Section II

Further GML Concepts and Application Modeling

Prof. Dr. Thomas H. Kolbe

Institute for Geodesy and Geoinformation Science
Berlin University of Technology
kolbe@igg.tu-berlin.de

May 2008

EduServ6 Course on CityGML
This is copyrighted material. It is not allowed to distribute copies or parts of these slides and the video clips without the written consent of the author.

Please note, that the presentation also contains third-party copyrighted material used with permission.
Overview

▶ Section I
  • Introduction: Urban Information Modelling
  • CityGML Overview and Status
  • OGC Geography Markup Language (GML)

▶ Section II
  • Further GML Concepts and Application Modelling

▶ Section III – CityGML Details, Part 1

▶ Section IV – CityGML Details, Part 2

▶ Section V
  • Extending CityGML
  • Application Examples

▶ Section VI
  • Relations to Other Standards
GML is a meta-format used for the specification of exchange formats for geoinformation

- The GML specification only provides the modelling framework via abstract elements and types
- Comprises a wide range of directly usable geometry and topology elements

The concrete exchange format eventually results from the definition of application-specific types and elements

- Derivation from the abstract GML3-types and elements through extension or restriction

Different applications have individual, and therefore varying application schemas
Role of the Application Schema

See also:
- ISO 19101 “Reference model“
- ISO 19109 “Rules for application schema“
The application schema defines…

…every concrete, application-specific GML type or element.

These are generally

- Geoobject types including their characteristics (Features)
  - e.g. road, river, land parcel, city, point-of-interest,…

- Multitudes of geoobjects (FeatureCollections)
  - e.g. city model, river network, cadastral extract

- Field-based spatial models (Coverages)
  - e.g. grid-based digital terrain models (DTMs)

- Measurement data (Observations)

- Libraries of terms (Dictionaries)
Main components of GML

- **Feature** represents a real-world object
  
  A feature has got certain Properties
  
  - Differentiation between spatial and non-spatial properties
  - Spatial properties are modelled by geometry and topology objects

- **Feature Collection** (a group of Features)

- **Geometry** (objects)
  
  - e.g. Points, Polygons, TINs, Solids

- **Topology** (objects)
  
  - e.g. nodes, edges, faces
Interaction of the GML components

- **Feature** is the central (abstract) class
- Modelling of non-geometric properties of Features:
  - Via attributes with standard data types e.g. String, Integer, …
  - Via associations between Feature and other classes
- **Geometry** is the (abstract) superclass of all geometry objects
- Geometric properties of Features are modelled by geometry objects using the association `geometryProperty`
- The elements of a **FeatureCollection** can be accessed via the association `featureMember`
Example for GML modeling

Real-world objects are modelled as a Subclass of the class Feature. Individual geometrical properties are defined for all geometry classes. Specific geometrical property: line-like geometry.

- Geometry
  - Feature
    - Street
      - name: string
    - City
      - Non-geometric property
  - LineString
    - curve
      - Property
        - geometry
          - Property
            - *
Structure of GML: GML objects

_AObject_

+ description [0..1] : StringOrRef
+ name [0..*] : Code
+ id [0..1] : ID

_GML_ (from gmlBase)

+ metadataProperty [0..n] : ID

_Feature_

+ fid [0..1] : CharacterString

_Definition_

+ id : ID

_Dictionary_

+ dictionaryEntry [0..n] : ID

_Metadata_

+ id [0..1] : ID

“A GML-object has an identity”
The **abstract Element** `gml:_GML` is the **root element** of every GML3 instance document (data file).

- **Standard properties:** `name`, `description`, `metadata` and an **ID-attribute**
  - modelled as child elements of `gml:_GML`
In GML3, properties are **exclusively** represented by child elements of a GML object.

- The child element defines the data type of the property.
- No usage of XML-attributes for the representation of object properties.

The property type must not be derived from `gml:AbstractGMLType`.

- No GML object is a direct child element of another GML object.
- No XML element can be GML object and GML property simultaneously.
Property values can be specified in 2 ways:

- **By value**
  - Property values are **embedded as child element** ("inline")

  ```xml
  <gml:location>
  <gml:Point gml:id="punkt0815" srsName="epsg:4326">
    <gml:coordinates>5.5623,33.2323</gml:coordinates>
  </gml:Point>
  </gml:location>
  ```

- **By reference**
  - The property element is empty and **points to another object** instead (XML-element with ID)

  ```xml
  <gml:location xlink:href="http://meine.webseite.de/locations/punkt0815" />
  ```
Features may comprise an arbitrary number of non-geometric properties

- Every property is enclosed by an individual element

Features may comprise an arbitrary number of geometric properties

- Every geometric property is enclosed by an individual element
- The element denotes the data type / the role of the geometry-object (e.g. surfaceProperty)
- The child element of the “Geometry-property-element“ is a geometry-object (e.g. point, line, polygon, …)
Example of XML-encoding

```xml
<House>
  <Number>134</Number>
  <Owner>Jupp Zupp</Owner>
  <Street>Schoenhauser Allee</Street>
  <gml:extentOf>
    <gml:Polygon>
      ...
    </gml:Polygon>
  </gml:extentOf>
</House>

<complexType name="HouseType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="Number" type="positiveInteger"/>
        <element name="Owner" type="string"/>
        <element name="Street" type="string"/>
        <element ref="gml:extentOf"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

surfaceProperty, which comprises a polygon as a child element
Feature properties

- Features may also have **properties**, that are Features themselves

- Representation of associations / aggregations
  - 1:1 relationship via referenced Feature
  - 1:n relationship via FeatureCollection with referenced Features

```xml
<element name="featureMember" type="gml:FeaturePropertyType"/>
<element name="featureProperty" type="gml:FeaturePropertyType"/>
<complexType name="FeaturePropertyType">
  <sequence>
    <element ref="gml:_Feature" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```
A FeatureCollection

- is a compilation of Features
- may comprise zero or more FeatureMembers
  - featureMember is a Property of FeatureCollection
  - featureMembers is an ArrayProperty of FeatureCollection
- is a Feature itself (FeatureCollection of FeatureCollection is possible; also recursively)
  - A FeatureCollection can have its own spatial and non-spatial properties
A concrete FeatureCollection

- Derivation from the type gml:AbstractFeatureCollectionType
- Substitute for the abstract element <gml:_Feature>
- Individual Features are included into the FeatureCollection via <featureMember>

```xml
<Citymodel gml:id="cm1456">
  <gml:featureMember>
    <House gml:id="H567">....</House>
  </gml:featureMember>

  <gml:featureMember>
    <Street gml:id="Str812">....</Street>
  </gml:featureMember>
</Citymodel>
```

- Members do not need to belong to the same class
Example of a concrete **FeatureCollection**

```xml
<element name="Citymodel" type="Ex:CitymodelType"
    substitutionGroup="gml: Feature"/>
<element name="House" type="Bsp:HouseType" substitutionGroup="gml: Feature"/>
<element name="Street" type="Bsp:StreetType" substitutionGroup="gml:_Feature"/>

<complexType name="CitymodelType">
    <complexContent>
        <extension base="gml:AbstractFeatureCollectionType">
            <sequence>...<sequence>
        </extension>
    </complexContent>
</complexType>

<complexType name="HouseType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <extension base="gml:AbstractFeatureType">
                <sequence>....</sequence>
            </extension>
        </extension>
    </complexContent>
</complexType>

<complexType name="StreetType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <extension base="gml:AbstractFeatureType">
                <sequence>.....</sequence>
            </extension>
        </extension>
    </complexContent>
</complexType>
```
Modelling of the geometry

_GML
(from gmlBase)

+ description [0..1] : CharacterString
+ name [0..*] : CharacterString
+ id [0..1] : ID

_Geometry

+ gid [0..1] : CharacterString

_CoordinateReferenceSystem
(from CoordinateReferenceSystems)

+ srsName [0..1]

_GeometricPrimitive

GeometricComplex

_GeometricAggregate
Basic concepts of the geometry model

- **Primitives**
  - Simple, continuous geometric objects

- **Geometric complexes**
  - Ensemble of geometrically non-overlapping primitives

- **Composites**
  - Special type of complexes: homogenous composition of primitives and composites of the same dimension
  - Isomorphic with respect to primitives of the same dimension
  - Coherent
  - Individual primitives have common geometry subsets (primitives of low dimension, e.g. the borderline of two adjacent surfaces)

- **Aggregates**
  - Collections of individual geometry elements, that do not need to be connected; overlaps allowed!
Hierarchy of the geometry types

- _Geometry
  - +gid [0..1]: String

- _CoordinateReferenceSystem
  - srsName 0..1

- _GeometricPrimitive

- GeometricComplex

- _GeometricAggregate

- Point
  - 0D

- _Curve
  - 1D

- _Surface
  - 2D

- _Solid
  - 3D
1. Coordinates element: list of coordinates

The syntactic rule for the separation of the decimal places, the x and y values and the coordinate pairs is defined by the attributes.

```xml
<Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
  <coordinates> 49.11,24.12 </coordinates>
</Point>
```

Separation of decimal places (.)

coordinate separator (,)

tuple separator ( )
2. **Pos element**: list of ordinates (n-dimensional coordinate)

```xml
<Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
  <pos dimension="2">5.0 40.0</pos>
</Point>
```

```xml
<element name="pos" type="gml:DirectPositionType" />
<complexType name="DirectPositionType">
  <simpleContent>
    <extension base="gml:doubleList">
      <attribute name="srsName" type="anyURI" use="optional"/>
      <attribute name="dimension" type="positiveInteger" use="optional"/>
    </extension>
  </simplecontent>
</complexType>
```
A Point element consists of a coordinate tuple.

Example:

```xml
<Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
  <pos>56.1 13.1</pos>
</Point>
```
LineString: A sequence of points, connected by straight line segments

```xml
<element name="LineString" type="gml:LineStringType"
    substitutionGroup="gml:_Curve"/>
<complexType name="LineStringType">
    <complexContent>
        <extension base="gml:AbstractCurveType">
            <sequence>
                <choice>
                    <choice minOccurs="2" maxOccurs="unbounded">
                        <element ref="gml:pos" />
                        <element ref="gml:coord" />
                        <element ref="gml:pointRep" />
                    </choice>
                    <element ref="gml:coordinates"/>
                </choice>
            </sequence>
        </extension>
    </complexContent>
</complexType>

<LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
    <coordinates>100.0,100.0 230.0,80.0 350.0,130.0</coordinates>
</LineString>
```
The **envelope element** serves for the modelling of a spatial extent. It consists of two coordinate tuples, that describe the diagonally opposite corners.

**Typical application: Specification of a bounding box**

Example:

```xml
<Envelope srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
  <coordinates> 0.0,0.0 30.0,100.0 </coordinates>
</Envelope>
```
Polygon geometry element

_Underline Surface_

_Underline Polygon_

  +exterior 0..1

  +interior 0..n

_Underline Ring_

_Underline LinearRing_

+position [4..n]: Position

exterior

interior
Polygon geometry element - example

```xml
<Polygon srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
  <exterior>
    <LinearRing gml:id="Außen1">
      <coordinates>
        0.0,0.0 100.0,0.0 100.0,100.0 0.0,100.0 0.0,0.0
      </coordinates>
    </LinearRing>
  </exterior>
  <interior>
    <LinearRing gml:id="Innen1">
      <coordinates>
        60.0,60.0 60.0,90.0 90.0,90.0 90.0,60.0 60.0,60.0
      </coordinates>
    </LinearRing>
  </interior>
</Polygon>
```
Solid geometries

_GeometricPrimitive

+solidMember

_Solid

CompositeSolid

Solid

_Surface

+exterior

+interior

out-side
Solid - example

```xml
<gml:Solid srsName="...some reference system...">
  <gml:exterior>
    <gml:CompositeSurface>
      <gml:surfaceMember>
        <gml:OrientableSurface orientation="+">
          <gml:baseSurface>
            <gml:Polygon>
              <gml:exterior>
                <gml:LinearRing>
                  <gml:pos dimension="3">1.0 1.0 0.0</gml:pos>
                  <gml:pos dimension="3">3.0 1.0 0.0</gml:pos>
                  <gml:pos dimension="3">3.0 1.0 1.5</gml:pos>
                  <gml:pos dimension="3">2.0 1.0 2.5</gml:pos>
                  <gml:pos dimension="3">1.0 1.0 2.5</gml:pos>
                  <gml:pos dimension="3">1.0 1.0 1.5</gml:pos>
                  <gml:pos dimension="3">1.0 1.0 1.5</gml:pos>
                  <gml:pos dimension="3">1.0 1.0 0.0</gml:pos>
                </gml:LinearRing>
              </gml:exterior>
            </gml:Polygon>
          </gml:baseSurface>
        </gml:OrientableSurface>
      </gml:surfaceMember>
      ... <!-- other surfaces --> ...
    </gml:CompositeSurface>
  </gml:exterior>
</gml:Solid>
```
In general, geometry objects are positively directed

- Curves (each segment) from starting point to end point
- For surfaces, the normal vector determines the orientation:
  positive direction = upper side; Right-hand-rule

Explicit statement of the direction of a geometry object allows to re-use Primitives in Complexes

**Example:**

Curves made up of individual segments:

- Curve 1: +S1 +S2 +S3
- Curve 2: +S4 $-S_2$ +S5
Geometric aggregates

- MultiLineString
  - 0..n +lineStringMember
    - LineString
- MultiPoint
  - i +pointMember
    - 0..1 +pointMember
    - 0..n +pointMembers
    - MultiGeometry
      - i +geometryMember
        - 0..1 +geometryMember
        - 0..n +geometryMembers
  - MultiCurve
    - i +curveMember
      - 0..1 +curveMember
      - 0..n +curveMembers
  - MultiSurface
    - i +surfaceMember
      - 0..1 +surfaceMember
      - 0..n +surfaceMembers
  - MultiSolid
    - i +solidMember
      - 0..1 +solidMember
      - 0..n +solidMembers

+polygonMember
  - 0..n

_heterogeneous aggregate_
Summary

- **GML3** is currently the most comprehensive standard for the representation of geodata
  - 0D-, 1D-, 2D- and 3D-geometries; topology, time
  - Coverages; observations; relations between geoobjects
- GML3 specifies a **meta-format**
- GML3 format A ≠ GML3 format B
  - **Compatible** only if application schemas are identical
  - NAS is the common application schema for ALKIS
- Downside:
  - High **complexity**, especially concerning familiarization
  - **Files** become **very big**, due to XML overhead
Example: A simple 2D city model

District Euskirchen
Area 14
Land parcel 5
Owner Leo Land

District Euskirchen
Area 14
Land parcel 7
Owner
City of Euskirchen

Meckenheimer Allee
UML diagram of the 2D city model

City model

Street
  strname: string

_surfaceProperty
  1
  1

Polyon

LineString

_GeometryProperty

_Feature
  geometry Property

_featureMember
  1

_featureMember
  1

_FeatureCollection

_Geometry
2D city model application schema (1)

Header of the schema file

1. **Schema-element** including the namespace of the schema
2. **Declaration of all referenced namespaces**
   (here: XML-schema, XLink, GML and the namespace of the application schema)
3. **Import of required schema definitions**
   (here: base schema feature .xsd of GML3; loads others)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.net/example"
        xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:gml="http://www.opengis.net/gml"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:bsp="http://www.beispiel.net/beispiel">
    <import namespace="http://www.opengis.net/gml"
            schemaLocation="feature.xsd"/>
    ...
</schema>
```
Declaration of the root element

- First `<element>` tag in the XML-schema-file defines the root element of the GML instance documents (data files)

- Declaration of the element `Citymodel`; this may be used, where GML expects a FeatureCollection

- Definition of the type `CityModelType` as a subclass of the abstract GML class `AbstractFeatureCollectionType`

```xml
...  
<element name="Citymodel" type="bsp:CitymodelType"
   substitutionGroup="gml:_FeatureCollection"/>

<complexType name="CitymodelType">
   <complexContent>
      <extension base="gml:AbstractFeatureCollectionType"/>
   </complexContent>
</complexType>
...```
Representation of Features (1)

- **Declaration of the element** *Landparcel*; this may be used, where GML expects a Feature

- **Definition of the type** *LandparcelType* as a subclass of the abstract GML class *AbstractFeatureType*

```xml
<element name="Landparcel" type="bsp:LandparcelType"
    substitutionGroup="gml:_Feature"/>
<complexType name="LandparcelType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="District" type="string"/>
                <element name="Area" type="integer"/>
                <element name="Owner" type="string"/>
                <element ref="gml:surfaceProperty"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

Attributes are realized via child elements with simple data types

Representation of the extent via pre-defined GML-geometry-property
Representation of Features (2)

- **Declaration of the element Street**: this may be used, where GML expects a Feature
- **Definition of the type StreetType** as a subclass of the abstract GML class AbstractFeatureType

```xml
<element name="Street" type="bsp:StreetType"
    substitutionGroup="gml:_Feature"/>
<complexType name="StreetType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="strname" type="string"/>
                <element ref="gml:curveProperty"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

**Representation of the geometry of the street via pre-defined GML-geometry-property**
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.net/example"
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:bsp="http://www.example.net/example">
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <element name="Citymodel" type="bsp:CitymodelType" substitutionGroup="gml:_FeatureCollection"/>
    <complexType name="CitymodelType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <element name="Landparcel" type="bsp:LandparcelType" substitutionGroup="gml:_Feature"/>
    <complexType name="LandparcelType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="District" type="string"/>
                    <element name="Area" type="integer"/>
                    <element name="Owner" type="string"/>
                    <element ref="gml:surfaceProperty"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="Street" type="bsp:StreetType" substitutionGroup="gml:_Feature"/>
    <complexType name="StreetType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="strname" type="string"/>
                    <element ref="gml:curveProperty"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
GML instance document – example

District Euskirchen
Area 14
Land parcel 5
Owner Leo Land

Meckenheimer Allee
<?xml version="1.0" encoding="ISO-8859-1"?>
<Citymodel xmlns="http://www.example.net/example"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:xlink="http://www.w3.org/1999/xlink">
    <gml:name>Cadastre of the City of XY</gml:name>
    <gml:boundedBy>
        ...
    </gml:boundedBy>
    <gml:featureMember>
        ...
    </gml:featureMember>
</Citymodel>
**BoundedBy:**
The *Envelope* defined in *boundedBy* encloses all geodata of this file.

```xml
<gml:boundedBy>
  <gml:Envelope
       srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
    <gml:coordinates> 9500.0,4300.0 9650.7,4353.6</gml:coordinates>
  </gml:Envelope>
</gml:boundedBy>
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Citymodel xmlns="http://www.example.net/example"
           xmlns:gml="http://www.opengis.net/gml"
           xmlns:xlink="http://www.w3.org/1999/xlink">
    <gml:name>Cadastre of the City of XY</gml:name>
    <gml:boundedBy><gml:Envelope
                  srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        <gml:coordinates> 9500.0,4300.0 9650.7,4353.6</gml:coordinates>
    </gml:Envelope>
    </gml:boundedBy>
    <gml:featureMember><Landparcel> ... </Landparcel></gml:featureMember>
    <gml:featureMember><Street> ... </Street></gml:featureMember>
</Citymodel>
<gml:featureMember>
  <Landparcel>
    <gml:name>Flst. 5</gml:name>
    <District>Euskirchen</District>
    <Area>14</Area>
    <Owner>Leo Land</Owner>

    <gml:surfaceProperty>
      <gml:Polygon srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        ...
      </gml:Polygon>
    </gml:surfaceProperty>
  </Landparcel>
</gml:featureMember>
<gml:surfaceProperty>
  <gml:Polygon srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
    <gml:exterior>
      <gml:Ring>
        <gml:curveMember>
          <gml:LineString>
            <gml:coordinates>
              9500.0,4300.0  9566.8,4306.2  9572.2,4325.5
              9568.8,4341.0  9513.7,4343.6  9500.0,4300.0
            </gml:coordinates>
          </gml:LineString>
        </gml:curveMember>
      </gml:Ring>
    </gml:exterior>
  </gml:Polygon>
</gml:surfaceProperty>
<?xml version="1.0" encoding="ISO-8859-1"?>
<Citymodel xmlns="http://www.lecture.net/example"
    xmlns:gml="http://www.opengis.net/gml">
    <gml:name>Cadastre</gml:name>
    <gml:boundedBy>
        <gml:Box srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
            <gml:coordinates>9500.0,4300.0 9650.7,4353.6</gml:coordinates>
        </gml:Box>
    </gml:boundedBy>
    <gml:featureMember>
        <Landparcel>...</Landparcel>
    </gml:featureMember>
    <gml:featureMember>
        <Street>...</Street>
    </gml:featureMember>
</Citymodel>
<Street>
  <strname>Meckenheimer Allee</strname>
  <gml:curveProperty>
    <gml:LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
      <gml:coordinates>
        9510.0,4333.0  9536.4,4320.1  9555.5,4310.7
      </gml:coordinates>
    </gml:LineString>
  </gml:curveProperty>
</Street>
<?xml version="1.0" encoding="ISO-8859-1"?>
<Citymodel xmlns="http://www.example.net/example" xmlns:gml="http://www.opengis.net/gml"
  <gml:name>Cadastre of City XY</gml:name>
  <gml:boundedBy>
    <gml:Envelope srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
      <gml:coordinates>9500.0,4300.0 9650.7,4353.6</gml:coordinates>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <Landparcel>
      <gml:name>Flst. 5</gml:name>
      <District>Euskirchen</District>
      <Area>14</Area>
      <Owner>Leo Land</Owner>
      <gml:surfaceProperty>
        <gml:Polygon srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
          <gml:exterior>
            <gml:Ring>
              <gml:curveMember>
                <gml:LineString>
                  <gml:coordinates>9500.0,4300.0 9566.8,4306.2 9572.2,4325.5 9568.8,4341.0 9513.7,4343.6 9500.0,4300.0</gml:coordinates>
                </gml:LineString>
              </gml:curveMember>
            </gml:Ring>
          </gml:exterior>
        </gml:Polygon>
      </gml:surfaceProperty>
    </Landparcel>
    <gml:featureMember>
      <Street>
        <strname>Meckenheimer Allee</strname>
        <gml:curveProperty>
          <gml:LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
            <gml:coordinates>9510.0,4333.0 9536.4,4320.1 9555.5,4310.7</gml:coordinates>
          </gml:LineString>
        </gml:curveProperty>
      </Street>
    </gml:featureMember>
  </gml:featureMember>
</Citymodel>