GENERALIZATION OF SEMANTICALLY ENHANCED 3D CITY MODELS

R. Guercke, C. Brenner, M. Sester

Leibniz Universität Hannover, Institute of Cartography and Geoinformatics , Appelstr. 9a, 30167 Hannover, Germany
(richard.guercke, claus.brenner, monika.sester)@ikg.uni-hannover.de

KEY WORDS: Generalization, City Models, 3D, Semantics

ABSTRACT:

With the rapid advances in sensor – especially laser scanner – technology and the development of increasingly more sophisticated algorithms for the extraction of features from the data sets produced by those sensors, very detailed digital models are going to be produced for a large number of urban areas. In order to make these models available for different applications, concepts for the generalization of these models have to be developed to reduce the size and semantic complexity of the models to a degree that can be handled by the application without losing information that is relevant for the task at hand. Postulating a stricter separation and modularization of the processes of feature extraction and generalization, we present a workflow for the generalization of semantically enhanced models with a hierarchical structure and describe how such models can be used to integrate different algorithms for the generalization of special constellations of features.

1. INTRODUCTION

1.1 Motivation

With the growing availability of increasingly more detailed 3D city models, the demand for approaches towards their generalization can be expected to rise significantly in the next few years. This trend is going to be supported by an increasing number of applications that will also change the current practice of producing these models mainly for visualization purposes. An example for this trend are the 3D models that have to be used to calculate noise levels for urban areas according to recent EU regulations.

In the context of the German grid computing initiative (D-Grid), the GDI-Grid project is concerned with using grid technology for spatial data infrastructures. Within this project, one task is the development of generic generalization service for 3D city models.

In order to use 3D city models beyond the task for which they were produced, a generalization step is often necessary. Two main reasons can be identified for the necessity of generalization: The size of the data sets is too large to be processed in the application or there is information in the data that is not needed or cannot be handled by the application.

The main purpose of generalization is therefore to reduce the complexity of a data set with respect to its size and semantic content while retaining the pieces of information that are relevant to the task at hand – removing or rearranging those aspects that cannot be handled by the application.

With the concept of “relevant information” being inherently a semantic one, approaches towards generalization that do not (or only marginally) rely on semantic information yield satisfying results only for special applications. One of those cases is, however, the most popular one: (more or less) photorealistic visualization. For this application, the most important criteria for the importance of a feature are its size and reflection properties rather than specifically semantic information – with the term “*semantic information*” referring to the type of the feature and the values of parameters specific to the feature’s type and the application.

In cartography, semantics are modeled through specific feature classes and thematic layers. In the generalization process, semantics were initially introduced by using different mostly geometric operators – like the Douglas-Peucker or Jenk’s algorithm for line features – for different kinds of features. In recent years, however, the relevance of semantic enhancement by structure recognition has been emphasized, and a growing number of specific generalization algorithms have been developed like the one presented in Heinzle and Anders (2007) for road networks.

In the context of the generalization of 3D city models, this is even more necessary because in three-dimensional models there are often constraints that are extremely difficult to ensure on a purely geometrical basis.

A parameter giving the maximum tolerable inaccuracy for a feature is introduced to control the generalization process. This parameter will be referred to as the target resolution assigned to a feature. One way of introducing semantic criteria in the generalization process is to set different target resolutions for different features according to semantic conditions.

An important goal of this work is to make it possible to define the process of generalization in as natural a way as possible. For this purpose, semantics-based generalization approaches are an essential tool. A description like “at a given resolution, a Mansard roof is simplified to a gabled roof of equal height” is much more intuitive – and far less error-prone – than a description like “if there are four roof planes in a certain constellation (Mansard roof) then transform them to a pair of planes (gabled roof) at a given resolution.” Unfortunately, few models with a sufficient level of semantic information to directly apply a rule like the one described in the first example are available at the moment.

For this reason, most approaches towards the generalization of 3D city models that have been presented so far remind rather of the second example: Because they have to use models that contain mostly geometric and comparatively little semantic information, a feature recognition step is introduced implicitly in the algorithms.

This implicit combination of generalization and feature extraction has several important drawbacks: Such algorithms can usually only be used for specific geometrical representa-tions of (conceptually) the same situation (like walls having to be represented as one surface); there are two independent sources for problems; many approaches concerned with generalization end up having spent considerably more effort on feature extraction (structure recognition) than the original task of generalization – which is not surprising considering the fact that feature extraction is a wide field of research in its own right. In our opinion, it is therefore necessary to introduce a stricter separation between the processes of generalization and feature extraction.

An additional improvement of the separation of the steps is the fact that existing feature extraction solutions can be used. Milde et al. (2008) and Ripperda (2008), for example, present projects concerned with the extraction of detailed roof and façade structures from mostly geometric data. Models provided by these approaches contain a high level of semantic information and are therefore promising for semantics-based generalization.

Some generalization operators like typification are defined for groups of features. In order to use these operators in the generalization of data sets in which such group features are not labeled explicitly, algorithms for the detection of recurring or symmetric structures like the one presented in (Bokeloh, 2009) – for laser scanner data – have to be employed before these operators can be used.

1.2 Related Work

The approach of using hierarchical models for generalization has been introduced in (Lal, 2005) in his distinction between micro, meso and macro models for generalization. There are, however, only these three fixed levels in his hierarchy; it is therefore not possible to extend the model towards larger or more fine-grained structures. He also stresses the necessity of a stronger separation of the processes of feature extraction and generalization. The focus of his work is, however, on feature extraction and the specific generalization operation of aggregation.

M. Kada (2007) uses the wall surfaces of a building complex to detect structural parts (cells) of an ensemble of building components. He introduces parametric primitives for roof forms. Using the different roof primitives, regular patterns of roofs can be detected in order to apply the generalization operator of typification. For the general structure of the building complexes, the selection operator is used: If a cell is too small to be retained after the generalization process, it is removed from the model. The generalization approach works on geometric models and consists to a great part of a feature extraction component. It is limited to building models that consist of wall and roof surfaces.

Döllner and Buchholz (2005) introduce the concept of Continuous-Level-of-Quality buildings that allows the user to model buildings with custom granularity according to the task at hand. They do, however, not provide concepts for the automatic generalization of such models. The concepts for generalization introduced in Buchholz (2006) are mostly concerned with visualization issues, especially the treatment of textures.

H. Fan (2009) introduces an approach to extract the exterior shells of building models that contain interior and exterior surfaces for walls and roofs – with the generalization step consisting of replacing the original geometry by the exterior shell. Additionally, different strategies for the generalization of (regular arrays of) windows are evaluated.

There are lots of techniques for the reduction of polygon meshes for visualization from the computer graphics community. These approaches are, however, not designed to make sure that resulting models fulfill semantic constraints. For this reason, employing such models for the generalization of city models often results in mostly geometry-based approaches with semantic constraints introduced implicitly. An example for such an algorithm is the approach of Rau et al. (2006) in which rules for the detection of protrusions are introduced implicitly in rules for the collapsing of walls.

Thiemann and Sester (2004) also present an approach towards the generalization of 3D city models: The roof and wall planes in the model are used to derive a CSG representation of the building. The generalization step is a selection that is employed by removing those primitives from the representation that are too small for the given resolution.

2. A GENERIC GENERALIZATION SERVICE

2.1 A Workflow for the Generalization of 3D City Models

In the development of a generic generalization service, it is impossible to predict all requirements and peculiarities an application may introduce – especially if the purpose is not only visualization. In a flooding application, for example, it is possible that upright surfaces (like walls) cannot be used directly in FEM-based simulation software.

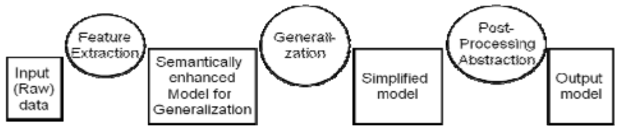

Figure 1: Generalization workflow

In a typical generalization scenario, the three different steps shown in Figure 1 can be identified: Feature extraction (possibly from different sources), the generalization step itself, and a post processing step to adjust the output data to the application.

In this context, the term *feature extraction* refers to the process of deriving information that was not explicitly modeled in the input.

A great increase in reusability can be gained through the modularization of different concepts in the fields of feature extraction and generalization. One interesting scenario in this context could, for example, be to use the algorithm for the extraction of the exterior shells from Fan et al. (2009) in order to prepare models for the partitioning into cells introduced by Kada (2007).

Feature extraction and generalization are very closely related, and in order to enhance the quality of the output of the generalization process, it can be reasonable to introduce further feature extraction steps – especially for the identification of patterns.

Such a nested feature extraction step can, for example, make sense in the context of an arrangement of similar but slightly different features in a regular pattern. At maximum resolution, these numerous small differences can make it problematic to model these features as a group. At lower resolutions, however, the differences may be irrelevant and to collect the features in a pattern offers new possibilities for generalization (like typification).

Such cases are, however, not arguments against the separation and modularization of the different generalization and feature extraction strategies but rather in favor of this because only atomic services can be combined with the necessary flexibility.

2.2 Usability

For a wide range of users, standard feature types and generalization strategies with the possibility to request special features at different resolutions depending on application data will be sufficient for most feature types. For this reason, the framework is going to provide a standard feature model with configurable generalization options.

A generic service can, however, not predict which influence the values of application-specific variables have on the importance of a feature.

For this reason, the user can configure the generalization process using semantic and spatial criteria. An example for such a generalization query could be “Give me all features within 1000m of the river with the name ‘Mississippi’ at a resolution of 2m, the bed of the river at 1m, and the rest of the model at 5m resolution”. If the data does not contain the required amount of detail, the user has to choose between either using the most detailed version available or aborting the process. In the simple standard case, a uniform resolution is set for all features.

Due to the modular approach in the design of the generalization process, the user can also choose between different generalization approaches for different features that could be mapped (if appropriate) to the different features in the same way as the different resolutions.

Unfortunately, there are features of special importance for different applications that can require custom generalization procedures. In order to deal with this problem, a third party is necessary: The developer of application-specific generalization components. In order to support the development of specific feature types and generalization procedures, the framework offers a standard model with the possibility of inheritance. A dike may, for example, be defined as a special type of ridge with parameters describing the construction and the pattern of breakwaters. The user can then, for example, choose between the dike generalization algorithms of developers A and B.

3. A SEMANTICALLY ENHANCED BUILDING MODEL FOR GENERALIZATION

3.1 Explicit and Parametric Modeling of Geometry

To represent geometric information explicitly in a model has the advantage that it is easy to extract this information without knowledge of the more specific semantics of the model. For this reason, the explicit modeling of geometry is popular with exchange formats. In many cases, however, the explicit modeling of geometry obscures semantic information. Two planes in a building model may, for example, form a gabled roof or be two opposite walls in the body of the building.

The CityGML model presented in (Kolbe et al., 2005) uses a semantics-based feature hierarchy with an explicit representation of geometry in the leaf features. This may lead to the inconsistencies introduced above if the modeler does not take care to avoid them.

If (geometrically relevant) semantic and explicit geometric information are combined in a model they can be redundant – with the ensuing problem of possible inconsistencies. If, for example, a roof is stored labeled as a gabled roof and the geometry corresponds to a flat roof, the semantic and geometric information are inconsistent.

The advantage of the parametric modeling of geometry is that operations on the geometry can be described in a more abstract way using semantic concepts and that constraints can be satisfied implicitly. This reduces the complexity of the generalization process and of ensuing integrity tests considerably. For this reason, parametric representations have been chosen for most feature types in the reference model. It is, however, possible to use both ways of representing geometry in the model.

Depending on the application, different geometric representations can be derived from the same model – a wall that is known to have certain thickness can, for example, be instantiated as a solid, as its two visible faces or as a single face.

Geometrical tests like intersection tests for the detection of conflicts introduced by overlapping features can never be avoided completely but the goal of using the parametric model is to reduce their number and complexity as far as possible.

3.2 A Hierarchical Feature Model

In order to deal with the complexity of 3D city models and the necessity to give users the possibility to extend the model by their own types, the reference model is organized hierarchically with a parent-child relation meaning that the child is part of the parent feature.

This makes it possible to deal with features of different granularity and to ensure constraints implicitly. Another advantage is the fact that the effects of the introduction of additional feature types can be limited to specific parts of the feature hierarchy.

For each feature in the hierarchy, a bounding box and a transform are stored – when a new child is added to a feature, its bounding box is updated to make sure that all children are enclosed in its bounding box. Using this information, the feature hierarchy offers possibilities of a scene graph like using local coordinate systems for the description of a feature and of the R-Tree data structure because search queries can be pruned if a feature’s bounding box (and through the containment relation, all of its subfeatures) does not intersect with the search interval.

For the detection of overlaps of features in the process of the identification of conflicts during generalization, these bounding boxes can also be used for quick tests whether conflicts can occur and to define areas occupied by a certain set of features that must not be intersected by others in order to avoid conflicts.

As in cartographic models, thematic layers for features from different thematic fields can be introduced. In the CityGML model, for example, there are layers for water bodies, buildings, traffic objects and vegetation features. The current version of the proof-of-concept prototype consists only of building-related features. It is, however, planned to incorporate traffic objects like roads in one of the next steps.

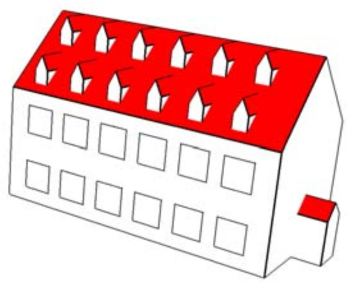

Figure 2: Example of a building model

Figure 2 shows a simple model of a building; the structure of this model is shown in Figure 3. The top level feature is a BuildingPart object with a gabled roof and an annex (modeled as another BuildingPart object) attached to one of its sides. There is an array containing two rows of windows with six windows in each row attached to one of the walls (the first one with index 0) and an array of six by two dormers attached to the roof. Both of these arrays are represented by the same 2DArray class with different template features for the windows and dormers and subjected to the same generalization procedure of typification in the generalization process.

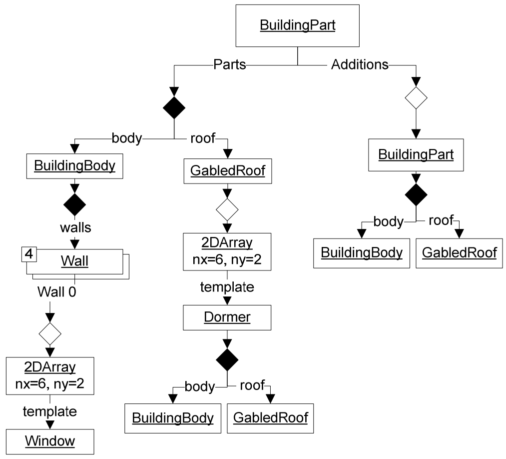

Figure 3: Structure of the model in Figure 2

The rhomboid shapes in Figure 3 symbolize the two different kinds of subfeatures a feature can have: The filled rhomboid stands for the essential parts of the feature, the empty ones for non-essential additions. This distinction is important for the generalization of the model because necessary parts of a feature should not be removed from the generalized model if the parent feature has been decided to be kept. In special features that represent arrangements of features, template features are used to model the features that appear in the different cells. As for all other features, the treatment of these special features in the generalization process depends on the application. In a typification strategy, for example, they would be emphasized to match the current resolution.
This notation was chosen to remind of the UML notation for the aggregation and composition relations because it has a similar meaning. It does, however, work in the opposite direction: In the UML definition, an aggregation relation is a composition if the parts do not make sense without the whole while in the model the whole is not valid without the essential parts.
Figure 3 shows another characteristic property of the model: The structure of the model represents an interpretation of the situation. One example with direct impact on the generalization process is the arrangement of the features in group or array features. The arrays of windows or dormers may as well have been represented as two independent rows – meaning that they are being considered independently in the standard generalization process which would lead to their being removed from the generalized model at much earlier steps. Such restructurings can be used in the harmonization and optimization steps in order to resolve conflicts or enhance the result of the generalization.

4. WORKFLOW FOR THE GENERALIZATION OF HIERARCHICAL CITY MODELS

4.1 Overview

The generalization process is implemented as a depth-first traversal of the feature hierarchy according to the process model shown in Figure 4.
In the first selection process, the feature is tested if it qualifies to be retained in the generalized version of the model. If this is not the case, the process terminates and nothing is returned. Common criteria are the size and type of the feature: Usually, a minimum size is given for a feature depending on a target resolution. This minimum size may vary for different types of features; additionally, special types of features can be excluded if they are not relevant for the purpose for which the generalized model is produced and other types of features can be enforced to be kept if they are of special relevance.

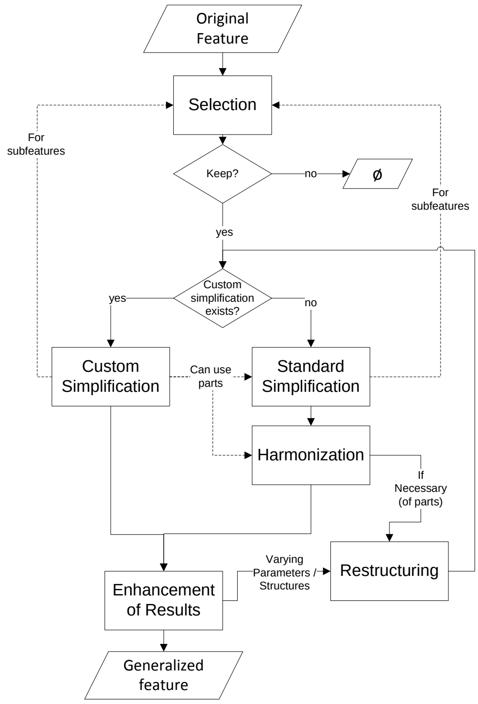

Figure 4: Process model for the generalization traversal

If a feature is an essential part of a feature that has already been decided to be kept, it is retained without having to pass the selection test.
In the first pass, the restructuring unit is not going to change the model. In the following processes, however, it may turn out that using a structurally slightly different (but semantically equivalent) representation of the situation described by the model yields considerably better results – or may be necessary to get a valid result at all: The restructuring step includes, for example, the setting of annotations that prevent special simplification steps that have been discovered to lead to inconsistencies in the model. A more detailed description of the restructuring step can be found in section 4.2 of this paper.
In the next step, it is tested whether custom simplification procedures exist for the current feature. If no such procedure is found, the standard simplification is applied: In the first step, the general structure of the generalized feature is constructed by applying the whole process recursively to the essential parts of the current feature – with the selection decision always being positive – and reassembling them to form the new one. After that, the additions are also subjected to the process and attached to the new feature – if they have been decided to be kept in the selection step.
Because the different parts are generalized independently, conflicts can occur that have to be resolved by the harmonization component. Such a harmonization step can require – possibly repeated – partial restructurings of the model with the subsequent new simplifications.

Custom simplification procedures can be defined for individual features – which is, of course, a very laborious thing to do for all features but can make sense for special features like buildings of very unusual architectural design – or for classes of features. Generally, the most specific simplification procedure is applied: If there is one for the individual feature, it takes precedence over all others; procedures for the class of the feature are applied before procedures of superclasses are considered. For features with different superclasses – or more than one implemented interface – the precedence can be defined in the selection component.
In principle, a custom simplification procedure has complete control over the way it generates the generalized feature. It is, however, possible to use components from the standard simplification. This is especially useful in order to deal with additions: Only in rare cases will a developer of a custom generalization procedure have to deal with all possible additions to a feature – especially as anyone might define a new feature class and want to use an instance of it as an addition to a feature of the class for which the custom simplification procedure is developed. For this reason, it makes sense to reuse the standard simplification approach of independent simplification and harmonization at least for the rest of the feature's additions for which no specific treatment has been specified.
In order to achieve better results, the whole process can be embedded into an optimization step. In such an optimization process, different restructurings of the model and different parameters for all parts of the simplification process can be tested and evaluated against each other.
Usually, a trade-off has to be found between the desired quality of the generalized model and the amount of processing resources available. In order to cover as many application scenarios as possible, a configurable system like the blackboard approach or a formal grammar is going to be used in the prototype.

4.2 Restructuring

The same situation in the real world can be described by different building or city models. Such models will be called *semantically equivalent* while identical models are not only semantically but also *structurally equivalent.*
Especially for constellations of similar features, different semantically equivalent models can be defined in which these features are grouped in different patterns or hierarchical structures.
Because the structure of the model is essential for its generalization, using a structurally different (but semantically equivalent) version of a feature can significantly enhance the result of the generalization – it may, for example, be possible to collect features in a regular array if they are similar enough after their simplification.
As indicated in the process model shown in Figure 4, such restructuring operations can be employed for two different reasons: To resolve conflicts that occur due to the independent generalization of sibling features (from the harmonization step) and in order to find the interpretation of the original model that yields the best results after generalization (in the optimization step).
As a simple case, restructuring operations can also be used to exclude certain generalization operations. Such measures can be necessary if, for example, a gabled roof has a group of dormers: It makes sense to prevent it from being generalized to a flat roof if the dormers have been decided to be kept at the given resolution.

Finding semantically equivalent but structurally different versions of a model is a graph rewriting problem: Parts of the feature tree defined by the current feature that can replaced by other features have to identified and replaced by the appropriate new features.
Unfortunately, the general graph rewriting problem is NP-complete. It is therefore not a promising approach to try all possible sets of structurally different representations of a detailed city model that may contain a lot of features.
For this reason, restructuring steps are used only over limited parts of the feature tree; mostly only between the direct children of the current feature. For nested patterns like the more sophisticated facade model introduced in (Ripperda, 2008), it can be useful to descend deeper in the feature hierarchy to get better results.
A substantial alleviation of the general complexity of the problem is the fact that – usually – only patterns of features have to be considered that start directly below the current feature in the hierarchy.
As mentioned in section 2.1, one way to find different interpretations of a model is to employ feature extraction algorithms from different sources on a collection of already simplified subfeatures in order to find patterns that were obscured by slight differences between the features in the original model.
Another approach is to directly define valid transformations for special constellations of nodes in the feature tree.

4.3 Pattern Features

In order to make the generalization process more transparent, pattern features can be introduced to explicitly model characteristic constellations of features.
Using these patterns, operators from cartographic generalization like typification can be defined as simplifications of pattern features.
A special kind of these pattern features are group features in which similar features are arranged in a more or less regular distribution along a line or in a grid. For those features, the cartographic operator of typification can be introduced as a simplification procedure.
In the current prototype, a class representing a regular array of features is included. Features of this class have a template feature that is copied to all cells in the reconstruction of the geometry. Additionally, the number of features and their distance in x and y direction on a virtual canvas are given; the height offset of the features is associated to a virtual height field defined over the canvas. This is necessary because otherwise dormers might, for example, end up below the roof surface.

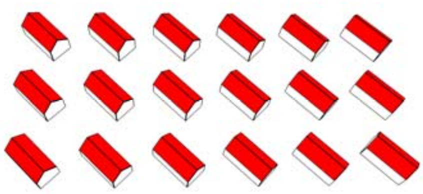

Figure 5. Regular array of rotated buildings

It is possible to specify a transformation for the template feature that is applied after the feature has been moved to its place in the grid defined by the array feature. Such a transformation can be used, for example, to model features – like the buildings shown in Figure 5 – that are arranged in patterns but rotated or otherwise not aligned with the main axes of the distribution defined by the pattern.

For "letter" patterns like "I", "T", "E", etc. – for example of buildings in a block as introduced in (Rainsford and Mackaness, 2002) –, the different parts of the letters can be defined as essential parts which may be simplified but not omitted outside of a specific simplification procedure.

4.4 Models in Scale Space

So far, only point queries in scale space have been described in this paper: It was assumed that one value of the resolution parameter was specified for each feature.

There are, however, applications for which it can be expected that different distributions of resolution values over the features will be needed.

A typical example of such a scenario is the derivation of models for visualization: According to the position and orientation of the camera, different features will have to be represented at different levels of detail. Especially in this case, generating a generalized model on the fly – especially if harmonization and optimization backtracking steps are necessary – is going to lead to unacceptable delays.

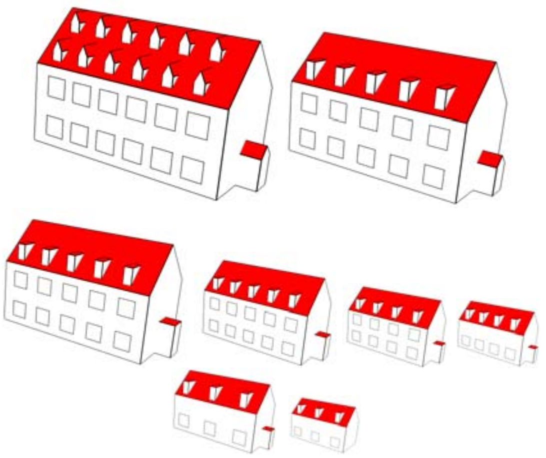

Figure 6: Steps in the generalization of a building model

As long as a uniform resolution is chosen for the whole model, such models with "precompiled" generalization levels can simply be stored as a sequence of generalization steps – with the resolution at which they occurred – inverted to a refinement series; this approach is similar to the "streaming generalization" introduced in (Sester and Brenner, 2009).

Figure 6 shows such a generalization sequence for the model introduced in chapter 3.2. The development of the incremental model is a sequence of events in the reverse order of those that occur in the generalization sequence with the first representation being the least detailed model – in this case, with a gabled roof, three windows and three dormers. If a more detailed representation is required, details are added until the desired accuracy is reached. In the example, the second step is to restore the small annex followed by the appearance of a fourth element in the rows of dormers and windows. In the succeeding steps, elements are added incrementally to the arrays of windows and dormers and the flat roofs of the smaller structures are replaced by the original gabled ones. If the requested accuracy is higher than the one associated with the first generalization (last refinement) step, the original model is returned.

If non-uniform resolutions are required, the problem becomes more difficult because very complex conflicts can arise in this case – especially if the difference in the required accuracy is motivated by the semantics of the model rather than visualization and there are close relations between parts that are needed in more and less detail.

4.5 Harmonization

There are two classes of problems that have to be resolved by the harmonization component: general and feature(type)-specific problems.

Most general problems are the result of the fact that features can change their shape in the generalization process.

In the course of such changes – especially in the context of typifications –, features may be moved or emphasized with the result that they cover areas on which they had no impact before. If there is a feature in such an area with which the generalized feature is not supposed to have a non-empty intersection, there is an overlap problem. Such a problem can be solved in several different ways: According to the relative relevance of the features, one of them may be moved, reduced in size (if possible) or omitted. Another approach is to restart the generalization of the feature of less importance with a constraint defining "no-go areas" in those locations that are occupied by parts of the more important ones. Combinations of these approaches and new ones can be tested in the optimization step or implemented directly for special situations.

Parent-induced problems occur if changes in the shape of the parent feature affect one of its child features. If a roof, for example, changes its shape from a mansard roof to a gabled roof, a dormer attached to it might end up "dangling" in the air. In other cases, it might be "drowned" below a "rising" roof surface. These problems can often be solved by a displacement of the child feature.

There are, however, cases that are more specific to special feature types: it may, for example, not be desired to keep dormers if a previously sloped roof is simplified to flat roof. In such a case, it must be decided in the harmonization step if the critical simplification step is not executed or the affected features can be changed or omitted. In order to deal with these specific conflicts, individual rules have to be specified in the harmonization component.

Difficult problems can occur if closely related features are required at different resolutions, especially if a subfeature is needed at a higher degree of accuracy than its parent feature – for example, special architectural details like gargoyles on the eaves of a cathedral's roof. In order to resolve such conflicts, more complex measures may have to be taken. It can be necessary to retain those parts of the less detailed features that are in conflict with the more detailed one at the same level of detail.

In the further course of our research, representations for such "precompiled" models with the option of non-uniform resolutions are going to be developed together with strategies for their automatic construction from a given more detailed model.

5. CUSTOMIZATION

There are several different ways in which the model and the generalization process can be customized to meet the demands of an application.

The model can be extended by the definition of application-specific attributes for existing features. It is also possible to introduce user-defined feature classes either as subclasses of existing ones or as independent classes.
There are different ways in which all parts of the generalization process can be customized. The easiest way is to define varying target accuracies according to the relevance of the different features. For the distribution of resolutions over the features, a query language similar to the XQuery specification is going to be developed that allows the user to address parts of the feature tree – with the possibility to define filters based on features types, application data and geometrical relationships – and assign a target resolution to the features in the selection. This includes the possibility to force a set of features to be omitted or kept in the generalization process.
In principle, all parts of the generalization workflow can be changed to be better suited for a given application. In the course of the development of the model and the generalization components, the focus will be on the implementation of frameworks with configurable default procedures and the possibility to define custom procedures for those parts of the model or generalization where the application demands it – offering the developer the possibility to fall back on existing components at every step. Using such a mechanism, a developer may, for example, develop generalization processes for the generalization of building ensembles leaving the generalization procedures for features below the level of individual walls and above the one of building blocks to the standard component – or one developed by a colleague.
Additionally, different approaches for the control components of harmonization and optimization can be implemented and evaluated for different applications.

6. CONCLUSIONS AND OUTLOOK

In this paper, a process model for the generalization of semantically enhanced city models has been presented. In a proof-of-concept prototype, parts of the feature hierarchy and the process model for generalization have been implemented.
It has been motivated that rather because of than despite the potential for different interpretations of the same situation, explicitly semantics-based approaches are considerably more promising for generalization than more or less geometry-centered ones. For this reason, a stricter separation of the processes of feature extraction and generalization is postulated in order to facilitate the development of generalization algorithms that do not depend on specific geometrical representations of a model and to allow researchers to concentrate on generalization rather than feature extraction.
In the next steps, a default model for a wider range of feature types is going to be developed. For this model, the implementation of the process model is going to be extended to cover all components proposed in this paper.

7. ACHKNOWLEDGEMENT

This work was funded by German Federal Ministry of Education and Research in the context of the GDI Grid project.

8. REFERENCES

M. Bokeloh, A. Berner, M. Wand, H.-P. Seidel, A. Schilling, 2009. Symmetry Detection Using Line Features, Computer Graphics Forum (Proceedings of Eurographics)

H. Buchholz, 2006. Real-time Visualization of 3D City Models, PhD thesis, Universität Potsdam

J. Döllner and H. Buchholz, 2005. Continuous Level-of-Detail Modeling of Buildings in 3D City Models. In GIS '05: Proceedings of the 13th annual ACM international workshop on Geographic information systems, ACMPress, New York, NY, USA, 173–181.

H. Fan, L. Meng, M. Jahnke, 2009. Generalization of 3D Buildings Modelled by CityGML, Advances in GIScience: Proceedings of 12th AGILE Conference on GIScience, Lecture Notes in Geoinformation and Cartography, p. 387-405, Springer, Berlin, 2009.

F. Heinzle and K.-H. Anders, 2007: Characterising Space via Pattern Recognition Techniques: Identifying Patterns in Road Networks In: Generalisation of Geographic Information: Cartographic Modelling and Applications, Mackaness, W., Ruas, A., and Sarjakoski, T. (ed.), Published on behalf of the International Cartographic Association by Elsevier, p. 233-253.

M. Kada, 2007. 3D Building Generalisation by Roof Simplification and Typification. In: Proceedings of the 23th International Cartographic Conference, Moscow, Russian Federation.

T. H. Kolbe, G. Gröger and L. Plümer, 2005. CityGML: Interoperable access to 3D city models. In: First International Symposium on Geo-Information for Disaster Management GI4DM.

J. Milde, Y. Zhang, C. Brenner, L. Plümer and M. Sester, 2008. Building Reconstruction Using a Structural Description Based on a Formal Grammar, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. XXXVII, Beijing, China.

J. Lal, 2005. Recognition of 3D Settlement Structure for Generalization, PhD thesis, Technische Universität München, 2005

Rau, J., Chen, L., Tsai, F., Hsiao, K. and Hsu, W., 2006. LoD generation for 3D polyhedral building models. In: Advances in Image and Video Technology, Springer Verlag, pp. 44–53.

Rainsford, D. and Mackaness, W., 2002. Template Matching in Support of Generalisation of Rural Buildings. In: D. Richardson and P.v. Oosterom (Editors), Advances in Spatial Data Handling. Springer, Berlin, pp. 137 - 151.

N. Ripperda, 2008. Determination of Facade Attributes for Facade Reconstruction, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 37, no. B3a, p. 285-290.

M. Sester, C. Brenner, 2009. A vocabulary for a multiscale process description for fast transmission and continuous visualization of spatial data, Computers & Geosciences, In Press, Available online 20 February 2009,
DOI: 10.1016/j.cageo.2008.11.003.

F. Thiemann and M. Sester, 2004. Segmentation of Buildings for 3D-Generalisation, Proceedings of the ICA Workshop on generalisation and multiple representation, Leicester, UK.