beeinflussen:

- *Texte/Dokumente* (z.B. XHTML, DocBook). In diesem Fall stehen die Lesbarkeit für den Menschen und die korrekte Reihenfolge/Strukturierung der XML-Elemente im Vordergrund. Datentypen sind weniger wichtig.
- *Meldungen* (z.B. Liegenschaftsmutation oder im Kontext AJAX und Web-Services). In diesem Fall steht die einfache Verarbeitung im Vordergrund, oft nimmt man darum bewusst Redundanzen der Daten in Kauf.
- *Massendatentransfer* (z.B. Amtliche Vermessungsschnittstelle). In diesem Fall stehen die effiziente Verarbeitung und die Grösse der XML-Datei im Vordergrund.
- *Riesige Datenmenge* (z.B. Bilder, Messreihen). In diesem Fall steht die Grösse der XML-Datei so dominant im Vordergrund, dass die Markierung der einzelnen Informationsteile teilweise oder vollständig entfällt.
- *durch Mensch editierte «Daten»* (z.B. XML-Schema-Dokumente, «ant build»-Skripts). In diesem Fall steht die Lesbarkeit für den Menschen im Vordergrund. Im Gegensatz zum Fall «Texte/Dokumente» sind hier auch Datentypen wichtig.

Weitere Aspekte:

- *Applikations-interner Gebrauch*. Man kann die XML-Struktur genau nach den Bedürfnissen dieser Applikation ausrichten.
- *Unterschiedliche Nutzungszusammenhänge*, z.B. verschieden «harte» Restriktionen, weil in einer Kette von Verarbeitungsvorgängen zu einem frühen Zeitpunkt noch nicht alle Informationen vorhanden sind.
- *Applikations-übergreifender Gebrauch*. Man kann die XML-Struktur nicht nach den Bedürfnissen einer Applikation ausrichten, die verschiedenen Applikationen haben aber typischerweise doch eine ähnliche Vorstellung von der Realität.
- *Fachbereichs-übergreifender Gebrauch*. Die verschiedenen Applikationen haben typischerweise eine unterschiedliche Vorstellung der Realität. Beispiel: In der Steuerverwaltung stehen die Personen im Vordergrund, Grundstücke und Gebäude kommen auch vor. Im Grundbuch stehen die dinglichen Rechte im Vordergrund, Grundstücke und Personen kommen auch vor. In der Amtlichen Vermessung stehen Geometrien von Grundstücken und Gebäuden im Vordergrund, Personen kommen nicht vor.
- Existierendes Schema oder existierender Schema-Designstil

- Metadaten, insbesondere auch Schema, als Teil der Daten
- Generische, statt fachspezifische Struktur, z.B.:

```
<object class="Person">  
  <attr name="Name" value="Muster"/>  
  <attr name="Vorname" value="Peter"/>  
</object>
```
- *Erweiterbarkeit*: kann man weitere Eigenschaften (z.B. für Versionierung) einfach in die bestehenden Strukturen einbetten oder darauf verweisen?
- *Änderungstoleranz*: kann man die Datenstrukturen leicht ändern, ohne dass man die Programme vollständig neu entwickeln *muss*?

5 Grundsätzliches zur Kodierung von Objekten in XML

Es gilt ein paar grundsätzliche Entwurfsentscheide zu treffen, die meisten davon sind allerdings schon durch die GML-Spezifikation getroffen worden.

5.1 Objekteigenschaften: Attribute im INTERLIS-Modell

Objekteigenschaften lassen sich als XML-Attribute oder XML-Subelemente kodieren. Durch GML wird die Variante XML-Subelemente verlangt.

5.2 Assoziationen

Assoziationen lassen sich als eigenständige Objekte («Zwischentabelle»), als einseitige Referenz, als gegenseitige Referenz oder als Objekteinbettung kodieren.

Von GML wird keine bestimmte Variante verlangt.

5.3 Typdiskriminator

Der Typ eines Objektes kann als XML-Attribut (in der Regel `xsi:type`) oder XML-Element codiert werden.

Durch GML wird die Variante XML-Element verlangt. GML bezeichnet das als «Object-Property-Object Pattern».

5.4 Vererbung

Die Vererbungshierarchie kann im XML-Schema ausgedrückt werden oder nach verschiedenen Verfahren zu inhaltsgleichen Strukturen abgebildet werden (die aber die Vererbung nicht mehr zum Ausdruck bringen; ähnlich wie beim O/R-Mapping¹).

5.5 Semantische Transformation

Im Prinzip kann man auch semantische Transformationen durchführen, solange sie den Inhalt nicht ändern, sondern nur anders strukturieren. Zum Beispiel könnte man zwei über eine 1:1-Assoziation verbundene Klassen zu einem einzigen XML-Element zusammenfassen.

¹ *dt.* Objektrelationale Abbildung; Verfahren zur Abbildung von Objekten in relationalen Datenbanken

6 Unterschiede zu den INTERLIS-XML-Kodierungsregeln

6.1 Kodierungsregeln

Das INTERLIS-Referenzhandbuch definiert Instanz-Kodierungsregeln. Im vorliegenden Dokument werden Schema-Kodierungsregeln definiert und die Instanzen ergeben sich aus dem so erzeugten Schema.

6.2 Aufbau eines Transfers

Das INTERLIS-Referenzhandbuch unterteilt einen Transfer in Vorspann und Datenbereich. Auf den Vorspann wird in diesem Dokument verzichtet, da dies durch eine GML-konforme Software als Daten interpretiert würde (statt als Metadaten zum Transfer).

6.3 Lesen von erweiterten/übersetzten Modellen

Das INTERLIS-Referenzhandbuch definiert als Teil des Vorspanns eine Abbildungstabelle, so dass ein Leseprogramm das für ein bestimmtes Datenmodell programmiert oder konfiguriert worden ist, Daten von Erweiterungen (oder Übersetzungen) dieses Datenmodells ohne Kenntnis der erweiterten Modelldefinitionen lesen kann. In diesem Dokument wird auf eine solche Abbildungstabelle verzichtet. Ein Leseprogramm muss also das erweiterte Datenmodell kennen, um Erweiterungen korrekt ignorieren zu können. Für übersetzte INTERLIS-Datenmodelle wird kein GML-Applikationsschema erzeugt, d.h. das Transferformat ändert sich bei einer Übersetzung nicht.

Anmerkung: Ist die Handhabung einer unbekanntem Modellerweiterung mit GML (ohne Spezialsoftware) überhaupt möglich?

Unbekannt im Sinne von «Schema nicht verfügbar»: nein.

Unbekannt im Sinne von «Schema verfügbar, aber nicht implementiert»: ja.

6.4 Zeichenvorrat

Das INTERLIS-Referenzhandbuch definiert einen eingeschränkten Zeichenvorrat (Anhang B im Referenzhandbuch) [1].

Das vorliegende Dokument definiert standardmässig den Unicode Zeichenumfang. Ein eingeschränkter Zeichenvorrat muss im Rahmen einer Transfergemeinschaft definiert werden.

6.5 Transferarten

Das INTERLIS-Referenzhandbuch definiert drei Transferarten: «FULL», «INITIAL» oder «UPDATE». Zusätzlich können Angaben zur Konsistenz übermittelt werden: «COMPLETE», «INCOMPLETE», «INCONSISTENT», «ADAPTED».

Das vorliegende Dokument beschreibt, wie im Rahmen von Web-Diensten einzelne Objekte transferiert werden. Optional wird der klassische dateibasierte Transfer definiert (entspricht in INTERLIS: «FULL» und «COMPLETE»).

Anmerkung: Inkrementeller Transfer lässt sich mit GML Standardsoftware nicht realisieren, da GML keine solche Semantik definiert. Konsistenzangaben könnte man als GML-Metadaten codieren.

7 Schema-Kodierungsregeln

7.1 Vorbemerkung

Für die Formalisierung der Transferformat-Ableitungsregeln wird die im INTERLIS-Referenzhandbuch in Kapitel 2.1 eingeführte EBNF-Notation benutzt [4].

7.2 Zeichenkodierung

Es gelten die XML-Regeln: Unicode-Zeichensatz und UTF-8 (empfohlen) oder UTF-16 als Zeichencodierung [5].

7.3 Allgemeine Regeln

Das generierte XML-Schema [3] kann beliebig viele Kommentare (<!-- ... -->) oder zusätzliche `xsd:annotation`-Elemente enthalten.

7.4 Abbildung der INTERLIS-Modellelemente auf XML-Schema-Namen

Jedes INTERLIS-Datenmodell enthält einen eigenen XML-Schema-Namensraum. Für die XML-Element- und -Typ-Definitionen werden die unqualifizierten INTERLIS-Namen verwendet. Bei Konflikten wird der qualifizierte Namen verwendet. Die Namen auf Stufe MODEL haben Vorrang gegenüber Namen aus der Stufe TOPIC. Bei Konflikten zwischen Namen aus unterschiedlichen TOPICs hat der Name aus dem im Modell zuerst definierten TOPIC Vorrang.

Beispiel INTERLIS	GML Element-Namen
<pre> INTERLIS 2.3; MODEL ModelA (de) AT "mailto:ceis@localhost" VERSION "2010-02-31" = TOPIC TopicA = CLASS ClassA = END ClassA; CLASS TopicA = !! qualifizierter Name !! wegen Konflikt END TopicA; END TopicA; END ModelA. </pre>	<pre> TopicA ClassA TopicA.TopicA </pre>

7.5 Abbildung von Basistypen

Für modellunabhängige Basistypen aus INTERLIS (z.B. «HALIGNMENT», «SurfacePropertyType») wird ein Basisschema (Anhang A) definiert.

7.6 Allgemeiner Aufbau des XML-Schemas

Jedes INTERLIS-Modell wird in ein XML-Schema in einem Schemadokument abgebildet.

```

ModelDef = '<xsd:schema
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns="%ModelNamespaceIdentifier%"
    targetNamespace="%ModelNamespaceIdentifier%"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    xmlns:gml="http://www.opengis.net/gml/3.2"
    { 'xmlns:%nsprefix%' namespace=
      "%Imported-ModelNamespaceIdentifier%" }
    '>'
  IliModelInfo
  '<xsd:import
    namespace="http://www.opengis.net/gml/3.2"/>'
  { ModelImport }
  { ClassDef | AssociationDef | DomainDef }
  { TopicDef }
  '</xsd:schema>'.

IliModelInfo = '<xsd:annotation>
  <xsd:appinfo source="http://www.interlis.ch/ili2c">
  <ili2:model>%ModelName%</ili2:model>
  <ili2:modelVersion>%ModelVersion%
  </ili2:modelVersion>
  <ili2:modelAt>%ModelAt%</ili2:modelAt>
  </xsd:appinfo>
  </xsd:annotation>'.

ModelImport = '<xsd:import
  namespace="%Imported-ModelNamespaceIdentifier%"/>'.

ModelNamespaceIdentifier = StdModelNamespaceIdentifier
  | %MetaAttributeValue%.

StdModelNamespaceIdentifier =
  'http://www.interlis.ch/ILIGML-1.0/'%ModelName%.

```

Der XML-Namensraumidentifikator (Regel «ModelNamespaceIdentifier») ergibt sich aus dem Modellnamen oder kann über das Metaattribut «ili2.iligml10.namespaceName» definiert werden [6].

Beispiel

INTERLIS	<pre> INTERLIS 2.3; MODEL ModelA (de) AT "mailto:ceis@localhost" VERSION "2010-02-31" = IMPORTS Units; </pre>
----------	--

	END ModelA.
GML	<pre> <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://www.interlis.ch/ILIGML-1.0/ModelA" targetNamespace= "http://www.interlis.ch/ILIGML-1.0/ModelA" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:ns1= "http://www.interlis.ch/ILIGML-1.0/Units"> <xsd:annotation> <xsd:appinfo source="http://www.interlis.ch/ili2c"> <ili2:model>ModelA</ili2:model> <ili2:modelVersion>2010-0-31</ili2:modelVersion> <ili2:modelAt>mailto:ceis@localhost</ili2:modelAt> </xsd:appinfo> </xsd:annotation> <xsd:import namespace="http://www.opengis.net/gml/3.2"/> <xsd:import namespace ="http://www.interlis.ch/ILIGML-1.0/Units"/> </xsd:schema> </pre>

7.7 Kodierung von Wertebereichen

Wertebereichsdefinitionen werden wie folgt abgebildet:

```

DomainDef = DomainDefComplex | DomainDefSimple.

DomainDefComplex = '<xsd:complexType name="%DomainName%">
  <xsd:complexContent>
    <xsd:restriction base="%BaseType%">
      <!-- any applicable xsd restrictions -->
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>'.

DomainDefSimple = '<xsd:simpleType name="%DomainName%">
  <xsd:restriction base="%BaseType%">
    <!-- any applicable xsd restrictions -->
  </xsd:restriction>
</xsd:simpleType>'.

```

Je nach Abbildungsregel des entsprechenden INTERLIS-Datentyps (siehe Kapitel 7.14) wird die Regel «DomainDefComplex» oder «DomainDefSimple» angewendet.

Beispiel	
INTERLIS	<pre> DOMAIN LKoord = COORD 480000.00 .. 850000.00 [m] {CHLV03[1]}, 60000.00 .. 320000.00 [m] {CHLV03[2]}, ROTATION 2 -> 1; Horizontbezeichnung = TEXT*20; </pre>

GML	<pre> <xsd:complexType name="LKoord"> <xsd:complexContent> <xsd:restriction base="gml:PointPropertyType"> </xsd:restriction> </xsd:complexContent> </xsd:complexType> <xsd:simpleType name="Horizontbezeichnung"> <xsd:restriction base="xsd:normalizedString"> <xsd:maxLength value="20"/> </xsd:restriction> </xsd:simpleType> </pre>
-----	---

7.8 Kodierung von Themen

Themen haben in INTERLIS zwei Funktionen: einerseits definieren sie einen eigenen Namensraum (getrennt vom Namensraum des Modells), andererseits definieren sie einen Behälter. Die Eigenschaft als Namensraum wird nicht abgebildet. Die Eigenschaft als Behälterbeschreibung wird wie folgt abgebildet.

```

TopicDef = '<xsd:complexType name="%TopicName%MemberType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureMemberType">
      <xsd:sequence>
        <xsd:choice>'
          { '<xsd:element ref="%ClassName%"/>' }
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="%TopicName%" type="%TopicName%Type"
  substitutionGroup="gml:AbstractFeature"/>
<xsd:complexType name="%TopicName%Type">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="member"
          type="%TopicName%MemberType"
          minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attributeGroup
        ref="gml:AggregationAttributeGroup"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>'.

```

Beispiel

INTERLIS	<pre> TOPIC Bodenbedeckung = CLASS BoFlaechen = END BoFlaechen; CLASS Gebaeude = END Gebaeude; </pre>
----------	---

	END Bodenbedeckung;
GML	<pre> <xsd:complexType name="BodenbedeckungMemberType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureMemberType"> <xsd:sequence> <xsd:choice> <xsd:element ref="BoFlaechen"/> <xsd:element ref="Gebaeude"/> </xsd:choice> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Bodenbedeckung" type="BodenbedeckungType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="BodenbedeckungType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="member" type="BodenbedeckungMemberType" minOccurs="0" maxOccurs="unbounded"/> </xsd:sequence> <xsd:attributeGroup ref="gml:AggregationAttributeGroup"/> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre>
GML Instanz	<pre> <Bodenbedeckung gml:id="b1"> <member> <BoFlaechen gml:id="o1"> ... </BoFlaechen> </member> <member> <BoFlaechen gml:id="o2"> ... </BoFlaechen> </member> </Bodenbedeckung> </pre>

7.9 Kodierung von Klassen, Strukturen und Assoziationen

Klassen und Strukturen werden wie folgt abgebildet:

```

ClassDef = '<xsd:element name="%ClassName%"
  type="%ClassName%Type"
  substitutionGroup="gml:AbstractFeature
    | %Base-ClassName%"/>
  <xsd:complexType name="%ClassName%Type">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractFeatureType
        | %Base-ClassName%Type">
        <xsd:sequence>
          { AttributeDef | EmbeddedRoleDef }
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>'.

```

Im XML-Schema lässt sich somit nicht mehr erkennen ob es sich um eine «CLASS» oder «STRUCTURE» handelt.

Assoziationen (siehe auch Kapitel 7.11) werden wie folgt abgebildet:

```

AssociationDef = '<xsd:element name="%AssociationName%"
  type="%AssociationName%Type"
  substitutionGroup="gml:AbstractFeature |
    | %Base-AssociationName%"/>
  <xsd:complexType name="%AssociationName%Type">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractFeatureType |
        | %Base-AssociationName%Type">
        <xsd:sequence>
          { RoleDef } { AttributeDef | EmbeddedRoleDef }
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>'.

```

Für die Reihenfolge der Attribute, Rollen, Eingebetteten-Rolle innerhalb der Klasse/Assoziation gilt: Zuerst werden alle Rollen, dann alle Attribute und dann alle Eingebetteten-Rollen codiert. Die Attribute und Rollen werden gemäss ihrer Definitionsreihenfolge in der Modelldatei codiert. Die Eingebetteten-Rollen werden alphabetisch aufsteigend sortiert. Ein spezialisiertes Attribut oder eine spezialisierte Rolle (mit dem Schlüsselwort «EXTENDED»), wird nicht generiert.

Ist die Klasse keine Erweiterung einer anderen Klasse, wird `gml:AbstractFeatureType` als Basistyp, bzw. `gml:AbstractFeature` als Basis-Element verwendet. Ist die Klasse eine Erweiterung, wird der entsprechend generierte Basistyp bzw. das Basis-Element verwendet. Die Vererbungsstruktur der Klassen ist somit im XML-Schema wiedergegeben.

Bei eingebetteten Assoziationen werden nur Attribute als Teil des Typs für die Assoziation codiert, d.h. keine «RoleDef» (siehe auch Kapitel 7.11.1).

Parameter werden mit einer Ausnahme, wie sie im Kapitel 7.12 angegeben ist, nicht übertragen.

Beispiel	
INTERLIS	<pre> CLASS Horizont = Horizontbezeichnung : MANDATORY TEXT*10; END Horizont; </pre>
GML	<pre> <xsd:element name="Horizont" type="HorizontType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="HorizontType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="Horizontbezeichnung"> <xsd:simpleType> <xsd:restriction base="xsd:normalizedString"> <xsd:maxLength value="10"/> </xsd:restriction> </xsd:simpleType> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre>
GML Instanz	<pre> <Horizont gml:id="o1"> <Horizontbezeichnung>Ah</Horizontbezeichnung> </Horizont> </pre>

7.10 Kodierung von Sichten

Zur Kodierung von Sichten vgl. Kapitel 3.2.4 im INTERLIS-Referenzhandbuch [1]. Als Attribute des Sichtobjekts werden nur diejenigen Attribute übertragen, welche in der Sicht explizit unter «ATTRIBUTE» bzw. implizit mit «ALL OF» angegeben wurden.

7.11 Kodierung von Beziehungen

Beziehungen werden auf zwei Arten kodiert: eingebettet oder nicht eingebettet. Eine eingebettete Beziehung wird als Sub-Element von einer an der Assoziation beteiligten Klasse kodiert. Die Instanz einer nicht eingebetteten Beziehung (Link) wird wie eine Instanz einer Klasse kodiert.

Beziehungen werden immer eingebettet, ausser

- wenn sie mehr als zwei Rollen haben
- wenn bei beiden (Basis-)Rollen die maximale Kardinalität grösser 1 ist
- wenn für die Beziehung eine OID gefordert wird
- bei gewissen themenübergreifenden Beziehungen (s. unten).

Falls bei einer der beiden (Basis-)Rollen die maximale Kardinalität grösser 1 ist, wird bei der Ziel-Klasse dieser Rolle eingebettet. Wenn diese Ziel-Klasse in einem anderen Topic definiert ist als die (Basis-)Assoziation, kann nicht eingebettet werden.

Falls bei beiden (Basis-)Rollen die maximale Kardinalität kleiner gleich 1 ist, wird bei der Ziel-Klasse der zweiten Rolle eingebettet. Wenn diese Ziel-Klasse in einem anderen Topic definiert ist als die (Basis-)Assoziation und die Ziel-Klasse der ersten Rolle im selben Topic definiert ist wie die (Basis-)Assoziation, wird bei der Ziel-Klasse der ersten Rolle eingebettet. Das bedeutet, dass nicht eingebettet werden kann, wenn die Ziel-Klassen der beiden Rollen in einem anderen Topic definiert sind als die (Basis-)Assoziation.

7.11.1 Eingebettete Beziehungen

Eingebettete Beziehungen werden als Referenz in der Klasse, bei der die Beziehung eingebettet wird, übertragen. Ist die Beziehung geordnet, wird ein zusätzliches XML-Attribut («ORDER_POS»; in Anhang A definiert) erforderlich.

```

EmbeddedRoleDef = ( EmbeddedUnorderedRoleDef
    | EmbeddedOrderedRoleDef)
    '<xsd:element name="%RoleName%.LINK_DATA"
      'minOccurs="0">'
      '<xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="%AssociationName%"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>'.

EmbeddedUnorderedRoleDef =
    '<xsd:element name="%RoleName%" type="gml:ReferenceType">
      <xsd:annotation>
        <xsd:appinfo>
          <gml:targetElement>%ClassName%</gml:targetElement>
        </xsd:appinfo>
      </xsd:annotation>
    </xsd:element>'.

EmbeddedOrderedRoleDef =
    '<xsd:element name="%RoleName%">
      <xsd:annotation>
        <xsd:appinfo>
          <gml:targetElement>%ClassName%</gml:targetElement>
        </xsd:appinfo>
      </xsd:annotation>
      <xsd:complexType>
        <xsd:sequence/>
        <xsd:attributeGroup ref="gml:OwnershipAttributeGroup"/>
        <xsd:attributeGroup ref="gml:AssociationAttributeGroup"/>
        <xsd:attribute ref="ili:ORDER_POS"/>
      </xsd:complexType>
    </xsd:element>'.
  
```

Für %RoleName% muss der Name der Rolle angegeben werden, welche auf das gegenüberliegende Objekt verweist (die andere Rolle wird nicht kodiert).

Hat die eingebettete Beziehung Attribute, werden diese als zusätzliche Elemente %RoleName%.LINK_DATA kodiert. Im Schema wird dieses Element immer definiert, ausser die Beziehung ist «FINAL» und hat keine Attribute. Ist die Beziehung nicht «FINAL», können die Attribute evtl. erst in einer Erweiterung hinzugefügt werden; das Element für die Einbettung muss darum auch ohne Attribute bei Beziehungen ohne «FINAL» definiert werden.

Beispiel	
INTERLIS	<pre> CLASS Liegenschaft = END Liegenschaft; CLASS Person = END Person; ASSOCIATION Eigentum = Grundsteuck -- {0..*} Liegenschaft; Eigentuerer -- {1} Person; END Eigentum; </pre>
GML	<pre> <xsd:element name="Liegenschaft" type="LiegenschaftType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="LiegenschaftType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="Eigentuerer" type="gml:ReferenceType"> <xsd:annotation> <xsd:appinfo> <gml:targetElement>Person</gml:targetElement> </xsd:appinfo> </xsd:annotation> </xsd:element> <xsd:element name="Eigentuerer.LINK_DATA" minOccurs="0"> <xsd:complexType> <xsd:sequence> <xsd:element ref="Eigentum"/> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Person" type="PersonType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="PersonType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Eigentum" type="EigentumType" substitutionGroup="gml:AbstractFeature"/> </pre>

	<pre><xsd:complexType name="EigentumType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType></pre>
GML Instanz	<pre><Person gml:id="o1"/> <Liegenschaft gml:id="o2"> <Eigentuermer xlink:href="#o1"/> </Liegenschaft></pre>

7.11.2 Nicht eingebettete Beziehungen

Nicht eingebettete Beziehungen werden wie Objektinstanzen von Klassen übertragen.

Hinweis: Für Beziehungen ohne expliziten Namen wird der (Klassen)Name durch zusammenhängen der einzelnen Rollennamen gebildet (z.B. %RoleName1RoleName2%).

Rollen werden wie Attribute behandelt. Die Rollen selbst werden wie folgt kodiert:

```
RoleDef = UnorderedRoleDef | OrderedRoleDef.

UnorderedRoleDef =
  '<xsd:element name="%RoleName%" type="gml:ReferenceType">
    <xsd:annotation>
      <xsd:appinfo>
        <gml:targetElement>%ClassName%</gml:targetElement>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>'.

OrderedRoleDef =
  '<xsd:element name="%RoleName%">
    <xsd:annotation>
      <xsd:appinfo>
        <gml:targetElement>%ClassName%</gml:targetElement>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence/>
      <xsd:attributeGroup ref="gml:OwnershipAttributeGroup"/>
      <xsd:attributeGroup ref="gml:AssociationAttributeGroup"/>

      <xsd:attribute ref="ili:ORDER_POS"/>
    </xsd:complexType>
  </xsd:element>'.

```

Ist die Beziehung geordnet, ist gegenüber `gml:ReferenceType` ein zusätzliches XML-Attribut («ORDER_POS»; in Anhang A definiert) erforderlich.

Ob die Referenz auf ein Objekt in der gleichen Transferdatei zeigt, lässt sich nicht immer erkennen (nur wenn die Referenz ein Dokument relativer XML-Fragment-Identifikator ist).

Beispiel	
INTERLIS	<pre> CLASS Person = END Person; CLASS Verein = END Verein; ASSOCIATION DauerhafteVereinigungMitglied = DauerhafteVereinigung -- {0..*} Verein; Mitglied -- {1..*} Person; END DauerhafteVereinigungMitglied; </pre>
GML	<pre> <xsd:element name="Verein" type="VereinType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="VereinType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Person" type="PersonType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="PersonType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="DauerhafteVereinigungMitglied" type="DauerhafteVereinigungMitgliedType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="DauerhafteVereinigungMitgliedType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="DauerhafteVereinigung" type="gml:ReferenceType"> <xsd:annotation> <xsd:appinfo> <gml:targetElement>Verein</gml:targetElement> </xsd:appinfo> </xsd:annotation> </xsd:element> <xsd:element name="Mitglied" type="gml:ReferenceType"> <xsd:annotation> <xsd:appinfo> <gml:targetElement>Person</gml:targetElement> </xsd:appinfo> </xsd:annotation> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre>

GML Instanz	<pre><Person gml:id="o1"/> <Verein gml:id="o2"/> <DauerhafteVereinigungMitglied gml:id=""> <DauerhafteVereinigung xlink:href="#o2"/> <Mitglied xlink:href="#o1"/> </DauerhafteVereinigungMitglied></pre>
----------------	--

7.12 Kodierung von Grafikdefinitionen

Für jede Grafikdefinition werden im Transfer die von der Grafikdefinition referenzierten Signaturklassen («Sign-ClassRef») übertragen. Die Objektinstanzen der Signaturklassen werden durch das Ausführen der Grafikdefinitionen auf einem konkreten Inputdatensatz erzeugt. Parameter werden dabei wie Attribute codiert.

7.13 Kodierung von Attributen

Jedes Attribut einer Objektinstanz (einschliesslich komplexer Attribute wie «COORD», «POLYLINE», «SURFACE», «AREA», «STRUCTURE», «LIST OF», «BAG OF» etc.) wird wie folgt kodiert:

```
AttributeDef = ExplicitTypeAttribute
              | ImplicitTypeAttribute
              | StructAttribute
              | ReferenceAttribute.

ExplicitTypeAttribute =
  '<xsd:element name="%AttributeName%" type="%DataType%"'
  [ 'minOccurs="0"' ] '/>'.

ImplicitTypeAttribute =
  '<xsd:element name="%AttributeName%"'
  [ 'minOccurs="0"' ] '>'
  '<xsd:simpleType>'
  '<xsd:restriction base="%BaseType%">'
  '<!-- any applicable xsd restrictions -->'
  '</xsd:restriction>'
  '</xsd:simpleType>'
  '</xsd:element>'.

StructAttribute = '<xsd:element name="%AttributeName%"'
  [ 'minOccurs="0"' ]
  [ 'maxOccurs="%maxCardinality%" ] '>'
  '<xsd:complexType>'
  '<xsd:sequence>'
  '<xsd:element ref="%StructureName%"' '/>'
  '</xsd:sequence>'
  '</xsd:complexType>'
  '</xsd:element>'.

ReferenceAttribute =
  '<xsd:element name="%AttributeName%" type="gml:ReferenceType"'
  [ 'minOccurs="0"' ] '>'
  '<xsd:annotation>'
  '<xsd:appinfo>'
  '<gml:targetElement>%ClassName%</gml:targetElement>'
```

```
</xsd:appinfo>
</xsd:annotation>
</xsd:element>'.

```

Beispiel

<p>INTERLIS</p>	<pre>DOMAIN Prozent100 = 0 .. 100; STRUCTURE Koernung = Ton : Prozent100; Schluff : Prozent100; Sand : Prozent100; END Koernung; CLASS Profil = Lage : COORD 480000.00 .. 850000.00, 60000.00 .. 320000.00; KoernungsklasseOberboden : Koernung; KoernungsklasseUnterboden : Koernung; Bodenpunktzahl : 0 .. 100; END Profil;</pre>
<p>GML</p>	<pre><xsd:simpleType name="Prozent100"> <xsd:restriction base="xsd:integer"> <xsd:minInclusive value="0"/> <xsd:maxInclusive value="100"/> </xsd:restriction> </xsd:simpleType> <xsd:element name="Koernung" type="KoernungType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="KoernungType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="Ton" type="Prozent100" minOccurs="0"/> <xsd:element name="Schluff" type="Prozent100" minOccurs="0"/> <xsd:element name="Sand" type="Prozent100" minOccurs="0"/> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Profil" type="ProfilType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="ProfilType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="Lage" minOccurs="0" type="gml:PointPropertyType"> </xsd:element> <xsd:element name="KoernungsklasseOberboden"> <xsd:complexType> <xsd:sequence> <xsd:element ref="Koernung"/> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> </xsd:element></pre>

	<pre> <xsd:element name="KoernungsklasseUnterboden"> <xsd:complexType> <xsd:sequence> <xsd:element ref="Koernung"/> </xsd:sequence> </xsd:complexType> </xsd:element> <xsd:element name="Bodenpunktzahl" minOccurs="0"> <xsd:simpleType> <xsd:restriction base="xsd:integer"> <xsd:minInclusive value="0"/> <xsd:maxInclusive value="100"/> </xsd:restriction> </xsd:simpleType> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> </pre>
<p>GML Instanz</p>	<pre> <Profil gml:id="o1"> <Lage> <gml:Point gml:id="o2" srsName="urn:ogc:def:crs:EPSG::21781"> <pos>754300.00 247600.00</pos> </gml:Point> </Lage> <KoernungsklasseOberboden> <Koernung gml:id="o3"> <Ton>30</Ton> <Schluff>20</Schluff> <Sand>50</Sand> </Koernung> </KoernungsklasseOberboden> <KoernungsklasseUnterboden> <Koernung gml:id="o4"> <Ton>30</Ton> <Schluff>10</Schluff> <Sand>60</Sand> </Koernung> </KoernungsklasseUnterboden> <Bodenpunktzahl>50</Bodenpunktzahl> </Profil> </pre>

7.14 Abbildung der INTERLIS-Typen auf XML-Schema-Typen

7.14.1 Übersicht

INTERLIS	Abbildung	Complex/Simple Type
TextType	xsd:normalizedString	Simple
EnumerationType	xsd:string / gml:CodeType	Simple/Complex
EnumTreeValueType	xsd:string / gml:CodeType	Simple/Complex
AlignmentType	xsd:string	Simple
BooleanType	xsd:boolean	Simple
NumericType	xsd:integer / xsd:decimal / xsd:double	Simple
FormattedType	xsd:string / xsd:date / xsd:time / xsd:dateTime	Simple
CoordinateType	gml:PointType	Complex
OIDType	xsd:int / xsd:token	Simple
BlackboxType	xsd:anyType / xsd:base64Binary	Complex/Simple
ClassType	xsd:string	Simple
AttributePathType	xsd:string	Simple
LineType	gml:CurveType / gml:PolygonType	Complex

7.14.2 Kodierung von Zeichenketten

Attribute/Wertebereiche vom Basistyp «TEXT» werden als `xsd:normalizedString`; «MTEXT» als `xsd:string`; «NAME» als `xsd:string` und «URI» als `xsd:anyURI` kodiert. Falls aus dem Modell ableitbar, wird auch `xsd:restriction` kodiert.

Beispiel	
INTERLIS	<pre>DOMAIN TestText = TEXT; TestTextLen = TEXT*10; TestMText = MTEXT; TestMTextLen = MTEXT*10; TestURI = URI; TestName = NAME;</pre>
GML	<pre><xsd:simpleType name="TestText"> <xsd:restriction base="xsd:normalizedString"> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestTextLen"> <xsd:restriction base="xsd:normalizedString"></pre>

	<pre> <xsd:maxLength value="10"/> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestMText"> <xsd:restriction base="xsd:string"> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestMTextLen"> <xsd:restriction base="xsd:string"> <xsd:maxLength value="10"/> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestURI"> <xsd:restriction base="xsd:anyURI"/> </xsd:simpleType> <xsd:simpleType name="TestName"> <xsd:restriction base="xsd:token"> <xsd:maxLength value="255"/> <xsd:pattern value="[a-zA-Z][a-zA-Z0-9_]*"/> </xsd:restriction> </xsd:simpleType> </pre>
--	--

7.14.3 Kodierung von Aufzählungen

In INTERLIS können Aufzählungen verfeinert und/oder erweitert werden. Beispiele:

```

Farbe = (rot, gelb, gruen);
FarbeVerfeinert EXTENDS Farbe = (rot (dunkelrot, orange, karmin));
FarbeErweitert EXTENDS Farbe = (schwarz, blau);

```

Da sich mit XML-Schema Aufzählungen weder verfeinern noch erweitern lassen, wird eine INTERLIS-Aufzählung nur dann in eine XML-Schema Aufzählung abgebildet, wenn (1) der Wertebereich oder das Attribut als «FINAL» markiert ist und (2) der Wertebereich oder das Attribut keine Erweiterung eines anderen Wertebereichs oder Attributs ist. Ansonsten wird die Aufzählung als `gml:Code` abgebildet. Nur durch die (unüblichen) Einschränkungen (1) und (2) wird ermöglicht, dass gängige XML-Software die in XML-Schema kodierten INTERLIS-Aufzählungen zur Validierung von GML-Instanzen beiziehen kann.

Für den Codelisten-Identifikator (`codeSpace`) der einzelnen Aufzählwerte kann der Namensraum-Identifikator des Modells (`%StdModelNamespaceIdentifizier%`; siehe Kapitel 7.6), ergänzt durch den abgebildeten Namen der Wertebereichsdefinition verwendet werden (Beispiel: «<http://www.interlis.ch/ILIGML-1.0/ModelA/Farbe>»). Wird die Aufzählung im Rahmen eines Attributs definiert, wird der Namensraum-Identifikator des Modells mit dem abgebildeten Klassennamen und Attributnamen ergänzt

(Beispiel: «<http://www.interlis.ch/ILIGML-1.0/ModelA/Person/HaarFarbe>»).

Mit dem Metaattribut `ili2.iligml10.identifizierCodeSpace` kann der Codelisten-Identifikator (der Werte für das XML-Attribut `gml:codeSpace` in der GML-Instanz) pro Wertebereich definiert werden.

Beispiel	
INTERLIS	<pre> MODEL ModelA ... DOMAIN FarbeFinal (FINAL) = (rot, gelb, gruen); Farbe = (rot, gelb, gruen); FarbePlus EXTENDS Farbe=(rot (dunkel,hell),gelb,gruen); CLASS Fahrzeug = CarrosserieFarbe : Farbe; END Fahrzeug; CLASS Auto EXTENDS Fahrzeug = CarrosserieFarbe (EXTENDED) : FarbePlus; END Auto; </pre>
GML	<pre> <xsd:simpleType name="FarbeFinal"> <xsd:restriction base="xsd:string"> <xsd:enumeration value="rot"/> <xsd:enumeration value="gelb"/> <xsd:enumeration value="gruen"/> </xsd:restriction> </xsd:simpleType> <xsd:complexType name="Farbe"> <xsd:restriction base="gml:CodeType"/> </xsd:complexType> <xsd:complexType name="FarbePlus"> <xsd:restriction base="gml:CodeType"/> </xsd:complexType> <xsd:element name="Fahrzeug" type="FahrzeugType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="FahrzeugType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="CarrosserieFarbe" type="Farbe" minOccurs="0"/> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Auto" type="AutoType" substitutionGroup="Fahrzeug"/> <xsd:complexType name="AutoType"> <xsd:complexContent> <xsd:extension base="FahrzeugType"> <xsd:sequence> <!-- CarrosserieFarbe ist schon in FahrzeugType definiert --> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <!-- Dictionary --> <gml:Dictionary gml:id="o1"> <gml:identifier codeSpace= </pre>

	<pre> "http://www.interlis.ch/ILIGML-1.0/ModelA" >Farbe</gml:identifizier> <gml:dictionaryEntry> <gml:Definition gml:id="o2"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/Farbe" >rot</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> <gml:dictionaryEntry> <gml:Definition gml:id="o3"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/Farbe" >gelb</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> <gml:dictionaryEntry> <gml:Definition gml:id="o4"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/Farbe" >gruen</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> </gml:Dictionary> <gml:Dictionary gml:id="o5"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA" >FarbePlus</gml:identifizier> <gml:dictionaryEntry> <gml:Definition gml:id="o6"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/FarbePlus" >rot.dunkel</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> <gml:dictionaryEntry> <gml:Definition gml:id="o7"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/FarbePlus" >rot.hell</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> <gml:dictionaryEntry> <gml:Definition gml:id="o8"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/FarbePlus" >gelb</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> <gml:dictionaryEntry> <gml:Definition gml:id="o9"> <gml:identifizier codeSpace= "http://www.interlis.ch/ILIGML-1.0/ModelA/FarbePlus" >gruen</gml:identifizier> </gml:Definition> </gml:dictionaryEntry> </gml:Dictionary> </pre>
<p>GML Instanz</p>	<pre> <Fahrzeug gml:id="o1"> <CarroserieFarbe>rot<CarroserieFarbe> </Fahrzeug> <Auto gml:id="o2"> <CarroserieFarbe>rot.dunkel<CarroserieFarbe> </Auto> </pre>

--	--

Die Aufzählungswerte werden in beiden Fällen wie folgt kodiert:

```
EnumValue = ( EnumElement-Name { '.' EnumElement-Name } ) .
```

Für die Kodierung von Aufzählungswerten (unabhängig davon, ob der Wertebereich nur die Blätter oder auch die Knoten umfasst) wird die Syntax für Aufzählungskonstanten angewendet (Regel «EnumValue»). Das Zeichen «#» wird dabei weggelassen.

Die vordefinierten Textausrichtungstypen «HALIGNMENT» und «VALIGNMENT» werden als Teil eines im Anhang gegebenen Basis-Schemas definiert. Für den Typ «BOOLEAN» wird der entsprechende XML-Schema Typ `xsd:boolean` verwendet.

7.14.4 Kodierung von numerischen Datentypen

Numerische Datentypen werden je nach Unter- und Obergrenzwert als `xsd:integer`, `xsd:decimal` oder `xsd:double` kodiert. Falls aus dem Modell ableitbar, wird auch `xsd:restriction` kodiert.

Beispiel	
INTERLIS	DOMAIN TestInt = 1 .. 10; TestDec = 1.0 .. 10.0; TestDouble = 0.123e1 .. 0.234e1;
GML	<pre> <xsd:simpleType name="TestInt"> <xsd:restriction base="xsd:integer"> <xsd:minInclusive value="1"/> <xsd:maxInclusive value="10"/> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestDec"> <xsd:restriction base="xsd:decimal"> <xsd:minInclusive value="1.0"/> <xsd:maxInclusive value="10.0"/> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="TestDouble"> <xsd:restriction base="xsd:double"> <xsd:minInclusive value="1.23"/> <xsd:maxInclusive value="2.34"/> </xsd:restriction> </xsd:simpleType> </pre>

Anmerkung: Man kann die Einheit im Schema nicht angeben. Mit «MeasureType» erreicht man nur, dass in den Daten die Einheit kodiert werden kann.

7.14.5 Kodierung von formatierten Wertebereichen

Formatierte Wertebereiche, die «INTERLIS.XMLDate», «INTERLIS.XMLTime» oder «INTERLIS.XMLDateTime» referenzieren, werden in die entsprechenden XML-Schema-Datumstypen (xsd:date, xsd:time, xsd:dateTime) abgebildet. Alle anderen formatierten Wertebereiche werden in ein xsd:string abgebildet.

Beispiel	
INTERLIS	<pre> STRUCTURE GregorianDate = Year: 1582 .. 2999; SUBDIVISION Month: 1 .. 12; SUBDIVISION Day: 1 .. 31; END GregorianDate; DOMAIN BuchungsDatum = FORMAT INTERLIS.XMLDate "2002-01-01" .. "2007-12-31"; StartZeit = FORMAT INTERLIS.XMLTime "00:00:00.000" .. "23:59:59.999"; MessZeitpunkt = FORMAT INTERLIS.XMLDateTime "2002-01-01T00:00:00.000" .. "2007-12-31T23:59:59.999"; EigenesDatum = FORMAT BASED ON GregorianDate (Year "Y" Month "M" Day "D"); </pre>
GML	<pre> <xsd:simpleType name="BuchungsDatum"> <xsd:restriction base="xsd:date"/> </xsd:simpleType> <xsd:simpleType name="StartZeit"> <xsd:restriction base="xsd:time"/> </xsd:simpleType> <xsd:simpleType name="MessZeitpunkt"> <xsd:restriction base="xsd:dateTime"/> </xsd:simpleType> <xsd:simpleType name="EigenesDatum"> <xsd:restriction base="xsd:string"/> </xsd:simpleType> </pre>

7.14.6 Kodierung von Gefässen

Die XML-Variante des Typs «BLACKBOX» wird als xsd:any kodiert, die binäre Variante als xsd:base64Binary.

Beispiel	
INTERLIS	<pre> DOMAIN BlackboxXml = BLACKBOX XML; BlackboxBinary = BLACKBOX BINARY; </pre>
GML	<pre> <xsd:complexType name="BlackboxXml"> <xsd:sequence> <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded" </pre>

	<pre> processContents="lax"/> </xsd:sequence> </xsd:complexType> <xsd:simpleType name="BlackboxBinary"> <xsd:restriction base="xsd:base64Binary"> </xsd:restriction> </xsd:simpleType> </pre>
--	--

7.14.7 Kodierung von Klassentypen

Der Klassentyp wird als `xsd:normalizedString` kodiert.

Beispiel

INTERLIS	<pre> DOMAIN InterlisClassRef = CLASS; </pre>
GML	<pre> <xsd:simpleType name="InterlisClassRef"> <xsd:restriction base="xsd:normalizedString"> </xsd:restriction> </xsd:simpleType> </pre>

Der Wert enthält den vollständig qualifizierten Klassen-, Struktur- oder Assoziationsnamen (z.B. «DM01AVCH24D.FixpunkteKategorie1.LFP1»).

7.14.8 Kodierung von Attributpfadtypen

Der Attributpfadtyp wird als `xsd:normalizedString` kodiert.

Beispiel

INTERLIS	<pre> DOMAIN InterlisAttributeRef = ATTRIBUTE; </pre>
GML	<pre> <xsd:simpleType name="InterlisAttributeRef"> <xsd:restriction base="xsd:normalizedString"> </xsd:restriction> </xsd:simpleType> </pre>

Der Wert enthält den vollständig qualifizierten Klassennamen gefolgt vom durch einen Punkt abgetrennten Attributnamen (z.B. «Grunddatensatz.Fixpunkte.LFP.Nummer»).

7.14.9 Kodierung von Koordinaten

Koordinaten werden als `gml:PointPropertyType` kodiert.

Beispiel

INTERLIS	<pre> DOMAIN LKoord = COORD 480000.00 .. 850000.00 [m] {CHLV03[1]}, 60000.00 .. 320000.00 [m] {CHLV03[2]}, ROTATION 2 -> 1; </pre>
----------	---

GML	<pre><xsd:complexType name="LKoord"> <xsd:complexContent> <xsd:restriction base="gml:PointPropertyType"> <xsd:sequence> <xsd:element ref="gml:Point"/> </xsd:sequence> </xsd:restriction> </xsd:complexContent> </xsd:complexType></pre>
------------	--

Angaben zu Achsen-Wertebereich, Einheit, Referenzsystem und Richtungssinn lassen sich im GML-Applikationsschema nicht festhalten.

Falls im INTERLIS Modell kein Referenzsystem definiert wurde, soll in der GML-Instanz ein werkzeugspezifischer Default-Wert für das XML-Attribut srsName verwendet werden. Falls im INTERLIS Modell ein Referenzsystem definiert wurde, soll in der GML-Instanz ein Wert der diesem Referenzsystem entspricht, verwendet werden. Der Wert kann bei der Deklaration des Referenzsystemobjektes mit Hilfe des Metattributes «ili2.iligml10.srsName» definiert werden. Falls kein Metattribut definiert ist, ist die Abbildung werkzeugspezifisch.

Beispiel	
INTERLIS	<pre>REFSYSTEM BASKET BCoordSys ~ CoordSys.CoordsysTopic OBJECTS OF GeoCartesian2D : !!@ ili2.iligml10.srsName="urn:ogc:def:crs:EPSG::21781" CHLV03;</pre>

7.14.10 Kodierung von Linienzügen

Linienzüge werden als `gml:CurvePropertyType` kodiert.

Beispiel	
INTERLIS	<pre>DOMAIN LKoord = COORD 480000.00 .. 850000.00 [m] {CHLV03[1]}, 60000.00 .. 320000.00 [m] {CHLV03[2]}, ROTATION 2 -> 1; Strassenachse = POLYLINE WITH (ARCS,STRAIGHTS) VERTEX LKoord WITHOUT OVERLAPS>0.10;</pre>
GML	<pre><xsd:complexType name="LKoord"> <xsd:complexContent> <xsd:restriction base="gml:PointPropertyType"> <xsd:sequence> <xsd:element ref="gml:Point"/> </xsd:sequence> </xsd:restriction> </xsd:complexContent> </xsd:complexType> <xsd:complexType name="Strassenachse"> <xsd:complexContent> <xsd:restriction base="gml:CurvePropertyType"> <xsd:sequence> <xsd:element ref="gml:AbstractCurve"/> </xsd:sequence> </xsd:restriction></pre>

	<pre></xsd:complexContent> </xsd:complexType></pre>
--	---

Angaben zu Stützpunkt-Wertebereich und zulässigen Kurvenstück-Formen lassen sich im GML-Applikationsschema nicht festhalten.

7.14.11 Kodierung von Einzelflächen

Einzelflächen werden als `gml:SurfacePropertyType` kodiert. In der Instanz ist dann aber nur `gml:Polygon` zulässig.

Beispiel	
INTERLIS	<pre>DOMAIN LKoord = COORD 480000.00 .. 850000.00 [m] {CHLV03[1]}, 60000.00 .. 320000.00 [m] {CHLV03[2]}, ROTATION 2 -> 1; GebaeudeFlaeche = SURFACE WITH (ARCS,STRAIGHTS) VERTEX LKoord WITHOUT OVERLAPS>0.10;</pre>
GML	<pre><xsd:complexType name="LKoord"> <xsd:complexContent> <xsd:restriction base="gml:PointPropertyType"> <xsd:sequence> <xsd:element ref="gml:Point"/> </xsd:sequence> </xsd:restriction> </xsd:complexContent> </xsd:complexType> <xsd:complexType name="GebaeudeFlaeche"> <xsd:complexContent> <xsd:restriction base="gml:SurfacePropertyType"> <xsd:sequence> <xsd:element ref="gml:Polygon"/> </xsd:sequence> </xsd:restriction> </xsd:complexContent> </xsd:complexType></pre>

Angaben zu Stützpunkt-Wertebereich und zulässigen Kurvenstück-Formen lassen sich im GML-Applikationsschema nicht festhalten.

Falls Linienattribute vorkommen, müssen entsprechende Erweiterungen zu `gml:Curve` generiert werden.

7.14.12 Kodierung von Gebietseinteilungen

Für jedes Attribut mit einem «AREA»-Wertebereich wird ein `gml:PointPropertyType` kodiert: der Gebietsreferenzpunkt.

Für jedes Attribut mit einem «AREA»-Wertebereich wird zusätzlich ein Feature-Typ generiert: die «Randlinientabelle». Dieser Feature-Typ enthält zwei Attribute: «geometry» vom Typ `gml:CurvePropertyType` für die einzelnen Linienstücke; und «lineattr» für die Linienattribute, falls der INTERLIS-Wertebereich solche enthält.

Beispiel	
INTERLIS	<pre> DOMAIN LKoord = COORD 480000.00 .. 850000.00 [m] {CHLV03[1]}, 60000.00 .. 320000.00 [m] {CHLV03[2]}, ROTATION 2 -> 1; STRUCTURE GrenzlinieEigenschaften = streitig : BOOLEAN; END GrenzlinieEigenschaften; CLASS Grundstueck = Grenze : MANDATORY AREA WITH (ARCS,STRAIGHTS) VERTEX LKoord WITHOUT OVERLAPS>0.10 LINE ATTRIBUTES GrenzlinieEigenschaften; END Grundstueck; </pre>
GML	<pre> <xsd:complexType name="LKoord"> <xsd:complexContent> <xsd:restriction base="gml:PointPropertyType"> <xsd:sequence> <xsd:element ref="gml:Point"/> </xsd:sequence> </xsd:restriction> </xsd:complexContent> </xsd:complexType> <xsd:element name="GrenzlinieEigenschaft" type="GrenzlinieEigenschaftType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="GrenzlinieEigenschaftType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="streitig" type="xsd:boolean" minOccurs="0"/> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Grundstueck.Grenze" type="Grundstueck.GrenzeType" substitutionGroup="gml:AbstractFeature"/> <xsd:complexType name="Grundstueck.GrenzeType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="geometry" type="gml:CurvePropertyType"/> <xsd:element name="lineattr"> <xsd:complexType> <xsd:sequence> <xsd:element ref="GrenzlinieEigenschaft"/> </xsd:sequence> <xsd:attributeGroup ref="gml:OwnershipAttributeGroup"/> </xsd:complexType> </xsd:element> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType> <xsd:element name="Grundstueck" type="GrundstueckType" substitutionGroup="gml:AbstractFeature"/> </pre>

	<pre><xsd:complexType name="GrundstueckType"> <xsd:complexContent> <xsd:extension base="gml:AbstractFeatureType"> <xsd:sequence> <xsd:element name="Grenze" type="LKoord"/> </xsd:sequence> </xsd:extension> </xsd:complexContent> </xsd:complexType></pre>
<p>GML Instanz</p>	<pre><Grundstueck.Grenze gml:id="o0"> <geometry> <gml:LineString gml:id="g0" srsName="urn:ogc:def:crs:EPSG::21781" srsDimension="2"> <gml:pos>146.92 174.98</gml:pos> <gml:pos>163.64 185.96</gml:pos> <gml:pos>158.15 194.31</gml:pos> <gml:pos>149.79 188.82</gml:pos> <gml:pos>147.04 193</gml:pos> <gml:pos>138.68 187.51</gml:pos> <gml:pos>146.92 174.98</gml:pos> </gml:LineString> </geometry> <lineattr> <GrenzlinieEigenschaft gml:id="s0"> <streitig>false</streitig> <GrenzlinieEigenschaft> </lineattr> </Grundstueck.Grenze> <Grundstueck gml:id="o1"> <Grenze> <gml:Point gml:id="g1" srsName="urn:ogc:def:crs:EPSG::21781" srsDimension="2"> <gml:pos>147.00 173.00 </gml:pos> </gml:Point> </Grenze> </Grundstueck></pre>

Angaben zu Stützpunkt-Wertebereich und zulässigen Kurvenstück-Formen lassen sich im GML-Applikationsschema nicht festhalten, ebenso wenig, dass es sich um eine Gebietseinteilung handelt.

7.14.13 Kodierung von Typen zur Objektidentifikation

Typen für Objektidentifikationen werden als `xsd:int` («OID NumericType») oder `xsd:token` («OID» als «ANY» oder «TextType») abgebildet.

Mit dem Metaattribut «`ili2.iligml10.identifizierCodeSpace`» kann die Formatierung des Identifikatorwertes (der Werte für das XML-Attribut `gml:codeSpace` in der GML-Instanz) definiert werden.

Mit dem Metaattribut «`ili2.iligml10.identifizierPattern`» kann die Formatierung des Identifikatorwertes (der Werte für das XML-Element `gml:identifizier` in der GML-Instanz) definiert werden.

Mit dem Metaattribut «`ili2.iligml10.hrefPattern`» kann die Formatierung des Referenzwertes (der Werte für das XML-Attribut `xlink:href` in der GML-Instanz) definiert werden.

In den Musterdefinitionen kann «`${value}`» als Platzhalter für den OID-Wert verwendet werden.

Beispiel	
INTERLIS	<pre> DOMAIN BFS_EGID = OID 1 .. 9999999999; UUID = OID TEXT*36; !!@ ili2.iligml10.identifizierCodeSpace= "http://inspire.jrc.ec.europa.eu/" !!@ ili2.iligml10.identifizierPattern= "urn:inspire:object:id:\${value}" !!@ ili2.iligml10.hrefPattern= "urn:inspire:object:id:\${value}" InspiredId = OID TEXT; !! z.B. "DENW1000:A00012ab" </pre>
GML	<pre> <xsd:simpleType name="BFS_EGID"> <xsd:restriction base="xsd:int"> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="UUID"> <xsd:restriction base="xsd:token"> </xsd:restriction> </xsd:simpleType> <xsd:simpleType name="InspiredId"> <xsd:restriction base="xsd:token"> </xsd:restriction> </xsd:simpleType> </pre>

7.14.14 Kodierung von weiteren Kurvenstück-Formen

Wird in dieser Version dieser Spezifikation nicht unterstützt.

Anmerkung: Es muss zuerst ein INTERLIS-Basismodell erstellt werden, so dass eine Abbildung auf in GML vordefinierte Kurventypen möglich ist.

8 Instanz-Kodierungsregeln

Zusätzlich zu den Schema-Codierungsregeln gelten die in diesem Abschnitt definierten Instanz-Regeln.

8.1 Kodierung mehrerer Behälter (TOPICs) in einer Transferdatei

Werden in einer Transferdatei mehrere Behälter transferiert (was typischerweise der Fall ist) wird als Wurzel-XML-Element «TRANSFER» verwendet. Dieses Element und die zugehörigen Typen sind im Anhang A definiert.

8.2 Kodierung von Objekten und Strukturelementen

Objekte werden immer inline zu einem Behälter kodiert. Objekte werden nie inline zu einem anderen Objekt kodiert.

Strukturelemente werden immer inline zu einem Objekt oder anderen Strukturelement kodiert. Strukturelemente werden nie referenziert, obwohl sie eine `gml:id` erhalten.

8.3 Kodierung von Objekt- und Behälteridentifikationen

Die Transferidentifikation eines Objektes oder Behälters wird als `gml:id` kodiert.

Die stabile Identifikation des Objektes oder Behälters wird als `gml:identifizier` kodiert.

Falls bei der Wertebereichsdefinition des OID für das Metaattribut «`ili2.iligml10.identifizierCodeSpace`» kein Wert definiert wurde, wird als «codeSpace» der Namensraum-Identifikator des Modells (`%StdModelNamespaceIdentifizier%`; siehe Kapitel 7.6), ergänzt durch den abgebildeten Namen der Wertebereichsdefinition für die OIDs verwendet (Beispiel für den Wertebereich «UUIDOID» aus dem Modell INTERLIS (Anhang A im Referenzhandbuch [1]): «`http://www.interlis.ch/ILIGML-1.0/INTERLIS/UUIDOID`»).

Falls bei der Wertebereichsdefinition des OID für das Metaattribut «`ili2.iligml10.identifizierCodeSpace`» ein Wert definiert wurde, wird dieser verwendet.

Falls bei der Wertebereichsdefinition des OID für das Metaattribut «`ili2.iligml10.identifizierPattern`» kein Wert definiert wurde, wird der Wert unverändert verwendet (das entspricht dem Muster «`#{value}`»).

Falls bei der Wertebereichsdefinition des OID für das Metaattribut «`ili2.iligml10.identifizierPattern`» ein Wert definiert wurde, wird dieser verwendet.

8.4 Kodierung von Beziehungen

Als Referenzwert wird immer die `gml:id` verwendet (z.B. `xlink:href="#o1` als Verweis auf ein Objekt mit `gml:id="o1"`), ausser wenn das referenzierte Objekt eine stabile OID hat.

Wenn das referenzierte Objekt eine stabile OID hat, wird diese für die Referenz verwendet.

Falls bei der Wertebereichsdefinition für den OID kein Muster zur Bildung des `xlink:href`-Werts definiert wurde, wird `«urn:x-ili:${value}»` als Muster verwendet (z.B. `xlink:href="urn:x-ili:o1"` als Verweis auf ein Objekt mit `gml:identifier="o1"`).

Falls bei der Wertebereichsdefinition für den OID ein Muster zur Bildung des `xlink:href`-Werts definiert wurde, wird dieses Muster verwendet,

8.5 Kodierung von Geometrien

Alle Geometrien müssen inline kodiert werden, auch wenn das GML-Basischema Referenzen zulassen würde.

Bei «SURFACE» und «AREA» ist nur `gml:Polygon` zulässig (`gml:Surface` ist also *nicht* zulässig).

Bei `gml:LineStringSegment` ist in GML-SF nur `gml:posList` zulässig.

Bei `gml:exterior/gml:interior` ist in GML-SF nur `gml:LinearRing` (mit `gml:posList`) zulässig.

9 Haftungsausschluss/Hinweise auf Rechte Dritter

eCH-Standards, welche der Verein **eCH** dem Benutzer zur unentgeltlichen Nutzung zur Verfügung stellt, oder welche **eCH** referenziert, haben nur den Status von Empfehlungen. Der Verein **eCH** haftet in keinem Fall für Entscheidungen oder Massnahmen, welche der Benutzer auf Grund dieser Dokumente trifft und / oder ergreift. Der Benutzer ist verpflichtet, die Dokumente vor deren Nutzung selbst zu überprüfen und sich gegebenenfalls beraten zu lassen. **eCH**-Standards können und sollen die technische, organisatorische oder juristische Beratung im konkreten Einzelfall nicht ersetzen.

In **eCH**-Standards referenzierte Dokumente, Verfahren, Methoden, Produkte und Standards sind unter Umständen markenrechtlich, urheberrechtlich oder patentrechtlich geschützt. Es liegt in der ausschliesslichen Verantwortlichkeit des Benutzers, sich die allenfalls erforderlichen Rechte bei den jeweils berechtigten Personen und/oder Organisationen zu beschaffen.

Obwohl der Verein **eCH** all seine Sorgfalt darauf verwendet, die **eCH**-Standards sorgfältig auszuarbeiten, kann keine Zusicherung oder Garantie auf Aktualität, Vollständigkeit, Richtigkeit bzw. Fehlerfreiheit der zur Verfügung gestellten Informationen und Dokumente gegeben werden. Der Inhalt von **eCH**-Standards kann jederzeit und ohne Ankündigung geändert werden.

Jede Haftung für Schäden, welche dem Benutzer aus dem Gebrauch der **eCH**-Standards entstehen ist, soweit gesetzlich zulässig, wegbedungen.

10 Urheberrechte

Wer **eCH**-Standards erarbeitet, behält das geistige Eigentum an diesen. Allerdings verpflichtet sich der Erarbeitende sein betreffendes geistiges Eigentum oder seine Rechte an geistigem Eigentum anderer, sofern möglich, den jeweiligen Fachgruppen und dem Verein **eCH** kostenlos zur uneingeschränkten Nutzung und Weiterentwicklung im Rahmen des Vereinszweckes zur Verfügung zu stellen.

Die von den Fachgruppen erarbeiteten Standards können unter Nennung der jeweiligen Urheber von **eCH** unentgeltlich und uneingeschränkt genutzt, weiterverbreitet und weiterentwickelt werden.

eCH-Standards sind vollständig dokumentiert und frei von lizenz- und/oder patentrechtlichen Einschränkungen. Die dazugehörige Dokumentation kann unentgeltlich bezogen werden.

Diese Bestimmungen gelten ausschliesslich für die von **eCH** erarbeiteten Standards, nicht jedoch für Standards oder Produkte Dritter, auf welche in den **eCH**-Standards Bezug genommen wird. Die Standards enthalten die entsprechenden Hinweise auf die Rechte Dritter.

Anhang A – Vordefiniertes Basis-Schema

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.interlis.ch/ILIGML-1.0/INTERLIS"
  xmlns:ili2="http://www.interlis.ch/ili2"
  targetNamespace=
    "http://www.interlis.ch/ILIGML-1.0/INTERLIS"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  xmlns:gml="http://www.opengis.net/gml/3.2">

  <xsd:annotation>
    <xsd:appinfo source="http://www.interlis.ch/ili2c">
      <ili2:model>INTERLIS</ili2:model>
      <ili2:modelVersion>2005-06-16</ili2:modelVersion>
      <ili2:modelAt>http://www.interlis.ch</ili2:modelAt>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:import namespace="http://www.opengis.net/gml/3.2"/>

  <xsd:attribute name="ORDER_POS" type="xsd:positiveInteger"/>

  <xsd:simpleType name="HALIGNMENT">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Left"/>
      <xsd:enumeration value="Center"/>
      <xsd:enumeration value="Right"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="VALIGNMENT">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Top"/>
      <xsd:enumeration value="Cap"/>
      <xsd:enumeration value="Half"/>
      <xsd:enumeration value="Base"/>
      <xsd:enumeration value="Bottom"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="I320ID">
    <xsd:restriction base="xsd:int">
      <xsd:minInclusive value="0"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="STANDARD0ID">
    <xsd:restriction base="xsd:token">
      <xsd:length value="16"/>
      <xsd:pattern value="[a-zA-Z][a-zA-Z0-9]*"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="UUI0ID">
    <xsd:restriction base="xsd:token">
      <xsd:length value="36"/>
      <xsd:pattern value=
        "[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}"/>
    </xsd:restriction>
  </xsd:simpleType>

```

```
<xsd:complexType name="TRANSFERMemberType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureMemberType">
      <xsd:sequence>
        <xsd:choice>
          <xsd:element ref="gml:AbstractFeature"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name="TRANSFER" type="TRANSFERType"
  substitutionGroup="gml:AbstractFeature"/>

<xsd:complexType name="TRANSFERType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="baskets"
          type="TRANSFERMemberType"
          minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attributeGroup ref="gml:AggregationAttributeGroup"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
</xsd:schema>
```

Anhang B – Beispiel

INTERLIS 2-Modell

```
INTERLIS 2.3;

MODEL Beispiel (de)
AT "mailto:ceis@localhost"
VERSION "2008-03-31" =

DOMAIN

    LKoord = COORD 100.00 .. 300.00, 100.00 .. 300.00;

TOPIC Bodenbedeckung =

    CLASS BoFlaechen =
        Art (FINAL): MANDATORY (
            Gebaeude,
            befestigt,
            humusiert,
            Gewaesser,
            bestockt,
            vegetationslos);
        Form : MANDATORY AREA WITH (ARCS,STRAIGHTS)
            VERTEX Beispiel.LKoord
            WITHOUT OVERLAPS>0.10;
    END BoFlaechen;

    CLASS Strasse =
        Achse : MANDATORY POLYLINE WITH (ARCS,STRAIGHTS)
            VERTEX LKoord;
    END Strasse;

    CLASS Gebaeude =
        PositionHauseingang : LKoord;
        AssNr : MANDATORY TEXT*6;
        UNIQUE AssNr;
    END Gebaeude;

    ASSOCIATION GebaeudeFlaeche =
        Gebaeude -- {0..*} Gebaeude;
        Flaeche -- {1} BoFlaechen;
    END GebaeudeFlaeche;

END Bodenbedeckung;

END Beispiel.
```

INTERLIS 1-Modell

```

TRANSFER Beispiel;

DOMAIN
  LKoord = COORD2 100.00 100.00
              300.00 300.00;

MODEL Beispiel

TOPIC Bodenbedeckung =
  TABLE BoFlaechen =
    Art: (Gebaeude,
          befestigt,
          humusiert,
          Gewaesser,
          bestockt,
          vegetationslos);
    Form: AREA WITH (STRAIGHTS, ARCS) VERTEX LKoord
           WITHOUT OVERLAPS > 0.10;
  NO IDENT
END BoFlaechen;

TABLE Strasse =
  Achse : POLYLINE WITH (ARCS,STRAIGHTS)
           VERTEX LKoord;
  NO IDENT
END Strasse;

TABLE Gebaeude =
  PositionHauseingang : LKoord;
  AssNr: TEXT*6;
  Flaechen: -> BoFlaechen // Art = Gebaeude //;
IDENT
  AssNr; !! Annahme AssNr sei eindeutig.
  Flaechen; !! Dem Gebaeude ist genau eine
             !! Flaechen zugeordnet
END Gebaeude;

END Bodenbedeckung.

END Beispiel.

FORMAT FREE;
CODE BLANK = DEFAULT, UNDEFINED = DEFAULT, CONTINUE = DEFAULT;
TID = ANY;

END.

```

GML-Applikationsschema

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.interlis.ch/ILIGML-1.0/Beispiel"
  targetNamespace="http://www.interlis.ch/ILIGML-1.0/Beispiel"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  xmlns:gml="http://www.opengis.net/gml/3.2">
  <xsd:annotation>
    <xsd:appinfo source="http://www.interlis.ch/ili2c"
      xmlns:ili2="http://www.interlis.ch/ili2">
      <ili2:model>Beispiel</ili2:model>
      <ili2:modelVersion>2008-03-31</ili2:modelVersion>
      <ili2:modelAt>mailto:ceis@localhost</ili2:modelAt>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:import namespace="http://www.opengis.net/gml/3.2"/>
  <xsd:complexType name="LKoord">
    <xsd:complexContent>
      <xsd:restriction base="gml:PointPropertyType">
        <xsd:sequence>
          <xsd:element ref="gml:Point"/>
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="BoFlaechen.Form" type="BoFlaechen.FormType"
    substitutionGroup="gml:AbstractFeature"/>
  <xsd:complexType name="BoFlaechen.FormType">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractFeatureType">
        <xsd:sequence>
          <xsd:element name="geometry"
            type="gml:CurvePropertyType"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="BoFlaechen" type="BoFlaechenType"
    substitutionGroup="gml:AbstractFeature"/>
  <xsd:complexType name="BoFlaechenType">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractFeatureType">
        <xsd:sequence>
          <xsd:element name="Art">
            <xsd:simpleType>
              <xsd:restriction base="xsd:normalizedString">
                <xsd:enumeration value="Gebaeude"/>
                <xsd:enumeration value="befestigt"/>
                <xsd:enumeration value="humusiert"/>
                <xsd:enumeration value="Gewaesser"/>
                <xsd:enumeration value="bestockt"/>
                <xsd:enumeration value="vegetationslos"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="Form" type="LKoord"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="Strasse" type="StrasseType"
    substitutionGroup="gml:AbstractFeature"/>

```

```

<xsd:complexType name="StrasseType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="Achse" type="gml:CurvePropertyType"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="Gebaeude" type="GebaeudeType"
  substitutionGroup="gml:AbstractFeature"/>
<xsd:complexType name="GebaeudeType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="PositionHauseingang"
          type="LKoord"/>
        <xsd:element name="AssNr">
          <xsd:simpleType>
            <xsd:restriction base="xsd:normalizedString">
              <xsd:maxLength value="6"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="Flaeche" type="gml:ReferenceType">
          <xsd:annotation>
            <xsd:appinfo>
              <gml:targetElement>BoFlaechen</gml:targetElement>
            </xsd:appinfo>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BodenbedeckungMemberType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureMemberType">
      <xsd:sequence>
        <xsd:choice>
          <xsd:element ref="BoFlaechen"/>
          <xsd:element ref="BoFlaechen.Form"/>
          <xsd:element ref="Strasse"/>
          <xsd:element ref="Gebaeude"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="Bodenbedeckung" type="BodenbedeckungType"
  substitutionGroup="gml:AbstractFeature"/>
<xsd:complexType name="BodenbedeckungType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="member"
          type="BodenbedeckungMemberType"
          minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attributeGroup ref="gml:AggregationAttributeGroup"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
</xsd:schema>

```

INTERLIS 1-Transferdaten

```

SCNT
Beispiel Transfer-File
////
MTID Beispiel
MODL Beispiel
TOPI Bodenbedeckung
TABL BoFlaechen_Form
OBJE 1
STPT 146.92 174.98
LIPT 138.68 187.51
LIPT 147.04 193.00
LIPT 149.79 188.82
LIPT 158.15 194.31
LIPT 163.64 185.96
LIPT 146.92 174.98
ELIN
OBJE 2
STPT 140.69 156.63
LIPT 118.19 179.82
LIPT 113.00 219.97
LIPT 148.30 228.97
LIPT 186.38 206.82
ELIN
OBJE 3
STPT 186.38 206.82
ARCP 183.26 188.19
LIPT 170.18 176.00
LIPT 140.69 156.63
ELIN
OBJE 4
STPT 186.38 206.82
LIPT 194.26 208.19
ARCP 190.75 185.21
LIPT 174.10 169.00
LIPT 145.08 149.94
LIPT 140.69 156.63
ELIN
ETAB
TABL BoFlaechen
OBJE 10 0 148.20 183.48
OBJE 20 1 168.27 170.85
OBJE 30 2 133.95 206.06
ETAB
TABL Strasse
OBJE 100
STPT 190.26 208.00
ARCP 187.00 186.00
LIPT 173.10 171.00
LIPT 141.08 152.94
ELIN
ETAB
TABL Gebaeude
OBJE 40 148.41 175.96 958 10
ETAB
ETOP
EMOD
ENDE

```

INTERLIS 2-Transferdaten («INTERLIS-XML»)

```

<?xml version='1.0' encoding='UTF-8'?>
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3">
<HEADERSECTION SENDER="ceis" VERSION="2.3">
  <MODELS>
    <MODEL NAME="Beispiel" VERSION="2008-03-31"
      URI="mailto:ceis@localhost" />
  </MODELS>
</HEADERSECTION>
<DATASECTION>
<Beispiel.Bodenbedeckung BID="itf0">
<Beispiel.Bodenbedeckung.Gebaeude TID="40">
  <PositionHauseingang>
    <COORD><C1>148.41</C1><C2>175.96</C2></COORD>
  </PositionHauseingang>
  <AssNr>958</AssNr><Flaeche REF="10" />
</Beispiel.Bodenbedeckung.Gebaeude>
<Beispiel.Bodenbedeckung.BoFlaechen TID="30">
  <Art>humusiert</Art>
  <Form><SURFACE><BOUNDARY>
    <POLYLINE>
      <COORD><C1>140.69</C1><C2>156.63</C2></COORD>
      <COORD><C1>170.18</C1><C2>176.0</C2></COORD>
      <ARC>
        <C1>186.38</C1><C2>206.82</C2><A1>183.26</A1><A2>188.19</A2>
      </ARC>
      <COORD><C1>148.3</C1><C2>228.97</C2></COORD>
      <COORD><C1>113.0</C1><C2>219.97</C2></COORD>
      <COORD><C1>118.19</C1><C2>179.82</C2></COORD>
      <COORD><C1>140.69</C1><C2>156.63</C2></COORD>
    </POLYLINE>
  </BOUNDARY>
</BOUNDARY>
  <POLYLINE>
    <COORD><C1>146.92</C1><C2>174.98</C2></COORD>
    <COORD><C1>163.64</C1><C2>185.96</C2></COORD>
    <COORD><C1>158.15</C1><C2>194.31</C2></COORD>
    <COORD><C1>149.79</C1><C2>188.82</C2></COORD>
    <COORD><C1>147.04</C1><C2>193.0</C2></COORD>
    <COORD><C1>138.68</C1><C2>187.51</C2></COORD>
    <COORD><C1>146.92</C1><C2>174.98</C2></COORD>
  </POLYLINE>
</BOUNDARY></SURFACE></Form>
</Beispiel.Bodenbedeckung.BoFlaechen>
<Beispiel.Bodenbedeckung.BoFlaechen TID="20">
  <Art>befestigt</Art>
  <Form><SURFACE><BOUNDARY>
    <POLYLINE>
      <COORD><C1>140.69</C1><C2>156.63</C2></COORD>
      <COORD><C1>145.08</C1><C2>149.94</C2></COORD>
      <COORD><C1>174.1</C1><C2>169.0</C2></COORD>
      <ARC>
        <C1>194.26</C1><C2>208.19</C2><A1>190.75</A1><A2>185.21</A2>
      </ARC>
      <COORD><C1>186.38</C1><C2>206.82</C2></COORD>
      <ARC>
        <C1>170.18</C1><C2>176.0</C2><A1>183.26</A1><A2>188.19</A2>
      </ARC>
      <COORD><C1>140.69</C1><C2>156.63</C2></COORD>
    </POLYLINE>
  </BOUNDARY></SURFACE></Form>
</Beispiel.Bodenbedeckung.BoFlaechen>

```

```

<Beispiel.Bodenbedeckung.BoFlaechen TID="10">
  <Art>Gebaeude</Art>
  <Form><SURFACE><BOUNDARY>
    <POLYLINE>
      <COORD><C1>146.92</C1><C2>174.98</C2></COORD>
      <COORD><C1>163.64</C1><C2>185.96</C2></COORD>
      <COORD><C1>158.15</C1><C2>194.31</C2></COORD>
      <COORD><C1>149.79</C1><C2>188.82</C2></COORD>
      <COORD><C1>147.04</C1><C2>193.0</C2></COORD>
      <COORD><C1>138.68</C1><C2>187.51</C2></COORD>
      <COORD><C1>146.92</C1><C2>174.98</C2></COORD>
    </POLYLINE>
  </BOUNDARY></SURFACE></Form>
</Beispiel.Bodenbedeckung.BoFlaechen>
<Beispiel.Bodenbedeckung.Strasse TID="100">
  <Achse>
    <POLYLINE>
      <COORD><C1>190.26</C1><C2>208.00</C2></COORD>
      <ARC>
        <C1>173.10</C1><C2>171.00</C2><A1>187.00</A1><A2>186.00</A2>
      </ARC>
      <COORD><C1>141.08</C1><C2>152.94</C2></COORD>
    </POLYLINE>
  </Achse>
</Beispiel.Bodenbedeckung.Strasse>
</Beispiel.Bodenbedeckung>
</DATASECTION>
</TRANSFER>

```

GML-Daten

```

<?xml version="1.0" encoding="UTF-8"?>
<ili:TRANSFER
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ili="http://www.interlis.ch/ILIGML-1.0/INTERLIS"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  gml:id="tid">
  <ili:baskets>
  <ceis:Bodenbedeckung
    xmlns:ceis="http://www.interlis.ch/ILIGML-1.0/Beispiel"
    gml:id="bid">
    <ceis:member>
      <ceis:Gebaeude gml:id="x50">
        <ceis:PositionHauseingang>
          <gml:Point gml:id="g0" srsName="urn:ogc:def:crs:EPSG::21781"
srsDimension="2">
            <gml:pos>148.41 175.96</gml:pos>
          </gml:Point>
        </ceis:PositionHauseingang>
        <ceis:AssNr>958</ceis:AssNr>
        <ceis:Flaeche xlink:href="#x10"/>
      </ceis:Gebaeude>
    </ceis:member>
    <ceis:member>
      <ceis:Strasse gml:id="x40">
        <ceis:Achse>
          <gml:Curve gml:id="g40.1">
            <gml:segments>
              <gml:LineStringSegment interpolation="linear">
                <gml:posList>190.26 208.00</gml:posList>

```

```

        </gml:LineStringSegment>
        <gml:Arc numArc="1" interpolation="circularArc3Points">
          <gml:posList>190.26 208.00 187.00 186.00 173.10
171.00</gml:posList>
        </gml:Arc>
        <gml:LineStringSegment interpolation="linear">
          <gml:posList>173.10 171.00 141.08 152.94</gml:posList>
        </gml:LineStringSegment>
      </gml:segments>
    </gml:Curve>
  </ceis:Achse>
</ceis:Strasse>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen.Form gml:id="g1">
    <ceis:geometry>
      <gml:Curve gml:id="g1.1" srsName="urn:ogc:def:crs:EPSG::21781"
srsDimension="2">
        <gml:segments>
          <gml:LineStringSegment interpolation="linear">
            <gml:posList>140.69 156.63 170.18 176</gml:posList>
          </gml:LineStringSegment>
          <gml:Arc numArc="1" interpolation="circularArc3Points">
            <gml:posList>170.18 176.0 183.26 188.19 186.38
206.82</gml:posList>
          </gml:Arc>
        </gml:segments>
      </gml:Curve>
    </ceis:geometry>
  </ceis:BoFlaechen.Form>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen.Form gml:id="g2">
    <ceis:geometry>
      <gml:Curve gml:id="g2.1" srsName="urn:ogc:def:crs:EPSG::21781"
srsDimension="2">
        <gml:segments>
          <gml:LineStringSegment interpolation="linear">
            <gml:pos>146.92 174.98</gml:pos>
            <gml:pos>163.64 185.96</gml:pos>
            <gml:pos>158.15 194.31</gml:pos>
            <gml:pos>149.79 188.82</gml:pos>
            <gml:pos>147.04 193</gml:pos>
            <gml:pos>138.68 187.51</gml:pos>
            <gml:pos>146.92 174.98</gml:pos>
          </gml:LineStringSegment>
        </gml:segments>
      </gml:Curve>
    </ceis:geometry>
  </ceis:BoFlaechen.Form>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen.Form gml:id="g3">
    <ceis:geometry>
      <gml:Curve gml:id="g3.1">
        <gml:segments>
          <gml:LineStringSegment interpolation="linear">
            <gml:posList>140.69 156.63 145.08 149.94 174.1
169.0</gml:posList>
          </gml:LineStringSegment>
          <gml:Arc numArc="1" interpolation="circularArc3Points">
            <gml:posList>174.1 169.0 190.75 185.21 194.26
208.19</gml:posList>
          </gml:Arc>
        </gml:segments>
      </gml:Curve>
    </ceis:geometry>
  </ceis:BoFlaechen.Form>
</ceis:member>
</ceis:member>

```

```

        <gml:posList>194.26 208.19 186.38 206.82</gml:posList>
      </gml:LineStringSegment>
    </gml:segments>
  </gml:Curve>
</ceis:geometry>
</ceis:BoFlaechen.Form>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen.Form gml:id="g4">
    <ceis:geometry>
      <gml:Curve gml:id="g4.1">
        <gml:segments>
          <gml:LineStringSegment interpolation="linear">
            <gml:posList>186.38 206.82 148.3 228.97 113 219.97
118.19 179.82 140.69 156.63</gml:posList>
          </gml:LineStringSegment>
        </gml:segments>
      </gml:Curve>
    </ceis:geometry>
  </ceis:BoFlaechen.Form>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen gml:id="x30">
    <ceis:Art>humusiert</ceis:Art>
    <ceis:Form>
      <gml:Point gml:id="g8.1" srsName="urn:ogc:def:crs:EPSG::21781"
srsDimension="2">
        <gml:pos>148.20 183.48</gml:pos>
      </gml:Point>
    </ceis:Form>
  </ceis:BoFlaechen>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen gml:id="x20">
    <ceis:Art>befestigt</ceis:Art>
    <ceis:Form>
      <gml:Point gml:id="g20.1"
srsName="urn:ogc:def:crs:EPSG::21781" srsDimension="2">
        <gml:pos>168.27 170.85</gml:pos>
      </gml:Point>
    </ceis:Form>
  </ceis:BoFlaechen>
</ceis:member>
<ceis:member>
  <ceis:BoFlaechen gml:id="x10">
    <ceis:Art>Gebaeude</ceis:Art>
    <ceis:Form>
      <gml:Point gml:id="g10.1"
srsName="urn:ogc:def:crs:EPSG::21781" srsDimension="2">
        <gml:pos>133.95 206.06</gml:pos>
      </gml:Point>
    </ceis:Form>
  </ceis:BoFlaechen>
</ceis:member>
</ceis:Bodenbedeckung>
</ili:baskets>
</ili:TRANSFER>

```