

Creation and Management of Object-Based Terrain Models of Urban Environments

Lutz ROSS, Birgit KLEINSCHMIT, Henrik BUCHHOLZ, Jürgen DÖLLNER

Lutz Ross, Birgit Kleinschmit, Berlin Technical University, Institute of Landscape Architecture and Environmental Planning, Straße des 17. Juni 145, Berlin, {lutz.ross; birgit.kleinschmit}@tu-berlin.de

Henrik Buchholz, Jürgen Döllner, Hasso-Plattner-Institut Potsdam, Professor-Doktor-Helmert-Str. 2-3, Potsdam, {Doellner; buchholz}@hpi.uni-potsdam.de

1 INTRODUCTION

Ongoing research in Computer Graphics, Geoinformation Science and Database Design provides the possibility to create very large and detailed, interactively explorable 3D-city and 3D-landscape models. State-of-the-art technologies and methods not only provide the opportunity to acquire and display 2.5D and 3D content, but to create realistic geovirtual environments, which can be operated in real-time on consumer computers. Geotainment products such as 3D RealityMaps (<http://www.reality-maps.de/>), or virtual globes such as Google Earth (<http://earth.google.com/>) are the best example for the progresses achieved in the recent years. These ‘products’ usually combine digital terrain models and high-resolution remote-sensing data with more or less detailed 3D models of structures, vegetation, and a virtual sky. Despite the progresses made, building realistic and information rich GeoVE still requires high effort in creating and managing 3D content and because of that effort, most 3D city models only encompass 3D models of large or eye-catching ‘real world’ objects, such as buildings or trees so far. Structures that cover and structure the natural terrain such as roads, pavements, retaining walls or squares are seldom integrated into virtual 3D city models. As a result the terrain geometry and texture in most 3D city models looks flat and unrealistic, whenever a user navigates in a pedestrian perspective. From a methodology point of view, the reasons therefore mainly depend on limitations of commonly used digital terrain models as regular grid in raster format or as triangular irregular networks. These models are restricted to resemble the terrain surface as a 2.5D surface. Because of this restriction, even very high resolution terrain models are not suited to represent the geometrical structure of typical man-made surface covering structures such as walls, kerbstones, or stairs. Moreover, the missing ‘object orientation’ of common terrain models neglects semantic aspects, as well as spatial relationships concerned with the discrete objects, which cover the terrain in urban environments. Object-based terrain models, in contrast, would enable us to link spatial temporal, economic or other semantic information to the visual elements of a GeoVE. Therefore, GeoVE aiming at imitating ‘the real world’, enabling users to explore near photorealistic environments from the perspective of a pedestrian, and at the same time providing access to information related to the spatial structures, such as models made for public participation, 3D navigation or the management of land, rely on very detailed and object-based models of (ground-related) structures.

The few examples including detailed 3D representations of open-space structure or traffic areas are based on manual modeling techniques. Manual modeling of 3D geometry is an established method and many software solutions are available to model 3D geometry and to texture these models. It provides very high flexibility with respect to geometric complexity and visual detail, but at the same time manual modeling techniques commonly require data preparation, data conversion, and additional expertise, and in most cases the resulting models are disconnected from the original geodata. Thus, manual modeling seems to be appropriate whenever a very high level of visual detail is needed and at the same time only few geoinformation have to be integrated into GeoVE. But if the aim is to develop city-wide representations of ground structures, which are linked to administrative databases, manual modeling techniques are too limited and inefficient.

To overcome these limitations we will present an approach which enables the semi-automatic creation of object-based terrain models in urban environments from 2D cadastre data. Therefore, an object-oriented modeling concept to process detailed urban land covering data is developed by combining the SmartTerrain modeling approach (Buchholz et al. 2006) with a GIS-based management and configuration system for data preparation.

2 METHODS & MATERIALS

SmartTerrain (ST) models are based on the idea to transform existing 2D geo-referenced vector features into 2.5D and 3D features. The idea is implemented by defining a class model of six basic structure elements (Table 1) and class-specific transformation algorithms. Each ST class possesses a set of parameters, which

can be used to configure geometry and material properties. These configuration parameters are used as input parameters for transformation functions that convert polyline and polygon features into 2.5D or 3D objects. Geometry generation and texturing algorithms have therefore been developed and integrated into a prototypic ST editor. The editor is implemented on the basis of LandXplorer technology (3DGeo GmbH, <http://www.3dgeo.de/>). It reads shapefiles and creates objects from feature geometry and configuration parameters which are managed in the attribute tables of the input features. Besides visualization and object-configuration capabilities, a material catalog for managing texture and color materials and two prototypic rendering techniques for water surfaces and ground vegetation are integrated into the editor.

ST class	Description
GroundArea	Class to represent 2.5D surfaces; provides access to an extrusion function (e.g. traffic lanes or pavements can be depicted as extruded features).
WaterArea	Class to represent water surfaces; uses an hardware-shader to simulate a vivid water-surface.
Stair	The abstract class is extended by the classes RegularStairs that can be interactively configured by few parameters and IrregularStairs which are composed out of a polygon stack.
Kerb	Class for the representation of kerb-type objects; objects are created from polyline features by buffering and extrusion.
Wall	Class for the representation of wall-type objects; objects can be represented by polygon or polyline features; extends the base functionality of Kerb by adding a wall topping.
Barrier	Class for the representation of barriers; provides a solution to depict e.g. fences by extruding line features and applying a texture that supports alpha channels to define transparent picture parts.

Table 1: Initial SmartTerrain classes and a short description of their functionalities

In the following section an example will be presented explaining the modeling of a ST model from the digital town map (“Stadtgrundkarte”, STGK) of Potsdam, Germany. The general approach and data preparation work have been presented in the context of a diploma thesis (Ross 2006). For this reason only a short introduction on the data used and data preprocessing is given. Following the introduction, two mapping-concepts are evaluated, which are used to map ST elements to the CityGML schema. Experiences gained during testing, development of the ST schema, preparatory work, and design of an export function, result in a modeling concept that facilitates the modeling process and at the same time considers integration of additional data, hierarchical structuring, and compatibility to CityGML.

2.1 Introduction to SmartTerrain modeling: City Channel Case Study, Potsdam

The ‘City Channel’ case study is our developing environment. It shows a small part of the historic city centre of Potsdam in Germany and is modeled from the STGK. The STGK includes detailed information on buildings, transportation objects, vegetation and street inventory. It is maintained by the commune and provides a base data layer for many administrative and planning-related tasks. The detailed information on ground and ground-related structures such as material information, kerbstones, stairs, walls, changes in surface covering material, footprints of buildings, rails, gullies and other features led to the initial idea of an ‘geometry generator’ facilitating the creation of detailed object-based 3D models of ground structures in urban environments.

The data of the STGK is derived by ground survey and maintained in a GIS. It includes a detailed feature type catalog and the target scale is 1:500. Land-use boundaries and boundaries between different surface covering types are described by polyline features representing either physical 3-dimensional objects, such as walls, stairs, and kerbstones, or one-dimensional idealized borderlines between different land-use, respectively surface-covering types. Land-use information and surface-covering material of ‘ground areas’ are represented by point features. The term “ground area” is used here as a metaphor for all surface patches which are enclosed by boundary polylines; at the same time GroundArea resembles the basic ST class to represent planar features. Buildings and some other objects, such as large walls, are represented by closed polylines or polygon features, and small ground-related objects, such as gullies or hydrants, are represented

by point features. Furthermore, locations of road signs and traffic lights, trees in public-space, as well as representations of complex structures like bridges, or passages are included in the STGK.

Modelling a ST model from this source data requires extracting all polyline and polygon features representing boundaries between different land-use, surface-covering types, or structures and building an area-wide polygonal dataset that represents uniformly covered ground areas. Information on the surface covering, respectively land-use, can be transferred to the created polygon features from the point signatures. The object-type catalog of the town map lists 16 ‘artificial’ surface covering materials and 16 ‘natural’ types. For each covering type, which occurred in the study area, a material identifier and material parameters, such as color, path to texture file, and size, have been assembled in a XML-based material catalog. To create an initial ST model this polygon dataset is converted with ESRI's 3D Analyst into a 3D feature dataset. The resulting 2.5D surface can be loaded into the interactive editor (figures 1a -1d). Figure 1a depicts a screenshot of the editor with simple building models, a raster based terrain model, and a cartographic representation of the input data. The in-built geometry generator processes the input features and creates geometry and appearance from attributes stored in the input features (Figure 1b). This primary result does not depict the ‘real’ situation so far. Because of the approach to reconstruct height information from a digital terrain model, the GroundArea, Wall, and Stair polygons form a continuous 2.5D surface without vertical offsets. Thus the usual vertical offset between e.g. a traffic lane and a pavement is not depicted in the model. For the same reason the brick walls stand on the terrain, although in reality it is part of an artificial channel. The editor, therefore, provides some basic functionality to adjust the height of objects, as well as the height of single vertices. It furthermore allows to interactively manipulating configuration parameters such as texture orientation or geometry parameters.

To complete the modeling all polyline features representing physical objects such as kerbstones and walls are added to the ST feature dataset. Analog to the preparation of polygon-based features, configuration parameters for geometry generation and texturing are stored in the attribute tables. Figure 1c shows the result from adding polyline-based features and manual adjustment of heights and configuration parameters.

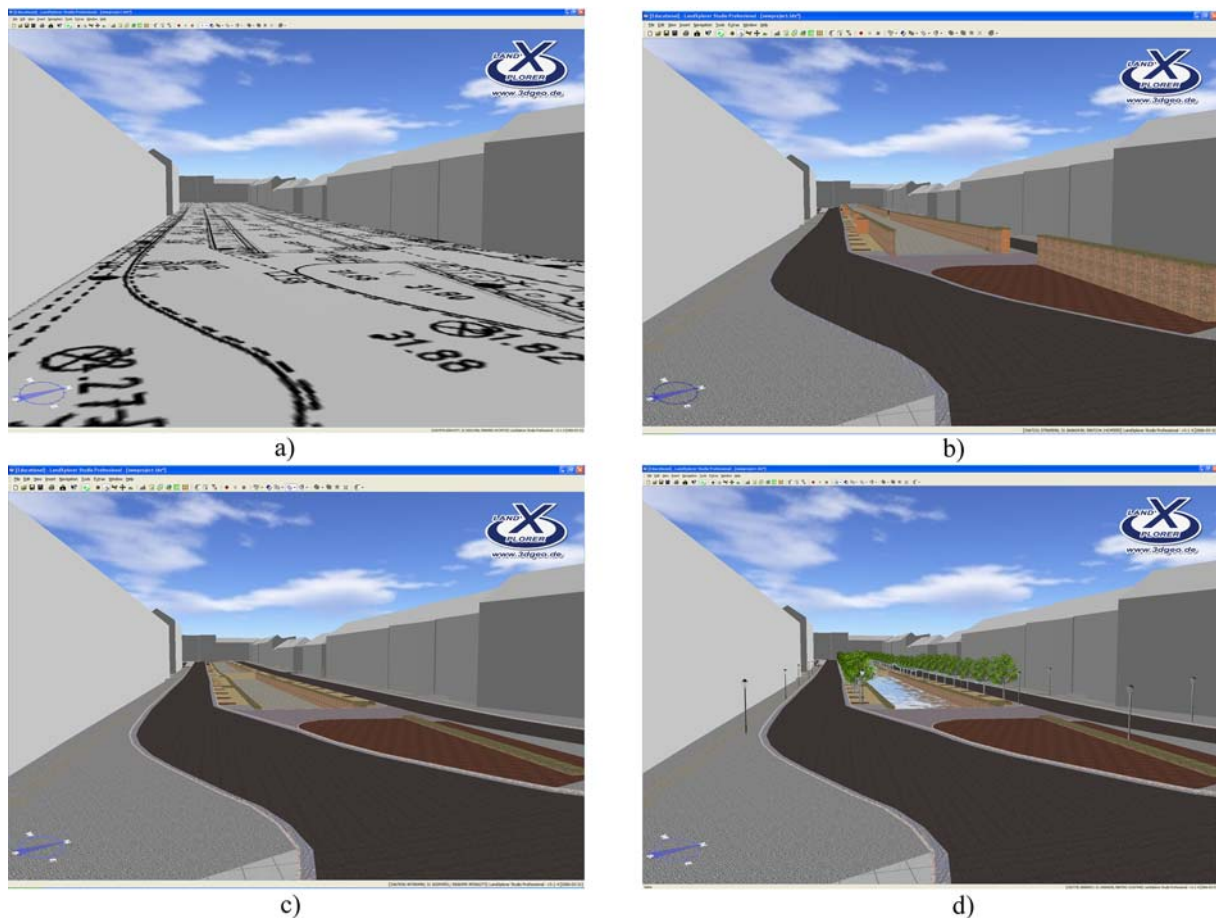


Figure 1 a-d): Screenshots of the SmartTerrain Editor showing a) a digital terrain model draped with the STGK in raster format and simple building models and b) the initial result from linking configured polygonal input features to the material catalog. In c) polyline-based features are added and heights and feature-specific texture parameters have been modified Figure 1 d) finally, shows

the usage of the prototypic shaders for the representation of water surfaces. Furthermore, 3D models of street-lights and a simple tree billboard are used to increase the overall detail of the scene.

The object-based terrain model shows high visual and geometrical detail, and a direct connection to the input features is maintained. Finally, additional elements such as models of trees and street lights are added increasing the realism of the scene (Figure 1d). The elements added in this image are: A water surface, which is rendered using a hardware-shader, street lights as 3D models, and simple billboards of trees.

2.2 Export of ST models to CityGML

The example ST model derived from STGK is still limited with respect to interoperability and semantics. In order to enhance these two aspects an export utility is currently under development. As promising potential future standard for virtual 3D-city models, CityGML is chosen as target schema for the export function. CityGML has been developed by the Special Interest Group 3D of the Geodata Initiative North Rhine – Westphalia since 2002. In 2006 it received the status of a discussion paper at the Open Geospatial Consortium (OGC 2006). CityGML is “a common information model for the representation of 3D urban objects. It defines the classes and relations for the most relevant topographic objects in cities and regional models with respect to their geometrical, topological, semantical and appearance properties” (<http://www.citygml.org/>). Besides spatial properties, the data schema includes hierarchically structured classes of city objects, aggregation, and relationships between objects. Thus CityGML provides a data schema to model structured 3D geoinformation of urban environments. In order to map the ST objects to CityGML objects, two approaches are evaluated. The first approach is to define ‘default’ CityGML values for each ST class, while the second approach is based on the definition of additional attributes, which determine the CityGML (sub-) classes and attributes, as defined by the candidate CityGML schema.

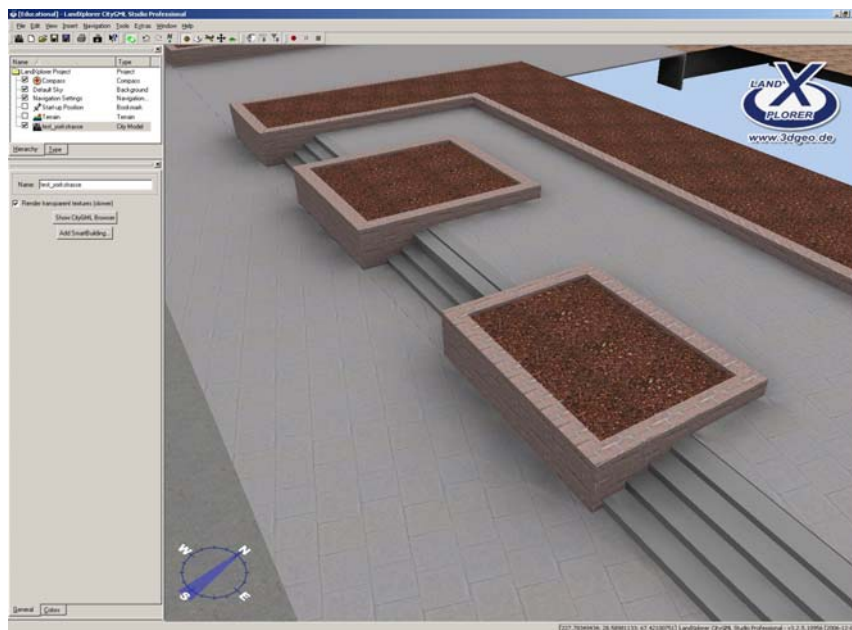


Figure 2: Detail from the City-Channel dataset created with the ST Editor and mapped to CityGML. The data is visualized in LandXplorer CityGML Viewer.

2.2.1 Export to CityGML by ‘default mapping’

A direct mapping of STM objects to CityGML would be the easiest way in terms of data creation and management. During the design of a direct mapping schema, it became apparent that additional information is needed to create a structured CityGML dataset. On the one hand this is due to the ambiguity of the ST class GroundArea, which encompasses elements from three possible CityGML subclasses: TrafficArea, AuxiliaryTrafficArea, and PlantCover, on the other hand the City Channel dataset misses information needed to create aggregation groups represented for example by the CityGML classes TransportationComplex and CityObjectGroup. Neglecting the latter problem, the five remaining ST classes (Kerb, Wall, Barrier, Stair, and WaterArea) can be mapped to CityGML. Stairs and kerbstones can best be defined by the ‘function’-attribute of TrafficArea objects (cp. CityGML schema 3.1.1, external codelist ‘TrafficAreaFunctionType’). Thus they are mapped to TrafficArea features with the respective attribute set. Walls are mapped to

GenericCityObjects with the class attribute ‘wall’, barriers are mapped to CityFurniture with the class attribute ‘1440’ (fence), and WaterAreas are mapped to WaterSurface features and the waterLevel attribute is derived from the height of the polygons that constitute water surfaces.

As stated before, with objects of the ST class GroundArea the point is to decide, whether they are mapped to PlantCover, TrafficArea or AuxiliaryTrafficArea. This can neither be decided from the ST classes, nor can it be presumed that the input data for a ST holds an explicit definition. Moreover, it might even be a domain specific question, to decide which CityGML feature type should be chosen. Therefore, CityGML allows to reference one geometric object through more than one feature. The TrafficAreaFunctionTypeList for example, lists ‘green_spaces’ and ‘flower_tubs as possible value for a TrafficArea object, but vegetation-carrying areas could as well be mapped to PlantCover or AuxiliaryTrafficArea objects. Thus it would be possible to describe a GroundArea object as being a TrafficArea feature and at the same time a PlantCover feature. For simplicity reasons the mapping function used for the City Channel dataset has been configured to map all GroundArea objects to TrafficArea objects. One exception is defined: Features having the prototypic ‘grass shader’ assigned to themselves are mapped to PlantCover objects and the height property is acquired automatically from the shader settings. The default-mapping rules have been tested with the initial City Channel data. Figure 2 depicts a detail from the mapped ST rendered in LandXplorer CityGML Viewer (3D Geo GmbH 2006).

As a result of the used approach, visual detail is maintained, but missing aggregation groups and hierarchical structuring reduce semantic detail and lead to an unstructured list of single CityGML features. Thus, the objects depicted in figure 2 know that they are part of a city model and what (sub-)class they represent, but they do not know if and how they stand in functional relation to other objects.

2.2.2 Export to CityGML by ‘attribute mapping’

The concept behind the attribute-mapping approach is to represent the target CityGML structure and semantics in the attribute tables of the input features. It introduces nine additional ‘CityGML-attribute keys’, which are added to the existing set of configuration parameters. The keys and short explanations are enlisted in table 2.

Key	Description	Values
citygmlClass	Constitutes the CityGML class of objects	Terrain related CityGML feature types: <i>TrafficArea</i> , <i>AuxiliaryTrafficArea</i> , <i>PlantCover</i> , <i>WaterSurface</i> , <i>CityFurniture</i> , <i>GenericCityObject</i>
citygmlGroup	Used to define a feature to be member of a <i>CityObjectGroup</i>	Unique ‘String’ identifier per group
transportationComplexType	Used to define the type of a <i>TransportationComplex</i>	<i>Road</i> , <i>Track</i> , <i>Railway</i> , <i>Square</i>
transportationComplexName	Assigns a unique name to <i>TransportationComplex</i> .	Unique ‘String’ identifier per <i>TransportationComplex</i>
classAttribute	Used to store class-specific attributes describing a classification of a feature type, e.g. ‘Querco-Fagetea’ as class attribute of a <i>PlantCover</i> feature.	Defined in the external codelists of CityGML
functionAttribute	Used to store class-specific attributes describing the function of a feature, e.g. ‘driving_lane’ as function attribute of a <i>TrafficArea</i> feature.	Defined in the external codelists of CityGML
usageAttribute	Used to store class-specific attributes describing the actual	Defined in the external codelists of CityGML

surfaceMaterialAttribute	usage of a feature, e.g. 'pedestrians' as usage attribute of a <i>TrafficArea</i> feature. Used to store class-specific material attributes	Defined in the external codelists of CityGML or defined in a additional codelist to gain flexibility
plantCoverHeight	Used to store the height attribute of <i>PlantCover</i> features	Any positive number

Table 2: Defined 'CityGML-Attributes' and their usage and possible values

These CityGML attributes can be used to structure the ST dataset and enable users to overrule the default-mapping rules. At the same time it requires users to think about structuring data. For further explanation, a CityGML Road feature in LoD 4 provides a good example. A Road feature is represented by a TransportationComplex, which is an aggregation of features from the sub-classes TrafficArea and AuxiliaryTrafficArea. Such information is not depicted in the original STGK data, because transportation areas are originally modeled by their boundary line and no larger spatial knowledge is embedded. Therefore, the surface patches derived from the ST modeling are not aggregated and aggregation can not be automatized. But aggregation groups could be defined by adding a 'spatial mask', which is unioned with the ST dataset. Unfortunately, union is not supported for PolygonZ geometry in ArcGIS. Therefore, to prepare a test dataset for the attribute mapping function, the existing base data has been edited manually with ArcGIS by cutting ST objects at crossroads and aggregation of ST object to CityGMLGroup and TransportationComplex features using the defined attribute keys. Using the attribute mapping approach the City Channel dataset will be mapped to a structured CityGML dataset, which will be evaluated by the developers of CityGML and, if it qualifies, will be available for download on the CityGML website (<http://www.citygml.org>).

2.3 Conceptual developments – introducing extensions to the initial ST schema

Bridges or sub-surface structures are not considered in the modeling approach, so far, although they are connected to the terrain. Figure 3 depicts a sketch of a typical sub-surface structure. The model is a design study, which shows that the already developed geometry-generation algorithms can be used to create a structure that looks like the entrance to a subway station or sub-surface passage from few polyline or polygon features.

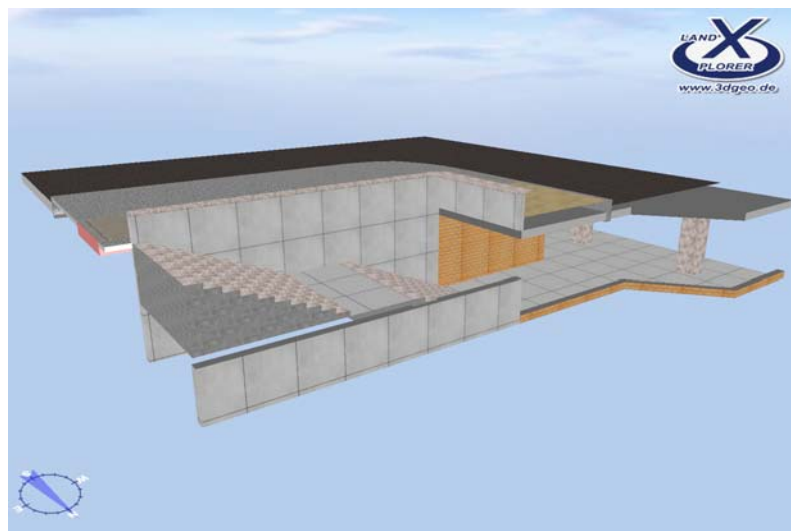


Figure 3: Design study on sub-surface objects. The depicted model is manually modeled from seven polygons representing the stairs, floor and columns. The ceiling is a copy of the floor plane and defined by a height offset and additional material information. The wall-type objects embracing the stairway and the 'room' are derived from the polygon boundaries.

It has been assembled using the interactive editing and configuration tools of the editor. The subsurface structures in the example are represented by GroundArea and Wall features representing floor, ceiling, outer walls and the three columns. The entrance or access to the subsurface structure is modeled by Wall, GroundArea, and Stair features. To create the represented 3D structure from polygonal base data, the height

of the room and the absolute height position of either ceiling or floor, the location of walls, respectively openings and the materials have to be defined. Furthermore, the geometry generator has to be instructed how the data should be assembled to a 3D representation. To realize the integration of such composite structures, the class model is extended by the class ComplexObject. Complex objects have to be defined by base geometry, a set of ST featurtypes (e.g. 'subway:wall', 'subway:floor', and 'subway:ceiling'), general object properties (e.g. 'height' and 'floorBaseHeight', 'openingSegments') and processing instructions (simplified: "Create floor from input geometry > create ceiling > move ceiling by 'height'"). To realize this concept, geometry and material configuration parameters are stored in separate cataloges, and linked to the input geometry by a 'ST feature type' identifier. ST feature types are defined by a unique identifier and a collection of parameters, which are related to vector objects in order to enable the generation of several construction parts from one base vector feature. The feature type definitions are stored in a separate table. The sub-surface room in figure 3 for example is represented by one polygon, whose 3D representation is defined by one ST feature type for the floor, one for the ceiling, and one for the outer walls. This way a growing catalog of construction parts can be built and at the same time much flexibility is gained. This object-oriented approach can also be used to define a catalog of several different kerbstones, walls or surface structures. The following section introduces a modeling example which is based on such an object-oriented modeling concept.

2.4 Transferability - Modeling ST models from other cadastral data sources

Using a sample map of a digital cadastre ("Liegenschaftskarte") of Dorningen, North Rhine-Westphalia (Landesvermessung Nordrhein-Westfalen 2007) the general transferability of the ST approach is evaluated and at the same time the modeling workflow is optimized. To provide high flexibility and a usual working environment, a geodatabase is set up in standard GIS software (ArcGIS), which enables the mapping of input feature types to ST objects. To create a ST from the sample map, the included feature types representing boundaries, such as land parcels, land-use boundaries, buildings, or boundaries of transportation objects are selected and stored in a separated data set. The next step encompasses the definition of GroundArea features. As mentioned previously, ground areas are not necessarily represented as polygonal features and have to be built from the input data by creating an area-wide polygon dataset from boundary lines and polygonal features. Both, the GroundArea features derived from the modeling process and the original polyline and polygon input features constitute the ST base geometry. Using point-in-polygon selections and feature attributes, ST feature types are defined and stored as values in the attribute tables. As introduced before, ST feature types are used to describe construction parts, or uniquely textured surfaces and are identified by a unique name, which can be used to link configuration parameters to the base geometries. The feature type 'street boundary' included in the sample map for example, is defined as ST feature type 'kerbstone, beton, DIN EN 1340', because this is a standard kerbstone definition commonly used in construction. The configuration parameters needed for the generation of 2.5D and 3D features are stored in separate tables. The visual appearance of objects is defined by material keys, which are to another catalog holding detailed material information, such as the material type (color, texture material, or shader) and material properties (Path to texture file, width, and height). Thus, modeling a kerbstone with equal dimensions but different material requires only a material-type definition. This setup reduces the complexity of the model by introducing a sharp division of base geometry, geometry generation parameters and material properties of objects, and at the same time a growing database of construction parts and materials can be established.

To create a ST model the material catalog and feature type catalog at present have to be joined to the ST base geometry and exported as shapefiles which are loaded into the editor for visualization and manual post-processing (Figure 4). In contradiction to the City Channel example, this time no digital terrain model is used to derive heights for the base geometry, and thus the object-based terrain model does not depict the 'real' situation, but resides on a flat plane. Approximating the base heights of objects is nonetheless possible and easy if a digital terrain model is available.

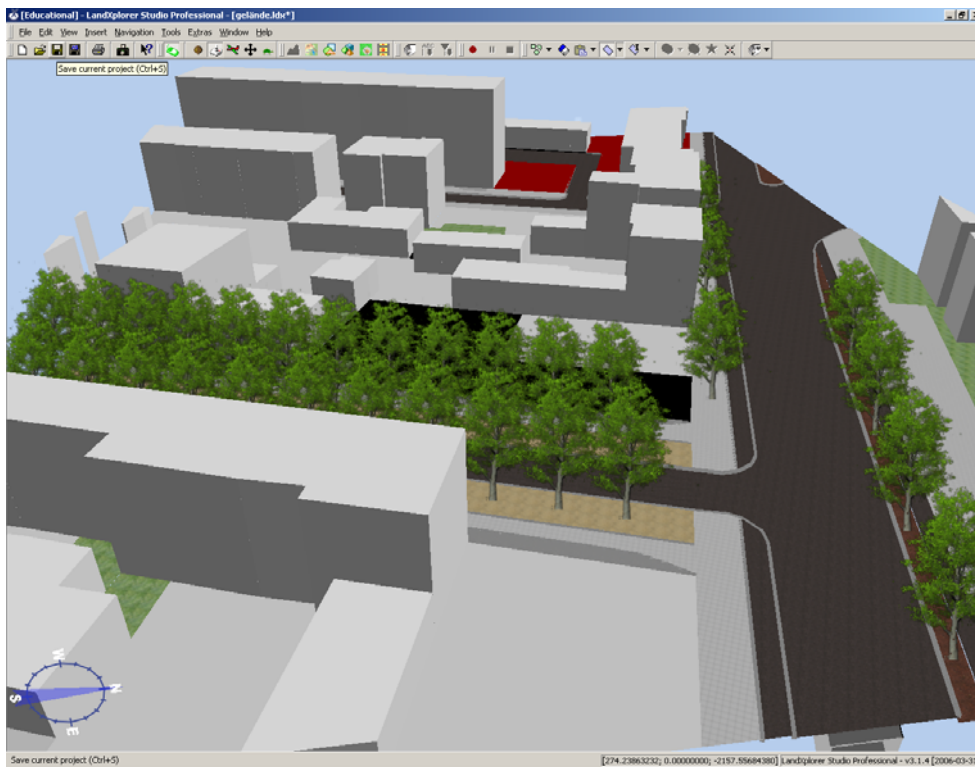


Figure 4: ST model from the example dataset Doringen

3 RESULTS

As a general result from the introduced ST modeling examples the conclusion can be drawn that the method developed is applicable to the creation of detailed 3D object-based terrain models from large scale geo data, such as STGK or official cadastre data. Moreover, if the input data provides a detailed feature type classification and is topologically clean, ST models can be established rather fast.

With respect to interoperability and semantics the initial modeling approach (compare 2.1) does have some drawbacks, because of the ambiguity of the class *GroundArea*, as well as the missing spatial context. This problem has been solved by a mapping function which can be used to transfer the model to CityGML. However, to create a structured dataset, additional attributes, respectively spatial knowledge is needed. Furthermore, the initial class model obviously does not cover all ground-related structures so far. Bridges, tunnels, rails and other structures are not depicted in the model, although they are related to the urban terrain. The design study presented in figure 3 shows that such structures can be composed from the available ST elements. Therefore, an object-oriented modeling concept is introduced, that divides the base geometry from configuration parameters, appearance parameters, and CityGML keys. This object-oriented approach has been chosen to process part of the sample map of Doringen. Although complex objects have not been included in the example, the modeling process is on the one hand facilitated, and on the other hand more flexibility is gained with respect to data update, and the visual properties of objects. This is due to the ST feature-type catalog, which provides a knowledgebase of configured ‘construction parts’. Thus, even if the base geometry of the cadastre is changed, only the ST feature-type keys have to be updated to process an update terrain model. Furthermore, this setup allows creating localized ST models, such as more detailed models made for public participation processes or in models for tourism information systems. In these application areas it might be the intention to depict a certain ‘style’ or ‘design’ of the virtual space, which can be achieved by reconfiguring parameters in or adding feature types and material types to the catalogs. Another improvement of the introduced modeling concept is the early integration of features providing spatial context such as the borders of land parcels. Incorporating such ‘spatial units’ during the modeling of *GroundAreas*, they can be used to select and define aggregation groups needed to structure the data, which is a key requirement to map ST models to CityGML. Figure 5 depicts the introduced modeling workflow and its processing steps.

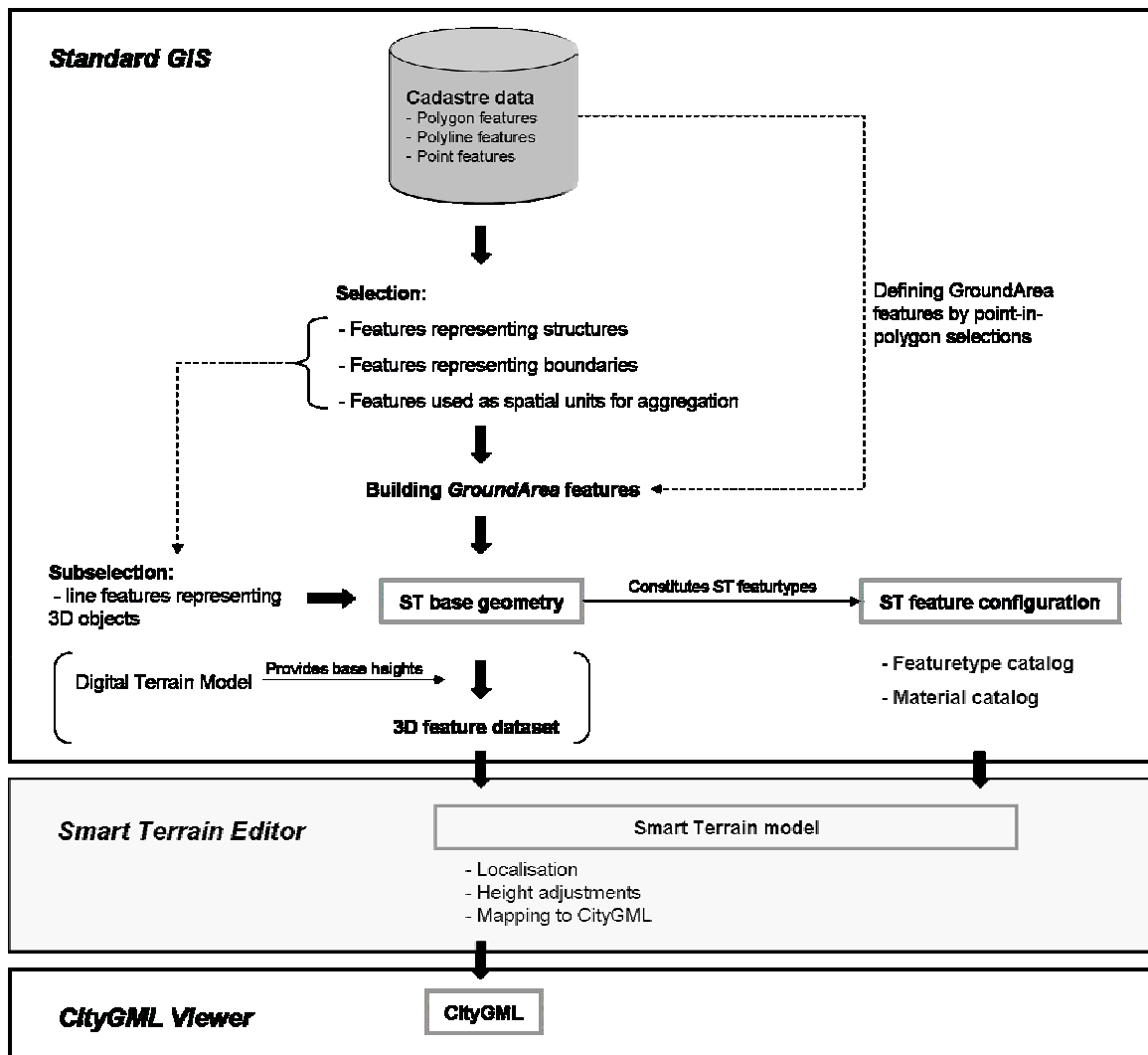


Figure 5: Modeling workflow for the creation of ST models. The graph depicts main steps needed to create ST models. As can be seen from the diagram the modeling is mainly done in standard GIS

The preparatory work is conducted in standard GIS software, which is a key requirement for the usability and acceptance. It ensures that users familiar with geo-data processing can use their usual working environment. The Editor is only used to visually explore the modeling results and to configure feature-specific parameters, such as the alignment of textures and for performing height adjustments. It further provides the export utility to map ST models to the CityGML schema.

Although the general modeling approach seems to be a promising method for the creation of object-oriented 3D urban terrain models, some questions still remain. The most comprehensive and labor-intensive task with the method so far, is the adjustment of heights. Though the editor provides some basic functionality to adjust heights, it does not provide a good solution in all cases. Especially at locations where objects such as retaining walls or a very steep relief can be found, the use of digital terrain models as a source for height information leads to unsatisfying results. Another difficulty is connected with the export function, which at present requires high expertise to configure the objects properly.

4 DISCUSSION & OUTLOOK

In Germany the situation, with respect to the existence of base-data for the generation of detailed object-based terrain models, is good. Classified datasets such as STGK and ALK/ALKIS („Automatisierte Liegenschaftskarte/Amtliches Liegenschaftskatasterinformationssystem“) are in many communes available. At present the migration progress from the old, sometimes still analog, STGK and the official cadastre data to the standardized ALKIS data model (ALKIS 2007) for the storage and management of cadastre data has begun. In general, it appears that the unified class model provided and the feature types and attributes defined will provide enough information to automate, or at least ease the creation process for ST models.

The presented Dorningen example, is much less detailed with respect to visual appearance, but much more detailed with respect to included geo information than the City Channel example. Moreover, if combined with detailed buildings, vegetation and street inventory, it appears that it can be used to create a detailed geovirtual 3D cadastre. But what would be the advantages of a 3D cadastre and how can it be integrated into planning and management processes? We will address these questions during a research project aiming at the development of new 3D land-use information systems. This project is part of the German research network REFINA, which wants to develop tools, methods and best practice examples to reduce land consumption.

Besides this general research questions, a redesign of the ST schemata and the integration of ComplexObjects will be necessary to enhance the smartness of the model. Therefore, we will introduce topology and new keys, that will ease the problems concerned with the mapping to CityGML and the configuration of height information.

5 LITERATURE & RESSOURCES

- ALKIS: Amtliches Liegenschaftskataster-Informationssystem der Vermessungsverwaltungen in Deutschland, Internet: <http://www.alkis.info/> - last viewed 14th March 2007.
- BUCHHOLZ, H., DÖLLNER, J., ROSS, L. & KLEINSCHMIT, B.: Automated Construction of Urban Terrain Models; In: W. Kainz, A. Riedl, G. Elmes (Eds.): Progress in Spatial Data Handling. 12th International Symposium on Spatial Data Handling, Vienna, 2006
- LANDESVERMESSUNG NORDRHEIN_WESTFALEN: EDBS-Musterdaten und Kartenmuster in den "Vorschriften für das automatisierte Zeichnen der Liegenschaftskarte in Nordrhein-Westfalen - Zeichenvorschrift-Aut NRW (ZV-Aut)", download available at: <http://www.lverma.nrw.de/produkte/liegenschaftsinformation/katasterinfo/alk/ALK.htm>, Internet, last viewed: 14th march 2007
- ROSS, L.: Interaktive 3D Geovisualisierung in der Freiraumplanung – Konzepte, Methoden und Verfahren der Modellierung und Visualisierung urbaner Landschaften, 2006 - unpublished
- OPEN GEOSPATIAL CONSORTIUM: Candidate OpenGIS CityGML Implementation Specification (City Geography Markup Language), 2006