CityGML 2.0 – Ein internationaler Standard für 3D-Stadtmodelle Teil 1: Datenmodell

Marc-O. Löwner, Joachim Benner, Gerhard Gröger, Ulrich Gruber, Karl-Heinz Häfele und Sandra Schlüter

Zusammenfassung

Der vorliegende Beitrag widmet sich den Neuerungen des im März 2012 vom Open Geospatial Consortium verabschiedeten Standards CityGML 2.0 zur Repräsentation virtueller 3D-Stadtmodelle gegenüber seiner Vorgängerversion. Nach einem kurzen Abriss der Entwicklungsgeschichte von CityGML als geometrisch-semantisches Datenmodell wird detailliert auf die jüngsten Erweiterungen der Version 2.0 eingegangen. Insbesondere werden die neu eingeführten Module für Tunnel und Brücken beschrieben. Darüber hinaus werden u. a. die ergänzend eingeführten Attribute *relativeToTerrain* und *relativeToWater* der *_CityObject*-Klasse, die Erweiterung der Verwendung impliziter Geometrien, die Einführung eines neuen Levels of Detail 0 für Gebäude sowie die Einführung der neuen thematischen Begrenzungsflächen *OuterFloorSurface* und *OuterCeilingSurface* dargestellt. Die Mechanismen zur Einschränkung oder Verschärfung eines GML-basierten Standards, aber auch die Erweiterung durch generische Objekte und Attribute sowie das Erweiterungskonzept *Application Domain Extension* (ADE) werden im Anschluss ausführlich beleuchtet. CityGML 2.0 wurde gegenüber der Vorgängerversion um zwei neue Module, 41 Featureklassen und 203 neue Attribute erweitert. Trotz dieser Änderungen ist der Migrationsaufwand für Instanzdokumente der Version 1.0 minimal.

Summary

This paper is devoted to the new version 2.0 of CityGML released by the Open Geospatial Consortium in March 2012. CityGML is a common information model for the representation, storage and exchange of virtual 3D city models. After a brief outline of the history of development the most recent enhancements of version 2.0 are explicated. In particular, these are the newly introduced modules for tunnels and bridges, the attributes relativeToTerrain *and* relativeToWater *added to the* _CityObject *class, the enhanced applicability of implicit geometries, the newly introduced Level of Detail 0 for buildings and the additional thematic boundary surfaces* OuterFloor Surface *and* OuterCeilingSurface. *Furthermore, adjustment mechanisms to confine or to expand CityGML by generic objects and generic attributes are discussed as well as the* Application Domain Extensions *(ADEs).*

CityGML 2.0 has been improved through the introduction of two new modules, 41 feature classes and 203 attributes. However, migration costs for CityGML's version 1.0 instance documents are extremely low.

Schlüsselwörter: CityGML 2.0, 3D-Stadtmodelle, Open Geospatial Consortium

1 Einleitung

Im März 2012 ist CityGML 2.0 vom Open Geospatial Consortium (OGC) als neuer internationaler Standard zur Repräsentation und für den Austausch virtueller 3D-Stadtmodelle herausgegeben worden. CityGML 2.0 weist signifikante Verbesserungen gegenüber der Vorgängerversion auf. Die wesentlichen Veränderungen gegenüber der Version 1.0 bestehen in der Einführung neuer sowie der Ergänzung schon bestehender thematischer Module. Insgesamt wurde CityGML 2.0 gegenüber der Vorgängerversion um zwei neue Module für die Repräsentation von Brücken und Tunneln, 41 Feature-Klassen und 203 neue Attribute erweitert.

Virtuelle 3D-Stadtmodelle sind heute wichtige Werkzeuge z. B. in den Bereichen der Stadtplanung, des Stadtmarketings, des Katastrophenmanagements, der Funknetzplanung und der Analyse von Solarpotenzialen auf Dachflächen. Sie werden in Deutschland von Städten, Kommunen und den Bundesländern unter Einbeziehung etablierter Dienstleister aufgebaut. Über die Visualisierung hinausgehende Analysen von 3D-Stadtmodellen sind aber nur möglich, wenn neben der äußeren Form der Objekte auch deren Eigenschaften und deren Beziehungen untereinander in einer Datenstruktur abgebildet werden können. Die City Geography Markup Language (CityGML) ist ein offenes und applikationsunabhängiges Datenmodell zur Repräsentation, Speicherung und zum Austausch von virtuellen 3D-Stadt- und Regionalmodellen. Neben der geometrischen Repräsentation der 3D-Elemente ermöglicht CityGML auch die Speicherung der semantischen Eigenschaften der Objekte sowie ihrer Relationen zueinander. Dies beinhaltet auch die Generalisierungen und Aggregationen der thematischen Klassen. Damit geht CityGML weit über Standards zur reinen Visualisierung von 3D-Stadtmodellen, wie etwa die Keyhole Markup Language (KML) für Google Earth, Collada oder X3D hinaus. Es erlaubt dem Anwender vielmehr, 3D-Stadt- und Regionalmodelle für anspruchsvolle Analyseschritte, etwa für Umweltsimulation, Energiefragen, urbanes Anlagenmanagement, Grundstücks- und Immobilienbewertung, Katastrophenmanagement, ortsbezogene Marktstrategien und andere Fachanwendungen auszutauschen. Darüber hinaus ist CityGML durch ihre Modularisierung und ihre verschiedenen Detailierungsstufen, den Level of Detail (LoD), stark skalierbar. Neben einer hochwertigen Repräsentation einzelner Gebäude können auch ganze Städte, Regionen oder Länder modelliert werden.

Damit unterscheidet sich CityGML von den von building SMART entwickelten Industry Foundation Classes (IFC), einem Standard zur Beschreibung von Gebäudemodellen. Die Repräsentation der Semantik in CityGML ist so detailliert, dass die Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV) in Zusammenarbeit mit der AG Modellierung der SIG 3D-Profile von CityGML definiert hat. Mit diesen können die von den Kommunen vorgehaltenen 3D-Stadtmodelle bis zur flächendeckenden Einführung des Amtlichen Liegenschaftskatasterinformationssystems (ALKIS®) in der Version GeoInfoDok 7.0 übergangsweise geführt werden (Gruber 2012). Darüber hinaus bildet CityGML auch in den Geodateninfrastrukturen der Niederlande, Frankreichs, Malaysias, Abu Dhabis und anderer Länder eine wichtige Plattform im Übergang von einer 2D- zu einer 3D-Geodatenbasis (citygml.org).

Das OGC, das CityGML als Implementierungsstandard verabschiedet hat, ist neben der International Organization for Standardization (ISO) die wichtigste Organisation zur Standardisierung im Bereich der Geoinformationstechnologie. Hauptziel des OGC ist es, frei zugängliche Standards im Bereich der Geoinformationstechnologie zur Verfügung zu stellen. Ein wesentlicher Aspekt ist dabei die Interoperabilität, um GI-Technologien effektiv anwendbar zu machen. Zu den wichtigsten der insgesamt 35 Standards zählen die Geography Markup Language (GML), der Web Feature Service (WFS) und die Keyhole Markup Language (KML), ein wesentliches Datenmodell für GoogleEarth. Als Implementierungsstandards haben diese Spezifikationen und damit auch CityGML die Möglichkeit, zum ISO-Standard erhoben zu werden.

CityGML ist als Anwendungsschema der erweiterbaren Geography Markup Language (GML 3.1.1) (Cox et al. 2004) realisiert, die wiederum syntaktisch auf der Extensible Markup Language (XML) fußt. Dadurch kann auch CityGML von der ganzen Bandbreite der GML-kompatiblen OGC-Internetservices zum Datenzugriff, zur Prozessierung und Katalogisierung genutzt werden. Dies gilt insbesondere für den Web Feature Service (WFS), den Web Processing Service (WPS) und den OGC Catalogue Service.

CityGML wird seit dem Jahr 2002 von der Special Interest Group 3D (SIG 3D) entwickelt. Die SIG 3D, ursprünglich ein Teil der Initiative Geodateninfrastruktur NRW, ist seit 2010 ein Arbeitskreis der Initiative Geodateninfrastruktur Deutschland (GDI-DE). Erste Modellkonzepte des Basis- und Gebäudemodells wurden von Gröger et al. (2005) in der ZfV veröffentlicht. Im Zuge der Zusammenarbeit mit der European Spatial Data Research (EuroSDR) wurde CityGML in der Version 0.3.0 im Juni 2006 als Candidate OpenGIS Implementation Specification in das OGC eingebracht und im Juli 2007 in der Version 0.4.0 (Gröger et al. 2007) als Best Practice Paper akzeptiert. Schon in dieser Version weist CityGML das Konzept der Application Domain Extensions (ADE) (engl.: Fachschalenerweiterung) auf. Hiermit ist es möglich, auf CityGML basierende, erweiterte Modelle zu spezifizieren (s. Kap. 3.4). Am 20. August 2008 wurde CityGML in der Version 1.0.0 (Gröger et al. 2008) von den Mitgliedern des OGC als offizieller Standard angenommen. Die entscheidende Veränderung gegenüber der Version 0.4.0 war eine vertikale Modularisierung durch die Einführung von zwölf XML-Namensräumen. In der Folge müssen Applikationen nicht mehr den gesamten Standard unterstützen, sondern nur noch einzelne Module.

CityGML 2.0 wurde schließlich am 14. März 2012 nach 14-monatiger Bearbeitung von Änderungsanfragen durch die SIG 3D vom OGC als neuer Standard veröffentlicht (Gröger et al. 2012). Im Einzelnen umfassen die wichtigsten Änderungen und damit auch die Schwerpunkte dieses Beitrages die folgenden Punkte:

- Ergänzung der *_CityObject*-Klasse durch optionale Attribute *relativeToTerrain* und *relativeToWater* (Kap. 2.1),
- Erweiterung der Verwendung der impliziten Geometrien (Kap. 2.2),
- Einführung einer LoD0-Repräsentation für Gebäude (Kap. 2.2),
- Einführung der neuen thematischen Begrenzungsflächen *OuterFloorSurface* und *OuterCeilingSurface* (Kap. 2.2),
- Möglichkeit, Gebäudeinstallationen mit thematischen Begrenzungsflächen zu assoziieren (Kap. 2.2),
- Einführung der neuen Module *Bridge (brid)* und *Tunnel (tun)* (Kap. 2.3 und Kap. 2.4),
- Erweiterung des *Generic*-Moduls um die Attribute *measureAttribute* und *genericAttributeSet* (Kap. 3.3), sowie die
- Erweiterung der Möglichkeit, Codelisten in einem Instanzdokument explizit zu referenzieren (Kap. 3.2).

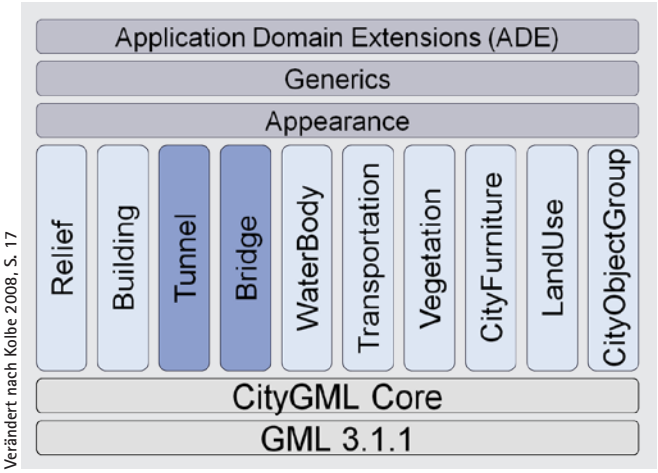
Trotz dieser Erweiterungen ist der Migrationsaufwand für existierende Modelle minimal. Instanzdokumente der Version 1.0 sind bei entsprechenden Namensraumangaben und einer angepassten Verlinkung auf das Schema der Version 2.0 valide Dokumente dieser Version. Der Sprung zu einer neuen Hauptversionsnummer wurde nur durch strikte Vorgaben des OGC (vgl. Reed 2011), z. B. bzgl. der Erweiterung bestehender Klassen durch optionale Attribute, notwendig.

Dies ist der erste von zwei Beiträgen über die Neuerungen von CityGML der Version 2.0, der sich mit den Änderungen im Datenmodell befasst. Der zweite Beitrag wird den Einfluss von CityGML auf ALKIS® und das Gebäudemodell von INSPIRE (INSPIRE Data Specification on *Building* Version 2.9), die Qualitätssicherung von CityGML-Modellen sowie CityGML-Anwendungen beleuchten. In den folgenden Kapiteln wird auf die wichtigsten Module von CityGML 2.0 mit der Fokussierung auf die Neuerungen gegenüber der Version 1.0 eingegangen. Dabei erfolgt die Angabe der Klassen- und Attributnamen aus den originalen UML-Diagrammen des Standards in englischer Sprache und *kursiver* Schrift. Teilweise werden so auch Instanzen der jeweiligen Klassen kenntlich

gemacht. Ein paralleles Lesen des Standards (in englischer Sprache), der unter www.opengeospatial.org/standards/citygml herunter geladen werden kann, wird empfohlen.

2 Das thematische Modell von CityGML 2.0

CityGML ist ein objektorientiertes, semantisches Datenmodell für virtuelle, dreidimensionale Stadtmodelle. Es repräsentiert die Elemente einer Stadtlandschaft, wie



Verändert nach Kolbe 2008, S. 17

Abb. 1: Übersicht über die Module von CityGML 2.0

Gebäude, Infrastruktureinrichtungen oder Gewässer als Objekte, denen zusätzlich zur geometrischen Form auch nichtgeometrische Eigenschaften, die Semantik der Objekte, zugeordnet werden können. Darüber hinaus können diese Objekte über Relationen mit anderen Objekten in Beziehung stehen. Allen Flächen können Oberflächeneigenschaften zugeordnet werden, die nicht auf das optische Erscheinungsbild beschränkt sind, sondern auch verschiedenen Themen, wie etwa Messwerten zugeordnet werden können.

CityGML 2.0 ist wie die Vorgängerversion als GML 3.1.1 Anwendungsschema realisiert, welches die Geometrierepräsentationen und einige grundlegende Modellierungskonzepte bereitstellt. CityGML ist modular aufgebaut, indem unterschiedliche fachliche Themen durch jeweils eigene XML-Schemata mit separaten Namensräumen repräsentiert sind. Durch diese Modularisierung müssen Applikationen nicht das gesamte CityGML-Schema implementieren, um CityGML-konform zu sein. In der Version 2.0 enthält CityGML folgende Module mit den in Klammern angegebenen Namensräumen (s. Abb. 1):

- ***CityGML Core (core):*** Repräsentation abstrakter Basisklassen sowie modulübergreifend benutzter Klassen und Datentypen, erweitert in CityGML 2.0 (Kap. 2.1).
- ***Relief (dem):*** Repräsentation der geometrischen Struktur der Erdoberfläche.
- ***Building (bldg):*** Repräsentation von Gebäuden und Gebäudekomponenten, erweitert in CityGML 2.0 (Kap. 2.2).
- ***Bridge (brid):*** Repräsentation von Brücken und deren Komponenten, neu in CityGML 2.0 (Kap. 2.3).
- ***Tunnel (tun):*** Repräsentation von Tunneln und deren Komponenten, neu in CityGML 2.0 (Kap. 2.4).
- ***WaterBody (wtr):*** Repräsentation von Wasserflächen bzw. -körpern.
- ***Transportation (tran):*** Repräsentation vom Verkehrsraum, wie Straßen oder Schienenwegen.
- ***Vegetation (veg):*** Repräsentation einzelner oder flächenhaft verteilter Vegetationsobjekte.
- ***CityFurniture (frn):*** Repräsentation weiterer Objekte einer Stadtlandschaft, wie Verkehrsampeln oder Reklametafeln, die nicht durch ein anderes thematisches Modul abgebildet werden.
- ***LandUse (luse):*** Repräsentation der physikalischen oder biologischen Landbedeckung oder der sozio-ökonomischen Landnutzung.
- ***CityObjectGroup (grp):*** Repräsentation von applikationsspezifischen Gruppierungen beliebiger CityGML-Objekte.

Darüber hinaus gibt es noch das Modul *Appearance (app)* für die Abbildung objektspezifischer Farben und Texturen, sowie Module und Mechanismen zur Erweiterung des Standards (*Generics*, Application Domain Extensions, s. Kap. 3.4).

Ein wichtiges Konzept zur Unterstützung der Skalierbarkeit von CityGML ist die Möglichkeit, alle Stadtobjekte mit unterschiedlichem Detaillierungsgrad zu modellieren. Das Konzept des Levels of Detail (LoD) ermöglicht sowohl eine abgestufte Verfeinerung der geometrischen Ausprägung von Objekten als auch die Hinzunahme von weiteren semantischen Eigenschaften. Das LoD-Konzept in CityGML wurde primär für das Gebäudemodul entwickelt und sinngemäß auf andere Fachthemen (CityGML-Module) übertragen. In CityGML 2.0 wurde die geometrische Repräsentation für Gebäude um den LoD0 erweitert. Insgesamt gibt es im Gebäudemodul fünf LoDs, die in Abb. 2 dargestellt sind.

- LoD0: Modellierung des Gebäudegrundrisses sowie des Dachumrisses durch horizontale Flächen (mit konstanten Z-Werten) ohne semantische Strukturierung (s. Kap. 2.2).
- LoD1: Modellierung der Gebäudegeometrie durch das sog. Klötzchenmodell, einen vertikalen Extrusionskörper ohne semantische Strukturierung. Die geometrische Repräsentation erfolgt als *gml:Solid* oder als *gml:MultiSurface*.
- LoD2: Geometrisch generalisierte Abbildung der Gebäude-Außenkontur, semantische Strukturierung der Außenflächen (Begrenzungsflächen) durch die Klassen *GroundSurface*, *RoofSurface*, *WallSurface* und durch die neu eingeführten Klassen *OuterCeilingSurface* und *OuterFloorSurface* (s. u.) sowie die separate Modellierung wichtiger Gebäudemerkmale (Gebäudeinstallationen) durch die Klasse *BuildingInstallation*.

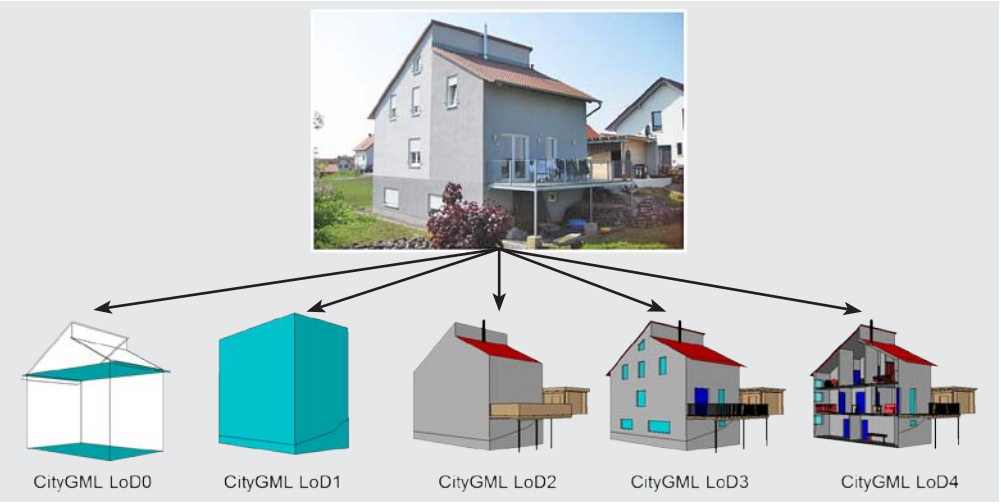
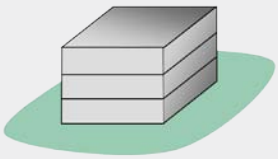
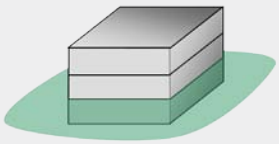
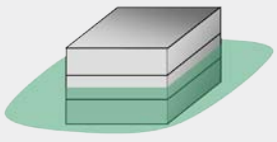
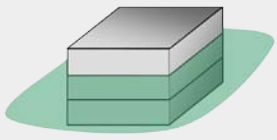
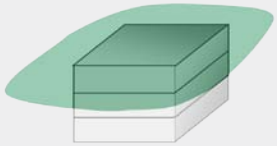


Abb. 2: Repräsentation eines Gebäudes in den LoDs 0 bis 4 (mit Möbeln)

- LoD3: Geometrisch exakte Abbildung der Gebäude-Außenkontur. Bei der semantischen Strukturierung werden Türen und Fenster als Spezialisierungen von *_Opening* mit den Klassen *Door* und *Window* explizit modelliert.
- LoD4: Zusätzliche Modellierung des Gebäudeinneren (Räume mit Begrenzungsflächen, Möbel und interne Gebäudeinstallationen). Die semantische Differenzierung des Innenraumes erfolgt durch die Klassen *InteriorWallSurface*, *FloorSurface* und *CeilingSurface*.

Tab. 1: Die möglichen Werte des optionalen Attributs *relativeToTerrain* der Klasse *_CityObject* und ihre Erläuterungen

	*entirelyAboveTerrain* Das *_CityObject* liegt vollständig oberhalb des Geländes, z. B. ein kellerloses Haus.
	*substantiallyAboveTerrain* Das *_CityObject* liegt zum größten Teil oberhalb des Geländes, z. B. ein unterkellertes, mehrstöckiges Haus.
	*substantiallyAboveAndBelowTerrain* Das *_CityObject* liegt gleichermaßen oberhalb und unterhalb des Geländes, z. B. ein unterkellertes, einstöckiges Haus.
	*substantiallyBelowTerrain* Das *_CityObject* liegt zum größten Teil unterhalb des Geländes, z. B. ein Tunnel mit oberirdischen Bauteilen.
	*entirelyBelowTerrain* Das *_CityObject* liegt vollständig unterhalb des Geländes, z. B. ein *TunnelPart* (s. Kap. 2.4).

2.1 Das Basismodul *CityGML Core*

Den Grundstein des semantischen Modells in CityGML bildet seit der Version 0.3.0 das Basismodul, das seit der Version 1.0 im Namensraum *core* angesiedelt ist. Da alle weiteren Module auf diesem Modul aufbauen, muss es in jedem validen System implementiert sein. Das Basismodul importiert die Schema Definition von GML 3.1.1 sowie die von der Organization for the Advancement of Structured Information Standards (OASIS) definierte, erweiterbare Adressenbeschreibungssprache xAL.

Die Hauptklasse bildet das von *gml:_Feature* abgeleitete *_CityObject*, aus welchem sich das von *gml:FeatureCollection* abgeleitete *CityModel* aggregiert. *_CityObject* stellt die Basis für alle thematischen Klassen in CityGML dar. Sie beinhaltet die Attribute *creationDate* (Erstellungsdatum) und *terminationDate* (Löschdatum) zur Unterstützung einer Historienverwaltung von 3D-Stadtmodellen. Darüber hinaus bietet die Klasse *_CityObject* die mögliche Assoziation zu einer externen Referenz auf dasselbe Objekt in einem anderen Datensatz, etwa ALKIS®.

In CityGML 2.0 ist es im Vergleich zu älteren Versionen möglich, die relative Lage eines *_CityObjects*, etwa eines Gebäudes oder anderer Realweltobjekte, in Bezug auf die Erdoberfläche oder Wasserfläche explizit zu speichern. Dies birgt den Vorteil, eine Aussage über die Lage des *_CityObjects* ohne die Analyse eines digitalen Höhenmodells abzufragen. Hierfür wurde die Klasse *_CityObject* durch die optionalen Attribute *relativeToTerrain* (wtl.: bezüglich des Geländes) und *relativeToWater* (wtl.: bezüglich des Wassers) erweitert. Die zugelassenen Werte der Attribute sind für *relativeToTerrain* in Tab. 1 erläutert.

Analog dazu wird die Wertemenge für das Attribut *relativeToWater* definiert. Allerdings ist diese Liste durch den Wert *temporarilyAboveAndBelowWaterSurface* ergänzt, was einem möglicherweise durch Tide oder Niederschlag schwankendem See- bzw. Wasserspiegel geschuldet ist.

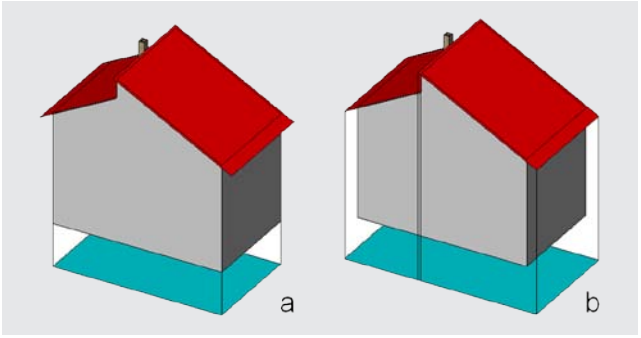
Das Basismodul ermöglicht die Repräsentation von Adressen. Dabei ist die Klasse *Address* als GML Feature realisiert, das über die Eigenschaft *xalAddress* mit der von OASIS definierten, erweiterbaren Adressenbeschreibungssprache xAL (OASIS 2003) assoziiert ist. Zudem hat sie eine optionale Assoziation zu einem *gml:MultiPoint*, um die Adresse zu verorten. Die Modellierung der Klasse *Address* als Feature birgt den Vorteil, über die XLink Technologie von GML 3.1.1 externe Ressourcen, etwa einen Web Feature Service einzubinden, der die Adressen als *FeatureCollection* vorhält. Seit der Version 1.0 erfolgt dies z. B. im Gebäudemodul durch eine Assoziation von *Address* mit *_AbstractBuilding* und damit auch *Building* und *BuildingPart* sowie mit der Klasse *Door*.

Bereits in ihrer ersten Veröffentlichung in der Version 0.3.0 gibt es in CityGML das Konzept der impliziten Geometrie, die ab Version 1.0 unverändert im Basismodul durch die Klasse *ImplicitGeometry* repräsentiert wird. Eine implizite Geometrie ist eine geometrische Repräsentation mehrerer Objekte, die prototypisch nur einmal gespeichert wird. Dabei kann im Attribut *libraryObject* auch eine externe Quelle, etwa eine VRML-, DXF- oder 3D Studio MAX-Datei oder aber ein entsprechender Web Service referenziert werden. Alternativ kann auch eine GML 3.1.1 Geometrie verwendet werden. Diese prototypische Geometrie kann beliebig häufig wiederverwendet werden, etwa bei der Darstellung von Bäumen oder Straßenschildern. Neu in CityGML 2.0 ist die Verwendung der impliziten Geometrie bei der Repräsentation von inneren und äußeren Gebäudeinstallationen (s. Kap. 2.2). Die Verortung der Geometrie ist durch eine Assoziation der *ImplicitGeometry* mit einem *gml:Point* repräsentiert. Da die implizite Geometrie in der Regel in einem lokalen, kartesischen Koordinatensystem vorliegt, enthält sie als Attribut eine 4 × 4-Transformationsmatrix für Rotation, Skalierung und lokale Translation. Die Einführung des Konzeptes der impliziten Geometrie orientiert sich an der Instanziierung von Primitiven zur Repräsentation von Szenengraphen in der Computergraphik (vgl. Foley et al. 1995). Dadurch wird neben einem verminderten Speicherbedarf auch eine schnellere Visualisierung ermöglicht.

2.2 Das Gebäudemodul *Building*

Das wichtigste thematische Modul in CityGML ist das Gebäudemodul (Namensraum *bldg*) mit der zentralen, abstrakten Klasse *_AbstractBuilding*. Seit der Version 0.4.0 hält diese Klasse unverändert die semantischen Attribute des Gebäudes, wie etwa *yearOfConstruction*, *YearOfDemolition*, *storeysAboveGround*, *storeysBelowGround*, *storeyHeightsAboveGround* und *storeyHeightsBelowGround*. Die Referenzierung der externen Codelisten für *class*, *function*, *usage* und *rooftype* wird in CityGML 2.0 durch *gml:CodeType* realisiert. Wie bei allen Klassen in CityGML enthält das Attribut *class* eine Klassifikation des Gebäudes, etwa *habitation* (engl.: Wohnraum) oder *business* (engl.: Geschäft). Die Attribute *function* und *usage* beinhalten Informationen über die geplante bzw. tatsächliche Nutzung des Gebäudes, etwa *holiday house* (engl.: Ferienwohnung) oder *public building* (engl.: öffentliches Gebäude).

Die Klasse *_AbstractBuilding* kann entweder zu *Building* oder *BuildingPart* spezialisiert werden, wodurch die Repräsentation einer beliebig tiefen Hierarchie von zusammenhängenden Gebäuden und Gebäudeteilen



Verändert nach Gröger et al., S. 67

Abb. 3: Mögliche Repräsentation eines Gebäudes in LoD0 durch a) den Grundriss *(lod0Footprint)* und b) die Fläche der projizierten Traufkante *(lod0RoofEdge)*

möglich ist. Nicht zusammenhängende Gebäudekomplexe, wie z. B. Industrieanlagen, sind als *CityObjectGroup* zu modellieren, mit der beliebige *_CityObjects* aggregiert werden können.

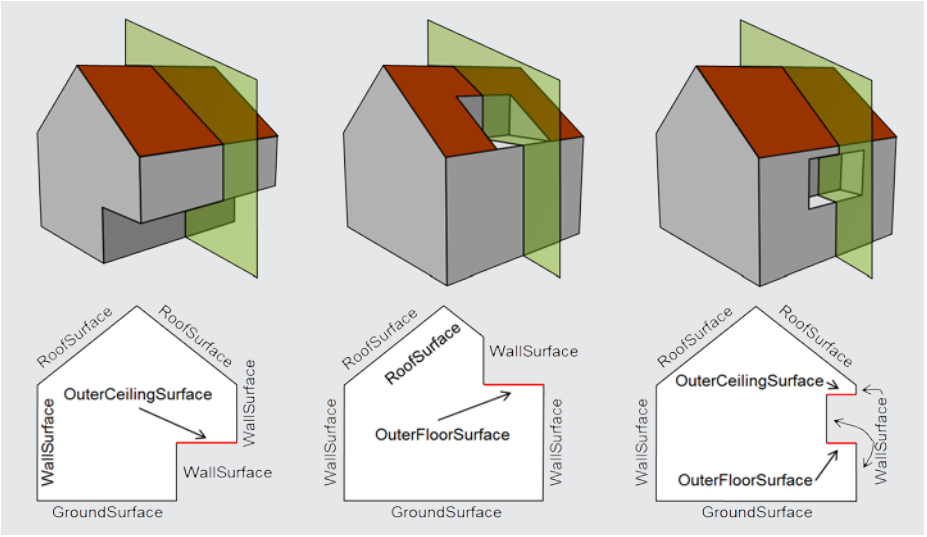
Geometrie und Semantik eines *_AbstractBuildiungs* und damit auch der abgeleiteten Klassen können in verschiedenen LoDs sukzessive verfeinert werden.

Neu in der Version 2.0 ist eine LoD0-Repräsentation von *_AbstractBuilding* als horizontale 3D-Fläche. Dafür kann sowohl der Grundriss *(lod0FootPrint)* als auch die projizierte Dachfläche *(lod0RoofEdge)* in der Form einer *gml:MultiSurface* verwendet werden (s. Abb. 3). Diese neue LoD0-Repräsentation von Gebäuden erleichtert die Integration von 2D-Daten in ein 3D-Stadtmodell. Während Katasterdaten i. d. R. den Grundriss der Gebäude vorhalten, können bei anderen topographischen Daten auch Mischformen aus Grundriss und Dachflächen auftreten, wenn die 2D-Daten etwa aus Luftbildern oder luftgestützten Laserscans gewonnen wurden.

CityGML 2.0 wartet mit einer erweiterten Version der Klasse *BuildingInstallation* zur Modellierung der Gebäudeinstallationen auf, mit der bereits seit der Version 0.3.0 die Gebäudeumhüllende weiter differenziert werden kann. Gebäudeinstallationen sind permanent an das Gebäude gebundene Bauteile, wie etwa Balkone, Gauben oder Schornsteine. In CityGML 2.0 ist es nun möglich, die Umhüllende der Gebäudeinstallationen durch die

Assoziation *boundedBy* semantisch zu differenzieren und so z. B. auch Dachflächen von Gauben tatsächlich als solche auszuweisen. Diese Neuerung gilt ebenfalls für innere Gebäudeinstallationen, etwa Kamine oder Radiatoren, die ab dem LoD4 durch die Klasse *IntBuildingInstallation* modelliert und entweder mit einem Gebäude oder direkt mit einem Raum assoziiert werden können.

Mit der Version 2.0 sind die zwei neuen Begrenzungsflächen *OuterFloorSurface* (engl.: äußere Bodenfläche) und *OuterCeilingSurface* (engl.: äußere Deckenfläche) als Spezialisierung von *_BoundarySurface* in das Gebäudemodul und in die neuen Bücken- und Tunnelmodule (s. Kap. 2.3 sowie Kap. 2.4) eingeführt worden. Als äußere Bodenfläche sollen diejenigen Flächen eines Gebäudes modelliert werden, die nahezu horizontal nach oben orientiert sind, z. B. der Boden einer Dachterrasse. Gegensätzlich dazu verhalten sich die äußeren Deckenflächen, z. B. die Decke einer Loggia. Damit wird eine weitere Differenzierung der Begrenzungsflächen bei Gebäuden und Gebäudeinstallationen möglich. Einige Beispiele zur Anwendung der in CityGML 2.0 eingeführten Klassen zeigt Abb. 4.



Verändert nach Gröger et al., S. 70

Abb. 4: Beispiele für die Modellierung einzelner Gebäudebegrenzungsflächen ab LoD2 entlang der Schnittfläche (grün)

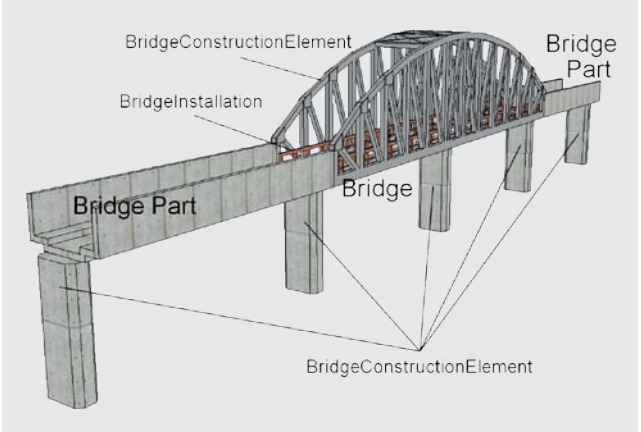
2.3 Das neue Brückenmodul *Bridge*

CityGML 2.0 enthält gegenüber der Version 1.0 zwei neue Module zur Repräsentation von Brücken und Tunneln. Die Notwendigkeit dieser zwei Module wurde in der Praxis beim Aufbau von 3D-Stadtmodellen ersichtlich. Beide Objektarten interagieren, anders als das Gebäude, in besonderer Weise mit dem Gelände. Zudem fehlte im Gebäudemodul die Möglichkeit, spezielle semantische Informationen für Brücken und Tunnel abzulegen, etwa die Information, ob eine Brücke beweglich ist oder nicht.

Das Brückenmodul (Namensraum *brid*) lehnt sich stark an das Gebäudemodul an. So gibt es auch hier eine zentrale Klasse *_AbstractBridge*, die zu *Bridge* und *BridgePart* spezialisiert werden kann und die Abbildung einer beliebig tiefen Hierarchie von zusammenhängenden Brücken und Brückenteilen ermöglicht. Die Klasse *_AbstractBridge* beinhaltet analog zur Klasse *_AbstractBuilding* die Attribute *class*, *function* und *usage* sowie *yearOfConstruction* und *yearOfDemolition*. Dabei beschreibt das Attribut *class* die unterschiedlichen Konstruktionstypen von Brücken. Damit ist es möglich, in einem Datensatz gewisse Brückenarten zu suchen, ohne deren Geometrie detailliert untersuchen zu müssen. Das Attribut *function* beschreibt die vom Konstruktionstyp unabhängige Nutzung, wie z. B. *roadway bridge* (engl.: Straßenbrücke) oder *foot bridge* (engl.: Fußgängerbrücke). Im Attribut *usage* hingegen ist die tatsächliche, primäre Nutzung der Brücke abgelegt, die von der geplanten abweichen kann. Ergänzend zu *_AbstractBuilding* weist die *_AbstractBridge* das Attribut *isMoveable* zur Kennzeichnung von Klapp- oder Drehbrücken auf. Auf eine vollständige Modellierung von Bewegungsparametern einer schwenk- oder klappbaren Brücke wurde in der ersten Version des neuen Brückenmoduls bewusst verzichtet.

CityGML 2.0 sieht auch für Brücken eine stark skalierbare Repräsentation vor. So können Brücken in den LoDs 1–4 modelliert werden, wodurch sich neben der geometrischen auch die semantische Differenzierung erhöht. Für den LoD1 ist dabei zu beachten, dass eine Klötzchengeometrie nicht vom Boden ausgehen muss, sondern auch schweben kann.

Eine Besonderheit des Brückenmoduls ist die Klasse *BridgeConstructionElement*, die in allen LoDs modelliert werden kann. Brückenkonstruktionselemente sind all diejenigen Brückenteile, die wesentlich zur Stabilität der Brücke beitragen. Hier sind etwa Pylone, Pfeiler, Tragseile, Fundamente oder ähnliches zu nennen (s. Abb. 5).



Quelle: Karlsruher Institut für Technologie

Abb. 5: Beispiel für die Modellierung einer Brücke

Die Klasse *BridgeConstructionElement* enthält lediglich die Attribute *class*, *function* und *usage*. Brückenkonstruktionselemente können neben den in den verschiedenen LoDs zur Verfügung stehenden Geometriekonzepten auch durch implizite Geometrien differenziert werden. Zur weiteren geometrischen und semantischen Verfeinerung der Außenhülle einer Brücke um nicht tragende Elemente, etwa Geländer oder Signaleinrichtungen, ist die Klasse *BridgeInstallation* vorgesehen.

Das in CityGML 2.0 neu eingeführte Brückenmodul sieht auch die Modellierung von Innenräumen im LoD4 vor. Analog zum Gebäudemodul stehen dabei die Klassen *BridgeRoom* und *IntBridgeInstallation* mit den entsprechenden geometrischen Repräsentationen zur Verfügung.

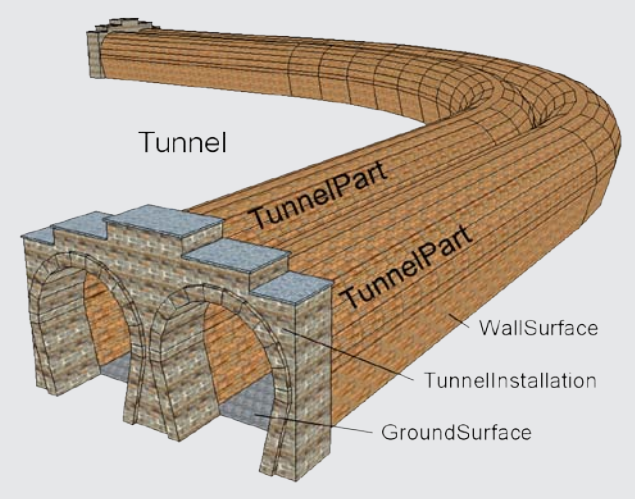
2.4 Das neue Tunnelmodul *Tunnel*

Das Tunnelmodul (Namensraum *tun*) ist ebenfalls in starker Anlehnung an das Gebäudemodul neu in CityGML 2.0 eingeführt worden. Analog dazu kann die zentrale Klasse *_AbstractTunnel* zu *Tunnel* oder *TunnelPart* spezialisiert werden. Über diese Spezialisierung erhalten diese die Attribute *class*, *function*, *usage*, *yearOfConstruction* und *yearOfDemolition*. Weitere Attribute können einem *_AbstractTunnel* nicht hinzugefügt werden. Während einfache Tunnel mit der Klasse *Tunnel* repräsentiert sind, dient der *TunnelPart* der Modellierung von Tunneln, die mehrere Transportröhren oder differenzierbare Eingangsbereiche haben (s. Abb. 6). Hierfür bieten die Codelisten für *function* und *usage* in CityGML 2.0 allerdings noch wenige Differenzierungsmöglichkeiten. So fehlen etwa die wichtigen Nutzungsmöglichkeiten *escape way* (engl.: Fluchtweg) oder *flue* (engl.: Belüftungsschacht). Diese und auch alle anderen Codelisten in CityGML 2.0 können allerdings vom Anwender problemlos erweitert werden (s. Kap. 3.2).

Auch das Tunnelmodul lässt eine zunehmende geometrische und semantische Differenzierung durch die LoDs 1–4 zu. Analog zu Gebäude- und Brückenmodul ist dabei die Ausgestaltung des Innenraumes ausschließlich im LoD4 vorgesehen. Die LoDs 1–3 dienen weiterhin der Verfeinerung der äußeren Hülle eines Tunnels als Grenzfläche zwischen Erdreich, Wasser oder Außenluft. Obwohl gerade bei unterirdischen Objekten die Innenräume von großer Bedeutung sind, wird von einer Aufweichung des LoD-Konzeptes bei Tunneln abgesehen. Diese zöge eine notwendige Trennung zwischen unterirdischen und überirdischen LoD-Konzepten nach sich. Ein aus mehreren unter- und überirdischen Tunnelteilen aggregierter Tunnel müsste sonst durch eine semantisch nicht zu rechtfertigende Trennfläche geteilt werden, um zwischen den von der relativen Lage der Objekte zur Geländeoberfläche abhängigen LoD-Konzepten zu unterscheiden. Der Nachteil dieses konsequent eingehaltenen LoD-Konzeptes ist dafür eine mögliche Durchdringung von Tunnelobjekten in den LoDs 1–3 mit Objekten, die innerhalb des Tunnels verlaufen, wie etwa Straßen oder Schienen. Allerdings ist es in CityGML ohne weiteres möglich, Objekte verschiedener LoDs in einem Modell zu halten, sodass ein Tunnel unabhängig von anderen Objekten stets in LoD4 repräsentiert werden kann.

In den LoDs 2–4 kann die Umhüllende eines Tunnels, wie auch die des Gebäudes mit der Assoziation *boundedBy* durch Spezialisierungen der abstrakten Klasse *_AbstractBoundarySurface* semantisch weiter differenziert werden. Zusätzlich können auch nicht bewegliche zum Tunnel gehörige Objekte, etwa Treppen oder Geländer, mit der Klasse *TunnelInstallation* repräsentiert werden, die auch als implizite Geometrien modelliert werden können.

Zur Innenraummodellierung stehen analog zu der Klasse *Room* des Gebäudemoduls der *HollowSpace* (engl.: Hohlraum) und die *IntTunnelInstallation* zur Verfügung.



Quelle: Karlsruher Institut für Technologie

Abb. 6: Beispiel für die Modellierung eines Tunnels

Somit können z. B. U-Bahnhöfe begehbar modelliert werden. Hohlräume sollen geschlossen sein und genau einem *Tunnel* oder *TunnelPart* zugeordnet werden. Sie können mit der Klasse *TunnelFurniture* bewegliche Objekte enthalten. Anders als für die Klassen *_AbstractBuilding* und *_AbstractBridge* sind Adressen für Tunnel in CityGML 2.0 nicht vorgesehen.

Eine besondere Bedeutung bei Tunneln kommt der auch in den Modulen *Building* und *Bridge* modellierten *TerrainIntersection* zu, die in allen LoDs mittels *gml:MultiCurve* realisiert werden kann. Sie repräsentiert die exakte Position des Überganges zwischen einem Tunnel und dem gegebenen Geländemodell. Dadurch wird ein unerwünschtes Abtauchen oder Schweben über dem Gelände verhindert, das auch durch die parallele Haltung unterschiedlicher LoDs entstehen kann.

3 Erweiterung und Anpassung von CityGML

CityGML ist ein internationaler Standard für die generischen Aspekte virtueller, dreidimensionaler Stadtmodelle. Deshalb kann der Standard weder spezifische Verhältnisse einzelner Länder, wie z. B. baurechtliche Vorschriften abbilden, noch spezielle Fachinformationen vollständig modellieren. Trotz seiner hohen informationstechnischen Komplexität kann CityGML alleine die Anforderungen spezieller Fachanwendungen meist nicht erfüllen. Um diese Situation zu verbessern, unterstützt der Standard verschiedene Methoden und Konzepte, die seinen Funktionsumfang einschränken, anpassen oder erweitern.

3.1 Profile

Für eine funktionelle Einschränkung des Standards oder eine Verschärfung der Regeln kann es unterschiedliche Gründe geben. Zum einen wird für viele Anwendungen nicht die gesamte Funktionalität des Standards gebraucht. Eine Einschränkung auf die tatsächlich benötigte Teilmenge von Klassen, Attributen und Relationen verringert insbesondere die Komplexität von Datenbank-Schemata und kann damit die Zuverlässigkeit und Performance von Anwendungen verbessern. Zum anderen sind fast alle Attribute und Relationen von CityGML-Klassen optional. Daher kann vielfach dieselbe Information (z. B. die Geometrie der Außenhülle eines Gebäudes) auf unterschiedliche Art und Weise ausgedrückt werden. Dies kann immer dann zu Problemen führen, wenn für bestimmte geometrische oder semantische Auswertungen ein Mindestinformationsgehalt notwendig ist.

Formal definierte Einschränkungen oder Verschärfungen eines GML-basierten Standards werden als Profile bezeichnet. CityGML lässt zwei verschiedene Arten von Profilen zu:

- Der modulare Aufbau des Standards (s. Kap. 2) ermöglicht die Definition *vertikaler Profile*. Ein derartiges Profil unterstützt nur eine Teilmenge der CityGML-Module (s. Abb. 1), zu der allerdings immer das CityGML-Basismodul gehören muss.
- Ein *horizontales Profil* dient dazu, über einzelne Module hinweg die Schemaregeln zu verschärfen. Dies kann dadurch geschehen, dass die Verwendung bestimmter Klassen verboten wird, die Verwendung bestimmter optionaler Attribute einer Klasse verboten wird, oder die Verwendung bestimmter optionaler Attribute zwingend vorgeschrieben wird.

Bei der Definition horizontaler Profile ist wichtig, dass die Profilregeln eine echte Teilmenge der CityGML-Schemaregeln darstellen, damit die einem bestimmten Profil genügenden Instanzdokumente auch gültige CityGML-Dokumente sind. Ein horizontales Profil sollte möglichst durch ein eigenes XML-Schema definiert werden, damit die Einhaltung der Profilregeln einfach überprüfbar ist. Es sind allerdings auch Einschränkungen denkbar, die mit den syntaktischen Möglichkeiten der W3C-Empfehlung XML-Schema nicht auszudrücken sind. Diese sollten dann in einer anderen Spezifikationsmethode, wie der Object Constraint Language (OCL) (Warmer und Kleppe 2003) formal definiert werden.

3.2 Codelisten

Eines der wichtigsten Konzepte zur semantischen Anpassung und Erweiterung von CityGML sind Codelisten. Eine Codeliste zählt die endlich vielen zulässigen Werte auf, die ein bestimmtes Attribut annehmen kann. Beinahe jede CityGML-Klasse hat mehrere derartiger Attribute, die z. B. eine Grobklassifikation des Objektes wiedergeben oder seine vorgesehene bzw. aktuelle Funktion beschreiben. Im Gegensatz zum ebenfalls verwendeten Konzept der Enumerationen werden Codelisten nicht im XML-Schema repräsentiert, sondern durch externe GML-Dictionaries. Diese Dictionaries definieren nicht nur den Satz der zulässigen Codes, sie ordnen im Regelfall auch jedem Code einen eindeutigen Identifier sowie einen freien Text zu, der die semantische Bedeutung des Codes erklärt. Instanzdokumente von CityGML 2.0 verwenden diese Codes und können zusätzlich eine (optionale) Referenz auf das zugehörige Dictionary enthalten. Diese Referenz kann in Form einer URL oder URN ausgeführt sein.

Die CityGML-Spezifikation enthält Vorschläge für eine ganze Anzahl von Codelisten, die aber nicht normativ sind. Damit steht es jedem Nutzer oder Implementierer des Standards frei, diese Codelisten abzuändern, zu erweitern oder vollkommen neue Listen zu generieren, um die Semantik der betroffenen Attribute an seine Anforderungen anzupassen. Allerdings kann das theoretische Konzept der Codelisten ohne eine informationstechnische Infrastruktur zur Bereitstellung derselben nicht genutzt werden. Leider macht der Basisstandard GML keinerlei Vorgaben, wie z. B. Codelisten-Referenzen zu interpretieren sind oder wie entsprechende Dienste, die Codelisten bereitstellen, aufgebaut sein sollen. Prinzipiell ließe sich ein Registry-Service nach ISO 19135 verwenden. Diese ISO-Norm beschreibt aber nur ein allgemeines Konzept, das für den praktischen Einsatz noch präzisiert werden muss.

3.3 Generische Objekte und Attribute

Mit Hilfe des Codelisten-Konzeptes ist es nur möglich, schon existierende Attribute in CityGML-Klassen semantisch anzupassen oder zu erweitern. Es ist damit nicht möglich, einer existierenden Klasse zusätzliche Attribute zu geben oder gar vollständig neue Klassen zu definieren. Für diesen Zweck sieht der Standard zwei weitere Konzepte vor:

- In jeder CityGML-Klasse kann die Liste der im Standard festgelegten Attribute um sogenannte generische Attribute erweitert werden, um zusätzliche Eigenschaften eines Objektes abzubilden;
- Objekte der realen Welt, die mit keiner der vorhandenen CityGML-Klassen semantisch korrekt beschrieben werden, können durch eine Klasse für generische Objekte *(GenericObject)* modelliert werden.

Bei beiden Erweiterungskonzepten gibt es allerdings eine Reihe von Einschränkungen, die zu beachten sind. So kann die semantische Bedeutung eines generischen Attributs nur über den Namen des Attributs spezifiziert werden, der als freier Text definiert ist. Damit kann auf Instanzebene nicht durch eine Schema-Validierung überprüft werden, ob korrekte Attributnamen verwendet werden. Zwar können die Datentypen generischer Attribute auf Schemaebene definiert und überprüft werden, es ist aber nur die Verwendung der einfacher Datentypen *Text*, *Integer*, *Double*, *Datum*, *URL* und (neu in CityGML 2.0) *Measure* als *Double*-Zahl mit Angabe einer physikalischen Maßeinheit möglich. In CityGML 2.0 können solche Attribute auch optional zu Listen zusammengefasst werden, denen ein Name zugewiesen werden muss. Damit können generische Attribute keine Relationen auf Geometrieobjekte oder andere Klassen enthalten, auch die Referenzierung auf Enumerationen oder Codelisten ist in einem generischen Attribut nicht möglich. Darüber hinaus ist die Erweiterungsklasse *GenericObject* nur mit den Standardattributen zur Klassifikation und Festlegung von Funktion und Nutzung (*class*, *function*, *usage*) ausgestattet. Alle weiteren Eigenschaften des Objektes müssen als generische Attribute abgebildet werden.

3.4 Das ADE-Konzept

Die oben beschriebenen Einschränkungen von generischen Attributen und Objekten führten in CityGML zur Entwicklung des Erweiterungskonzeptes *Application Domain Extension* (ADE), das mittlerweile auch in anderen GML-basierten Standards, wie XPlanGML (Benner et. al. 2009) verwendet wird. Jede ADE wird durch ein spezielles XML-Schema in einem eigenen Namensraum repräsentiert, gegen das Instanzdokumente geprüft werden können. Ein ADE-Schema baut auf den CityGML-Schemata auf, wobei zwei verschiedene Arten der Ableitung unterstützt werden:

- CityGML-Klassen können um zusätzliche Attribute oder Relationen, die im Namensraum der ADE definiert sind (zusammenfassend als ADE-Attribute bezeichnet), ergänzt werden. Die ADE-Attribute ersetzen die generischen Attribute, lassen aber auch Relationen auf Geometrieobjekte, Klassen des CityGML- oder ADE-Schemas sowie die Verwendung von Enumerationen oder Codelisten zu.
- Es können in der ADE neue Klassen definiert werden, die optional mit dem Konzept der Generalisierung von existierenden CityGML-Klassen abgeleitet werden können und damit deren Attribute und Relationen erben. Dieses Konzept ersetzt die Verwendung generischer Objekte, ist aber wesentlich mächtiger, weil die Syntax der zusätzlichen Klassen in einem eigenen XML-Schema definiert ist.

Instanzdokumente einer CityGML-ADE beinhalten kein reines CityGML und validieren deshalb auch nicht gegen das CityGML-Schema, sondern nur gegen das spezielle ADE-Schema. Solange Klassen aber nur um zusätzliche ADE-Attribute erweitert werden, können prinzipiell CityGML-Software-Werkzeuge diese Attribute richtig zuordnen und sollten in der Lage sein, die Werte zumindest anzuzeigen. Eine Verarbeitung der zusätzlichen Informationen unter Ausnutzung der spezifischen semantischen Bedeutung ist allerdings nur in Spezialsoftware möglich.

Die spezielle Art und Weise, wie ADE-Attribute mit den erweiterten CityGML-Klassen in Verbindung stehen, bedingt einige Einschränkungen, die bei der Spezifikation von ADE-Schemata und der Generierung valider Instanzdokumente zu beachten sind:

- Die Namen der zusätzlichen ADE-Attribute müssen im Namensraum der ADE eindeutig sein. Dies gilt auch, wenn sie unterschiedliche Basisklassen des CityGML-Schemas erweitern.
- ADE-Attribute und -Relationen sind stets optional und können in Instanzdokumenten beliebig oft verwendet werden.
- In gültigen Instanzdokumenten müssen die ADE-Attribute immer nach den CityGML-Attributen einer erweiterten Klasse aufgeführt sein.

4 Zusammenfassung und Ausblick

Der Standard CityGML für die Repräsentation und den Austausch virtueller 3D-Stadtmodelle bietet ein wohldefiniertes semantisches Modell in mehreren fest definierten Detailierungsgraden. Als Basismodell in der Tradition semantischer Modelle der deutschen Vermessungsverwaltung wie die ALK, das ATKIS® und das ALKIS® definiert CityGML die Objektarten, Relationen und Attribute, die grundlegend für die wesentlichen Anwendungen von 3D-Stadtmodellen sind, und ermöglicht einen interoperablen Austausch solcher Modelle über Geodateninfrastrukturen. Für anwendungsspezifische Erweiterungen wie umweltrelevante Simulationen oder den Denkmalschutz steht das Konzept der ADEs bereit. In der Version 2.0 wurde das semantische Modell von CityGML um Module für Brücken und Tunnel, weitere thematische Begrenzungsflächen, die explizite Angabe der Lage von Objekten zum Gelände und die explizite Referenzierung

von Codelisten für Attribute erweitert, um auch in diesen Bereichen Interoperabilität sicherzustellen.

Bei der Entwicklung von CityGML 2.0 stand im Vordergrund, die Aufwärtskompatibilität zur Version 1.0 weitestgehend zu erhalten. Viele als wichtig einzustufende Änderungen und Erweiterungen konnten deshalb in dieser Version noch nicht realisiert werden, oder sie mussten aus Zeitgründen zurückgestellt werden. Dazu gehören u. a.:

- die Überarbeitung zentraler Konzepte wie z. B. des LoD-Konzeptes,
- die Umstellung von CityGML auf die GML-Version 3.2.1,
- die Komplettierung der Spezifikation, insbesondere durch formale Definitionen für alle CityGML-Klassen, Attribute und Relationen,
- die Überarbeitung des konzeptionellen UML-Modelles und der XML-Schemadateien, sodass die im GML-Standard (ISO 19136) definierten Namenskonventionen und Abbildungsregeln bei der Transformation von UML nach XML-Schema eingehalten werden.

Vor allem der letzte Punkt ist von zentraler Bedeutung, damit die Weiterentwicklung und Pflege des CityGML-Datenmodells und der zugehörigen Dokumentation zukünftig mit effektiven Softwarewerkzeugen unterstützt werden kann. Die AG Modellierung der SIG 3D hat mit der Bearbeitung dieser Punkte bereits begonnen, und wird sich darüber hinaus auch um eine inhaltliche Weiterentwicklung des Standards kümmern. Eine Grundlage dafür wird u. a. das im Rahmen der INSPIRE-Initiative entwickelte Gebäudemodell sein.

Dank

Wir danken den weiteren Mitgliedern der Arbeitsgruppe Modellierung der SIG 3D für die Mitwirkung an der Erstellung der Version CityGML 2.0.

Literatur

Benner, J., Eichhorn, T., Krause, K.-U., Kirchenbauer, V.: Konzepte länderspezifischer Erweiterungen standardisierter Objektmodelle am Beispiel des Standards XPlanung der Freien und Hansestadt Hamburg. In: Schrenk, M., Popovich, V. V., Zeile, P. (Hrsg.) »Proc. REAL CORP 2010, Wien, 18.–20. Mai 2010, S. 375–382, 2010.

citygml.org: Offizielle CityGML-Internetpräsenz. www.citygml.org, letzter Zugriff 6/2012.

Cox, S., Daisey, P., Lake, R., Portele, C., Whiteside A. (Hrsg.): OpenGIS® Geography Markup Language Implementation Specification, Version 3.1.1, OGC Doc No. 03-105r1, Open Geospatial Consortium, 2004.

Foley, J., van Dam, A., Feiner, S., Hughes, J.: Computer Graphics: Principles and Practice. Addison Wesley, 2nd Ed., 2004.

Gröger, G., Benner, J., Dörschlag, D., Drees, R., Gruber, U., Leinemann, K., Löwner, M.-O.: Das interoperable 3D-Stadtmodell der SIG 3D. Zeitschrift für Geodäsie, Geoinformation und Landmanagement, 6/2005: 343–353, 2005.

Gröger, G., Kolbe, T. H., Czerwinski, A. (Hrsg.): City Geography Markup Language (CityGML). OGC Best Practices Document, Version 0.4.0, OGC Doc. No. 07-062, Open Geospatial Consortium, 2007.

Gröger, G., Kolbe, T. H., Czerwinski, A., C. Nagel (Hrsg.): OpenGIS® City Geography Markup Language (CityGML) Encoding Standard, Version 1.0.0, OGC Doc No. 08-007r1, Open Geospatial Consortium, 2008.

Gröger, G., Kolbe, T. H., Nagel, C., Häfele, K.-H. (Hrsg.): OGC City Geography Markup Language (CityGML) Encoding Standard, Version 2.0, OGC Doc No. 12-019, Open Geospatial Consortium, 2012.

Gruber, U.: Die dritte Dimension im amtlichen Vermessungswesen Deutschlands. In: Löwner, M.-O. Hillen, F. und Wohlfahrt, R. (Hrsg.): Geoinformatik 2012 »Mobilität und Umwelt«. Konferenzband zur Tagung Geoinformatik 2012 vom 28.–30.03.2012, Braunschweig, S. 303–309, 2012.

Kolbe, T. H.: CityGML, KML und das Open Geospatial Consortium. In: Tagungsband zum 13. Fortbildungsseminar Geoinformationssysteme des Runder Tisch e. V., 2008.

Kolbe, T. H.: Representing and Exchanging 3D City Models with CityGML. Proceedings of the 3rd International Workshop on 3D Geo-Information, Seoul, Korea, 15–31, 2008.

OASIS (2003): xNAL Name and Address Standard. Organization for the Advancement of Structured Information Standards. www.oasis-open.org/committees/ciq/ciq.html#6, letzter Zugriff 6/2012.

OGC (2012): Internetseite des Open Geospatial Consortiums. https://portal.opengeospatial.org, letzter Zugriff 6/2012.

Reed, C. (Hrsg.): Policy Directives for Writing and Publishing OGC Standards: TC Decisions, OGC Doc 06-135r11, Open Geospatial Consortium, 2011.

SIG 3D: Offizielle Internetpräsenz der Special Interest Group der Geodateninfrastruktur Deutschland (GDI-DE). www.sig3d.org, letzter Zugriff 6/2012.

Warmer, J. und Kleppe, A.: The Object Constraint Language: Getting Your Models Ready for MDA (2nd Edition), Addison-Wesley Professional, 2003.

Anschrift der Autoren

Prof. Dr.-Ing. Marc-O. Löwner
Institut für Geodäsie und Photogrammetrie
Technische Universität Braunschweig
Postfach 3329, 38023 Braunschweig
m-o.loewner@tu-bs.de

Dr.-Ing. Joachim Benner | Dipl.-Ing. Karl-Heinz Häfele
Institut für Angewandte Informatik
Karlsruher Institut für Technologie (KIT)
Postfach 3640, 76021 Karlsruhe
joachim.benner@kit.edu | karl-heinz.haefele@kit.edu

Priv.-Doz. Dr. rer. nat. Gerhard Gröger
Institut für Geodäsie und Geoinformation, Universität Bonn
Meckenheimer Allee 172, 53115 Bonn
groeger@igg.uni-bonn.de

Dipl. Ing. Ulrich Gruber | Dipl. Ing. Sandra Schlüter
Fachdienst 62 – Kataster und Geoinformation, Kreis Recklinghausen
Kurt-Schumacher-Allee 1, 45657 Recklinghausen
ulrich.gruber@kreis-recklinghausen.de
sandra.schlueter@kreis-recklinghausen.de